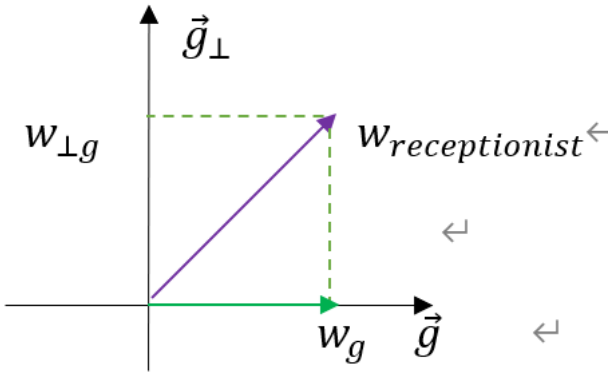
关于消除词嵌入向量的偏向性的解析

本文针对Bolukbasi的论文Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings中关于neutralize和equalize这两个过程做详细说明，论文中对这两个过程解释的并不是很详细，尤其是equalize缺少详细的推导过程，仅仅给出换算公式，我们具体的解释下这两个过程，并且重点对equalize的数学公式做详细的解答。并给出代码。

我先解释下去嵌入向量的偏向性的概念，大家就会明白为什么要进行除偏的操作。我们已经知道在NLP中我们经常使用词嵌入向量来帮助我们解决一些问题。但是有学者发现经过词嵌入的向量带有一些明显的偏向性。我们举个例子来说，比如我们再处理一个类比问题，父亲对应工程师这个词，母亲对应的却是接待员，这就有明显的性别歧视问题了，也就是产生了性别的偏向性。当然也存在其它的一些偏向问题，比如家庭出身，政治背景等等。所以针对这个问题，Bolukbasi的论文通过neutralize和equalize这两个算法帮助我们解决了这个偏向问题。下面我们就针对性别偏向来举例解释说明消除偏向性的操作。

首先我们先解释下neutralize的过程，neutralize的思想就是将一些带有性别倾向的向量进行中和，是他们具有中立特点。我们先看下在执行neutralize之前的词嵌入向量的状态。我们绘制了一个图像来帮助我们更好的解释这个问题。我们无法绘制出几十维到几百维的词嵌入向量空间，我们仅仅用一个二维的坐标轴来说明问题，其实原理和高维度是一样的。请看下我们绘制的还未进行neutralize操作时，词嵌入向量的状态，请看下图：



before neutralizing

我们先解释下这个图像的数学符号，也是我们整篇文章都会使用这些符号。首先$\vec{g}$表示性别偏向的方向坐标轴，男性偏向负方向，女性偏向正方向。$\vec{g}_\perp$指的是与性别偏向方向正交的其它轴的方向，为什么是正交呢？因为词嵌入向量中的每个维度都是线性无关，正交反映了它们之间的属性是完全独立的。所以我们经常会用两个向量的点积去衡量两个向量的关系。其实就是看它们之间的夹角，通过余弦值反映出来，这些基础知识大家都应该了解。$w_{receptionist}$表示的是接待员这个向量在嵌入词空间的状态。$w_{\perp g}$是$w_{receptionist}$在$\vec{g}_\perp$上的投影。w_g是$w_{receptionist}$在$\vec{g}$上的投影。好了数学符号的相关表示就完成了。下面我们讨论下 neutralize 是如何操作的。

我们明显看到$w_{receptionist}$带有性别偏向，它更加偏向女性。所以我们要想办法让这个词尽量中性一些。那么如何做到呢。很简单，就是将$w_{receptionist}$投影到$\vec{g}_\perp$上面。即是下面的关系：

$$w_{receptionist} = \quad w_{\perp g}$$

这样$w_{receptionist}$在$\vec{g}$上的投影就是 0 了，这就解决了性别偏向的问题了，这个词

也就变为中性了，是不是很简单啊。那么具体如何计算呢。请看下面我们给出的计算公式。

$$\vec{g} = \frac{1}{N}\sum_{i=1}^{N} w_i^{\text{female}} - w_i^{male} \qquad (1)$$

$$w_g = \frac{w_{receptionist} \cdot \vec{g}}{|\vec{g}|^2} * \vec{g} \qquad (2)$$

$$w_{\perp g} = w_{receptionist} - w_g \qquad (3)$$

这样$w_{\perp g}$就是我们执行了 neutralize 后，得到的新的关于接待员的新向量。我们具体解释下这几个公式。公式 1 中是求出$\vec{g}$，也就是性别偏向的方向。我们用带有明显女性属性的词向量去减去带有明显男性属性的词向量，得到的差值再进行平均。比如说母亲减去父亲，姐姐减去弟弟，祖母减去祖父等等。

公式 2 稍微复杂一点，是要求出$w_{receptionist}$在$\vec{g}$的投影。那么这个公式如何推导出的呢。请看下面的表达式：

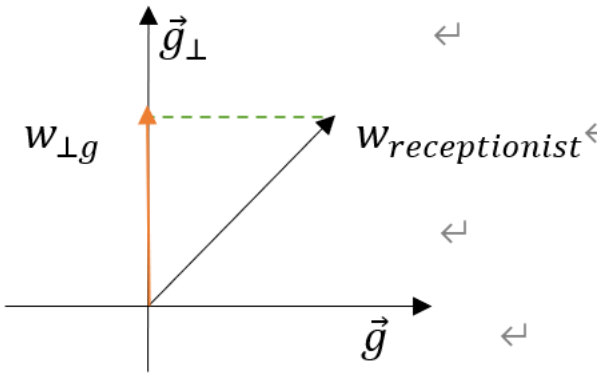
$$w_g = \frac{|w_{receptionist}| * \cos\theta}{|\vec{g}|} * \vec{g}$$

这个公式就比较好懂了吧，θ是$w_{receptionist}$和$\vec{g}$之间的夹角$|w_{receptionist}|$表示$w_{receptionist}$的模，也就是向量的大小。$|w_{receptionist}| * \cos\theta$就是$w_{receptionist}$在$\vec{g}$上投影的模的大小，再除以$|\vec{g}|$就能得到比例关系了，然后这个比例关系乘上$\vec{g}$

自然就是w_g了。那么它是如何转换成公式 2 的形式的呢。其实只要在分子和分母上同时乘上$|\vec{g}|$就可以了，我们再看下乘上$|\vec{g}|$的式子。

$$w_g = \frac{|w_{receptionist}| * |\vec{g}| * \cos\theta}{|\vec{g}| * |\vec{g}|} * \vec{g}$$

这个$|w_{receptionist}| * |\vec{g}| * \cos\theta$不就是$w_{receptionist}$和$\vec{g}$的内积嘛(点积)。那么就转换成就是公式 2 的表达方式了。至于公式 3 就更简单了，就是向量的和差关系。那么我们对 neutralize 的整个推导过程就很清楚了。我们看下执行完 neutralize 后向量的状态，看下图：



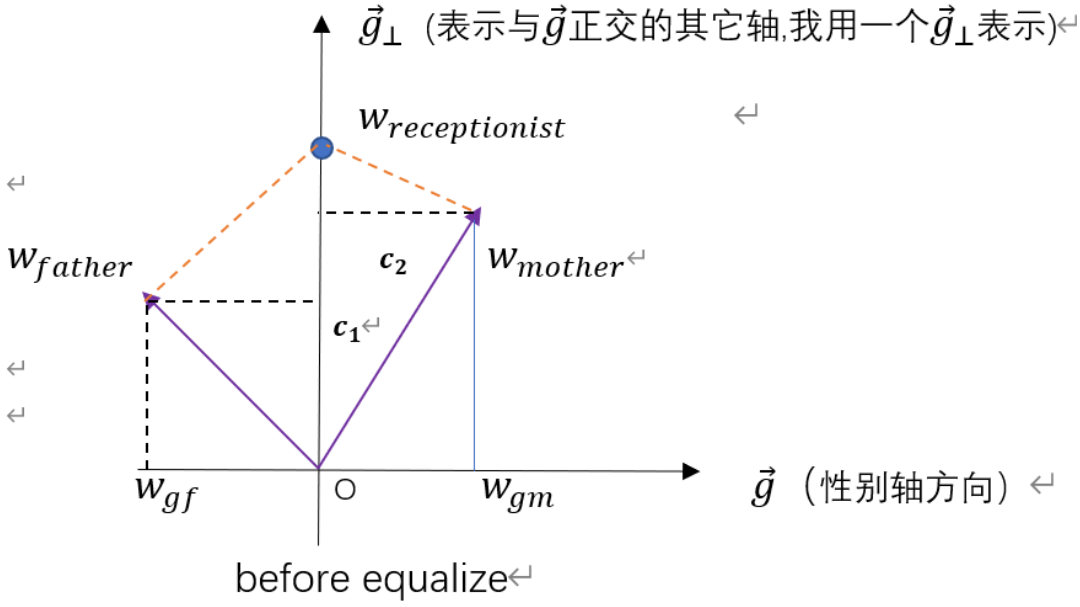
after neutralizing

很显然这样操作后$w_{receptionist}$转换成$w_{\perp g}$，实现了将性别中和的效果。我们再看下代码如何实现的，我们使用 python 来描述算法。我们定义一个 neutralize 方法，该函数实现将词嵌入向量投影到与某偏向轴正交的空间上，从而实现了词的中立性，消除了偏差。具体代码如下：

```
def neutralize(word_vec, bias_axis):
"""
    通过将“word”投影到与偏置轴正交的空间上，消除了词的偏向性。
    该函数确保这些词在性别等偏向轴的子空间中的值为 0

    参数：
        word_vec --  待消除偏差的词向量
        bias_axis --  维度为(ndims,)，对应于偏置轴（如性别）
    返回：
        e_debiased --  消除了偏差的向量。
    """
    #  根据公式 2 计算词向量在 bias_axis 上的投影
     w_bias = (np.dot(word_vec, bias_axis) /
                    np.square(np.linalg.norm(bias_axis))) * bias_axis
     e_debiased = word_vec - w_bias
     return e_debiased
```

接下来我们要具体谈谈本文的核心部分 equalize 操作了。我们首先还是要解释下 equalize 的概念以及为什么要执行这个操作。我们上一步通过 neutralize 操作将带有偏向色彩的词向量中立化。使它不会偏向某一属性，我们举了性别这个属性为例子。我们继续以性别这个属性为例子来讲解 equalize。我举个例子来说明 equalize 的作用。请看下图：



这是描述没有执行 equalize 操作的词向量在线性空间中状态。我来具体解释这个图像描述的含义。$w_{receptionist}$是图中的那个蓝色的点。还记得我们上面讲解 neutralize 时候，把这个向量投影到和$\vec{g}$正交的其它轴上，从而将$w_{receptionist}$这个向量中立化了，使它在性别上没有偏差。这个蓝色的点表示的就是$w_{receptionist}$投影所得的向量。w_{father}和w_{mother}是父亲和母亲的词嵌入向量，就是紫色的那两条线，我们发现w_{father}和w_{mother}这两个向量到$w_{receptionist}$的距离是不等的，明显w_{mother}到$w_{receptionist}$距离要大。这说明母亲比父亲更适合做接待员或者服务员，这又是产生了偏差，性别再一次遭到歧视了。说明我们之前通过 neutralize 并不能完全解决性别偏差的问题，那么如何解决这个问题呢，这就用到了我们接下来要讲解的 equalize 算法了。接下来正题开始。

Equalize 通过重新调整w_{father}和w_{mother}这两个向量在空间中的位置，使它们到$w_{receptionist}$距离相等，同时要w_{father}到$\vec{g}_\perp$的距离和w_{mother}到$\vec{g}_\perp$的距离也相等。这样是不是就起到了均衡作用了啊，接待员这个职业就不会倾向于某个性别了。这个很好理解吧。论文的作者给出了如何计算新的w_{father}和w_{mother}，难点在

于虽然我们很好的理解了如何做才能均衡他们这些词向量，但是作者给出的公式并没有很详细的给出这些公式的推导过程。我们接下来就详细的说明下这些公式产生的论据。我先把公式罗列出来：

$$\mu = \frac{1}{N}\sum_{i=1}^{N} w_i \qquad (4)$$

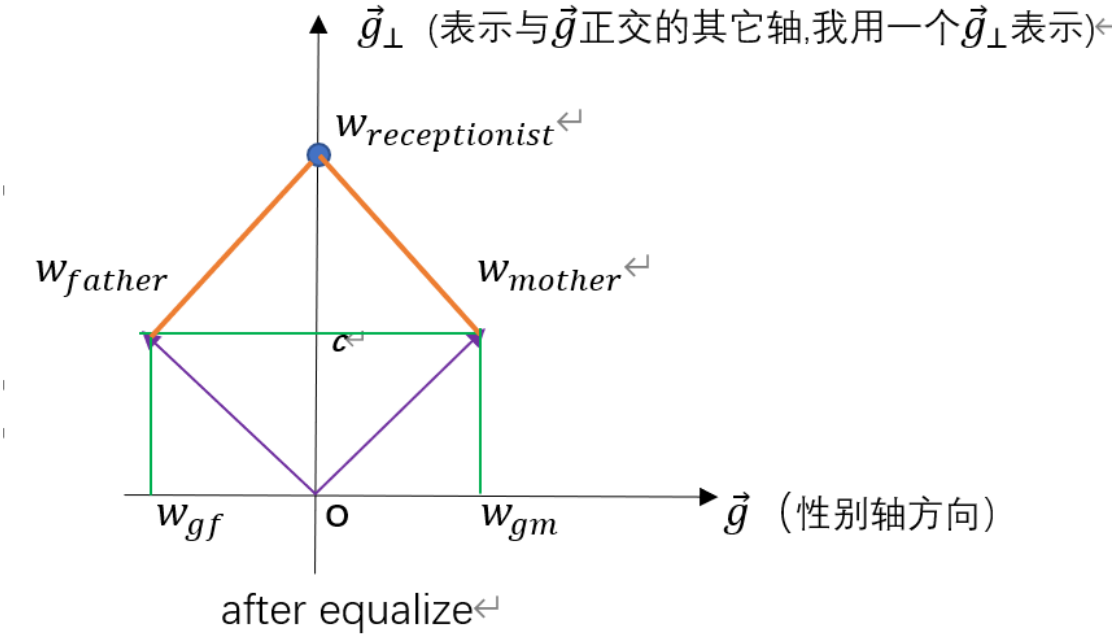
$$\mu_g = \frac{\mu.\vec{g}}{|\vec{g}|^2} * \vec{g} \qquad (5)$$

$$\mu_\perp = \mu - \mu_g \qquad (6)$$

$$w_g^{equalize} = \sqrt{1 - |\mu_\perp|^2} * \frac{\vec{w}_g - \mu_g}{|\vec{w}_g - \mu_g|} \qquad (7)$$

$$w_{equalize} = \mu_\perp + w_g^{equalize} \qquad (8)$$

这就是 equalize 算法相关的所有公式了，乍一眼看上去有些懵有没有。我一开始看这些公式，也有些恐慌，但仔细阅读论文，尽管作者没有太详细的说明，但还是给出了线索，只是需要有一定数学基础才能体会到。大家不要被这些表面的东西吓倒，都是纸老虎而已。言归正传，我开始解释这些公式产生的原因。为了更好的表达我的意思，我先给出一个执行完 equalize 后产生的效果图像，通过结合这个图像来给而大家解释。如下图所示：



从这幅图中我们看到在执行完 equalize 后，重新调整的w_{father}和w_{mother}，也就是那两条紫色线段。我们发现它们和$w_{receptionist}$之间取得了很好的均衡效果。它们到$w_{receptionist}$距离相等了，就是图中两条橙色线段的距离。同时w_{father}到$\vec{g}_\perp$的距离和w_{mother}到$\vec{g}_\perp$的距离也相等了。就是线段$w_{father}c$和线段$w_{mother}c$的距离大小相等。照着上面给出的那些公式来计算，就必然是这个结果。那么我们证明一下为什么按照公式计算就会产生这些线段相等的关系。

首先我们看到公式 4，很显然它是求解所有需要进行 equalize 的向量的均值向量。都是类似w_{mother}和w_{father}这样有明显性别属性的词向量，我们这里举例只用了两个，实际上会有很多。所以我们要求个均值作为代表。这里μ就是这些词向量的均值向量。这里我还要着重声明一点，就是我们所有的嵌入词向量都要单位化。这样就保证了它们的模也就是欧氏距离大小为 1。然后公式 5 就是求出μ在轴$\vec{g}$上的投影μ_g，公式 6 就是求出μ在$\vec{g}_\perp$上的投影$\mu_\perp$。因为μ是所有样本词向量的均值，所以μ_g和$\mu_\perp$也都是对应的均值投影。我们再往下看公式

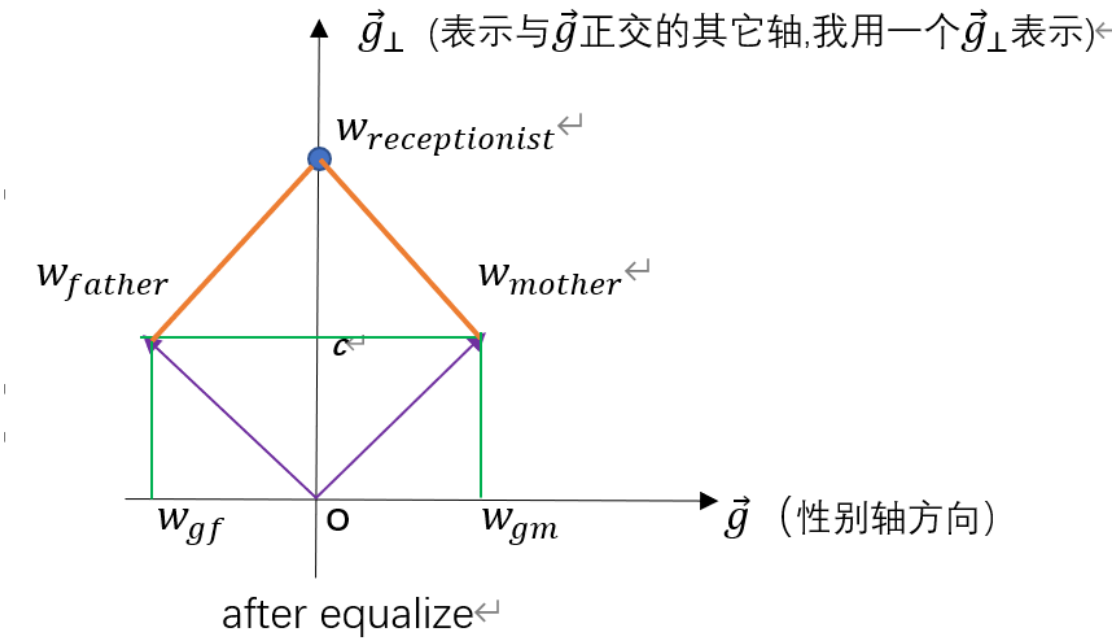
7，这是最核心的公式。难点全在这里。它这里是要求出一个新的在$\vec{g}$的投影向量。我们将这个公式分解一下：

$$w_g' = \frac{\vec{w}_g - \mu_g}{\left|\vec{w}_g - \mu_g\right|}$$

我们先看这部分，这个是什么呢，就是$\vec{w}_g - \mu_g$所形成的向量，然后再执行单位化得到新向量w_g'。$\vec{w}_g$是我们要进行 equalize 的那些向量在$\vec{g}$的投影，μ_g前面说了是这些向量的均值μ在$\vec{g}$上投影得到的均值投影向量，所以$\vec{w}_g$和μ_g是方向的，都是落在了$\vec{g}$轴上面。那我们知道这个表达式是对原始词向量进行中心化(减去均值)，然后再进行单位化得到的新的沿着$\vec{w}_g$方向的向量，即w_g'。这里我们其实应该或多或少明白了为什么要在各个轴上都使用均值向量，其一是使用均值向量，进行的这些运算，保证了所有词向量新计算出来的向量的模都是相等的，即所有w_g'的模都相等。其二进行中心化的作用是因为，仅执行单位化得到的向量，可能会产生相同的向量，造成无法使用类比操作。我稍微再仔细说明下，假设我们的性别轴是女性值趋向正方向，男性值则趋向负方向。如果有两个词向量，在性别轴上的投影一大一小，但都是正方向，那么直接执行单位化，就会得到两个相同的向量，这样它们就无法根据性别作类比了。两者之间也没有性别属性的差异了。而通过中心化，也就是减去均值，那么就会使它们两者的性别值在原点两侧，总而保证了性别的差异性。

再接看公式 7，得到的w_g'再乘上一个系数$\sqrt{1-|\mu_\perp|^2}$,是不是又被根号整迷糊了，其实这个表达式很明显了啊，就是μ_g的模啊，因为我前面说过，所有嵌入词向量都要单位化啊，所以大小是 1 啊。根据勾股定理$\sqrt{1-|\mu_\perp|^2}$不就是$\left|\mu_g\right|$了嘛。这样是不是就感觉明白了很多了啊。w_g'是单位化后的向量，再乘上对应

的模的大小，是不是就是w_{father}和w_{mother}等向量在执行中心化后在$\vec{g}$上的新的投影向量，就是公式其给出的$w_g^{equalize}$向量，并且保证了任意词向量按照此方式计算得到的$w_g^{equalize}$大小都相等，因为都是基于均值做的调整。那公式 8 中使用公式 7 得到的$w_g^{equalize}$，再加上$\mu_\perp$，得到执行 equalize 后的新向量。对每一个原始词向量通过实施 equalize 得到的$w_g^{equalize}$，都保证了其方向不同，但是对应的模都相等。再啰嗦一下，就是因为使用了均值和单位化才会产生所有生成的$w_g^{equalize}$都相等的效果。我把上面的执行完 equalize 后的那张图片再贴下来，方便读者观看，请看下图：



那我们根据前面的公式 4、5、6、7、8，就能知道图中紫色线段就是我们均衡化后产生的新的词向量。他们的模是相等的。图中线段 Ow_{gf}和 Ow_{gm}就是$w_g^{equalize}$，所以线段 Ow_{gf}和 Ow_{gm}的长度相等。线段$w_{father}w_{gf}$和$w_{mother}w_{gm}$就是$\mu_\perp$，所以这两条竖直的绿色线段的长度也相同。那么三角形 O$w_{father}w_{gf}$和 O$w_{mother}w_{gm}$是全等的。所以它们的角度也都相同，那么可以推导出

<cOw_{father}和<cOw_{mother}相同。那说明 Oc 这条线是等腰三角形的中位线，可以推出 oc⊥线段$w_{father}w_{mother}$，并且线段 cw_{father}等于 cw_{mother}。即是调整后的w_{father}到$\vec{g}_{\perp}$的距离和w_{mother}到$\vec{g}_{\perp}$的距离也相等。那么也就可以推出图中两条橙色线段也相同。也就是调整后的w_{father}和w_{mother}到$w_{receptionist}$的距离也相同了。这也就证明了 equalize 操作确实起到了均衡的作用。

下面我们再给出实现 equalize 的 python 代码。这个代码主要实现对任意两个向量，进行均衡调整后，返回修正后的向量。这里是为了简化说明，仅仅只调整了两个向量，可根据具体情况任意调整。

```
def equalize(w1, w2, bias_axis):
    """
    通过均衡方法来消除性别偏差。
    参数：
        w1,w2 -- 要消除偏差的词向量，比如（"mother", "father"）
        bias_axis -- 维度为(ndims,)，对应于偏置轴（如性别）
    返回：
        w1_update -- 第一个词均衡后的词向量
        w2_update -- 第二个词均衡后的词向量
    """
    # 计算 w1 与 w2 的均值
    mu = (w1 + w2) / 2.0
    # 计算 mu 在偏置轴与正交轴上的投影
```

```
  mu_g = np.divide(np.dot(mu, bias_axis),
np.square(np.linalg.norm(bias_axis))) * bias_axis
 mu_orth = mu - mu_g
#使用公式 2 计算 w1_g 与 w2_g
w1_g = np.divide(np.dot(w1, bias_axis),
            np.square(np.linalg.norm(bias_axis))) * bias_axis
w2_g = np.divide(np.dot(w2, bias_axis),
            np.square(np.linalg.norm(bias_axis))) * bias_axis

#根据公式 7 调整 w1_g 与 w2_g 的偏置部分
equalize_w1_g = np.sqrt(np.abs(1 - np.sum(mu_orth * mu_orth))) *
                    (w1_g - mu_g) / np.linalg.norm(w1_g - mu_g)
equalize w2_g = np.sqrt(np.abs(1 - np.sum(mu_orth * mu_orth))) *
                    (w2_g - mu_g) / np.linalg.norm(w2_g - mu_g)

  #使用公式 8，计算各方向投影之和，从而消除偏差
  w1_update = equalize_w1_g + mu_orth
  w2_update = equalize_w1_g + mu_orth

  return w1_update, w2_update
```