# 一个由单词向量构建的简单拼写检查器

如果你听说过词向量，你可能会知道它们可以对词进行代数运算。例如，

给出了一个最接近德国的答案。

鲜为人知的是，你可以使用相同的词向量代数来修正拼写错误。这篇文章描述了如何建立一个查找表，其中包含最常见的拼写错误及其使用预先训练的更正。

伴随着这篇文章的是一段代码。您还可以找到要下载的查找表版本。

词向量

计算机只处理数字，因此要让计算机分析文本数据，例如，要查找主题、翻译、总结等，必须首先将数据转换为数字。“词向量”只是表示一个词的一组数字：计算机对该词的内部表示。

如果我们训练计算机从一个句子中预测丢失的单词，给它上百万个可供学习的例子，并且我们允许计算机通过更改分配给每个单词的数字来改进其预测，我们会发现同义词最终会被分配给彼此接近的数字。

有很多博客文章和教程可以解释矢量训练过程背后的机制。我下面的目的是理解为什么这些词最后会出现在它们所处的位置，以及为什么同义词会出现在一起。

考虑预测下列句子中缺少的单词的问题：

“我拿起那本书开始写作。”

我们能说些什么呢？它必须是一个名词；它可能是一个东西（尽管它可能是一个隐喻性的东西）；但它最有可能是一个可以写的东西，或者是一个可以写的东西。

假设计算机对单词的表示由2个数字 - 组成，即单词向量有2维。因此，我们可以将每个矢量视为一个点的坐标世界地图 - 每个单词都位于这个世界上的一个不同点（单词？）地图。在训练过程开始时，单词分配是随机的，单词分布在世界各地。

接下来，假设计算机通过向地图投掷一个假设的飞镖来做出预测。它的预言将是最接近飞镖降落点的词。

我们假设的玩电脑的飞镖有一个相当不稳定的目标。面对上面的句子，它认为“笔”是最有可能的词。实际上，它认为是一组坐标 - 这个词的向量 - ，恰好是位于巴巴多斯岛上空的一个点。所以，计算机的目标是巴巴多斯，但飞镖降落在圣卢西亚的东边。考虑到单词的初始分配是随机的，它最终会预测一个与写作无关的单词。

计算机如何改进它的预测？很明显，如果所有的书写工具词都聚集在西印度群岛，那么当我们瞄准“笔”时，我们至少可以点击“铅笔”，这可能是至少在一些情况下“笔”是合适的正确答案。

到目前为止，还不错。但下面这个（有些做作的）例子呢

“我马上给他写封信。”

“写”和“笔”看起来都是不错的选择，但这会给我们带来麻烦。如果书写工具名词和书写动词都聚集在一起，我们如何避免在本例中预测“pencil”或“biro”；或者在第一个示例中预测“write”等动词？

答案是给计算机一个更高维的词向量。尽管展开了一个已经很粗略的类比，但想象一下，计算机现在抛出了两个飞镖：一个飞镖在地图上，另一个飞镖在一个从海底一直延伸到平流层的高度图上。这给了我们空间，让“与写作有关的事情”继续位于西印度群岛上空。类似于名词的词可能出现在海平面附近，而动词则出现在天空的某个地方；笔位于两者之间，可能更靠近地面，因为它最常用作名词。

最广泛使用的词向量有300维。也就是说，我们为每个单词分配300个数字。这提供了一个巨大的空间来存储关于单词的各种不同方面的信息 - 是否是名词、动词、形容词等；动词的时态；名词是否复数；与单词含义有关的各种不同方面；以及，正如我们将在下面看到的，它的拼写是否正确。

在开始拼写错误之前，请再次注意，计算机除了发现单词的上下文之外，对单词一无所知。尤其是，它不知道使用了哪些字母。它无法知道“写”和“写”有一个共同的根。这两个词在向量空间的相似部分结束，因为它们所在的上下文重叠。此外，两个词向量之间的一些差异将显示一种特定的模式，因为“write”是动词的不定式，“writing”是现在分词；这种差异在所有动词中都是相似的，因为使用不定式和现在分词的上下文也重叠。

词向量与拼写错误

其中一个版本的手套向量是在8400亿字的数据上训练的，这些数据是从网络上爬出来的，结果产生了220万字的向量。

太多的话了。比牛津英语词典里的要多得多。这种差异很大一部分是由专有名词，如人名、地名等，这些专有名词不会出现在词典中。然后是数字、复数、不同的动词结尾、标点符号、大写单词等等。

但是，对于包含拼写错误的单词，也有很多向量。这给了我一个想法来回答论坛上的一个问题，这个问题要求找出并纠正常见的拼写错误。假设一个拼写错误的单词与它的正确对应词在同一上下文中使用，那么也许我们可以通过查看向量空间中拼写错误的单词的邻域来找到正确的拼写？

结果这不管用。远远没有。但现实更有趣。

喷流微星的平行宇宙

下表显示了向量空间中最接近“可靠”的10个单词：

现在，想想拼写错误的“releable”的近邻：

我们还有7种拼写可靠错误的方法，其次是拼写错误的可靠同义词。正确的拼写“可靠”是看不到的。

这是一个奇怪的结果。当然，releable是可靠的同义词？它们会出现在非常相似的语境中，可能出现在完全相同的句子中。那么，为什么这些拼写错误不会出现在向量空间的同一个地方呢？

我再回到下面的问题。目前，我们注意到，向量空间中似乎有一个非常明显的部分与拼写错误有关。这就引出了一个问题：有没有一种方法可以通过将拼写错误的单词向量转换成在向量空间中的位置来修复拼写错误？

答案是你可以，而且很简单。它的工作方式与上面提到的概念代数完全相同。而不是：

你有点像：

为了推广这种方法（减少对可靠性的依赖……），我们可以通过计算一组拼写正确和错误的单词之间的平均差来构建一个拼写转换向量。然后，我们可以通过从拼写错误的单词向量中减去这个拼写转换向量，并找到最接近最终结果的单词来修复拼写错误。

拼写转换向量

下面的图表绘制了一组常见拼写错误的单词在300维范围内正确拼写和错误拼写单词向量之间的差异值。（注意，每个维度之间没有顺序关系。我画了一个折线图，使图案更明显）。

似乎有一个特征性的模式，有两个向下的尖峰朝向地块的左侧。

看起来更极端的拼写错误会产生更长的尖峰，换句话说，非常常见的拼写错误会产生更短的尖峰。例如，recieve - receive和calender - calendar都有很短的峰值。然而，我们无法在releable的近邻中找到可靠的，receive是receive的第二近邻（仅次于receive）。

因此，在单词向量空间中考虑拼写错误的方向可能比考虑特定区域要好。

从图表中可以清楚地看到，不正确和拼写正确的词向量在词对之间的差别很大。因此，要抓住从拼写错误到拼写正确转变的本质，我们需要对一大组词对取平均值。

不幸的是，我没有这样的数据集。作为起点，我使用了，它包含了100多个示例。留出这一小部分的15%用于测试，我构建了一个初始拼写转换向量，它等于错误和正确单词向量之间的平均差。

把这个应用到我们的17个单词的测试集中，我们成功地修复了15个（88%）。它唯一的两个对手是法老和飞轮海。与calendar和receive一样，它们很可能拼写错误，以至于正确和错误的词向量之间几乎没有区别。

构建更好的拼写转换向量

88%是好的，但我们可以做得更好。我们可以通过反向应用我们的初始转换来发现更多的拼写错误，从而构建一组更大的示例来进行“训练”。例如，应用于，因为我们得到：

这看起来是一个相当全面的不同拼写方式的列表，因为，但我们也得到了一个不幸的交易。不幸的是，这可能是因为这两个词通常位于 - 之间，但这也可能与“不幸”出现在转换向量所用的词集中有关。

申请后我们得到：

关于这个名单有几点需要注意。

* 所有的例子都是拼写错误，你不可能真的拼写错。因此，它们显然受到了QWERTY键盘布局的影响（s和f位于d的任意一侧，b位于n的旁边），但列表中缺少amd。原因是AMD更常见的是指计算机芯片公司，因此，它被认为是一个正确的拼写接近紧贴雅典，奔腾，英特尔和CPU。
* 它仍然包含和。我们可以通过在拼写错误方向 - 进一步移动来解决这个问题，方法是添加一个拼写转换向量的倍数。

与之相反的是，没有一个候选人是因为手指打错了。这指出了这种拼写纠正方法的一个局限性 - 我们只能处理出现在单词向量词汇表中的情况。

拼写正确率得分

稍加注意，我们现在有了一种为给定的正确拼写生成常见拼写错误的方法。现在我们只需要一长串拼写正确的单词。

我也没有这个，但是单词向量是按照它们出现的频率降序存储的。所以，我们可以只取10000个最常出现的单词，并假设其中绝大多数单词拼写正确。

这也许可以，但我们可以做得更好一点。我们可以使用初始拼写转换向量来判断单词拼写错误的可能性。记住，转换向量指向拼写错误的方向。如果我们将一个词向量投影到转换向量上，我们可以测量它指向那个方向的距离。也就是说，我们取两个向量的点积，称之为“拼写得分”。

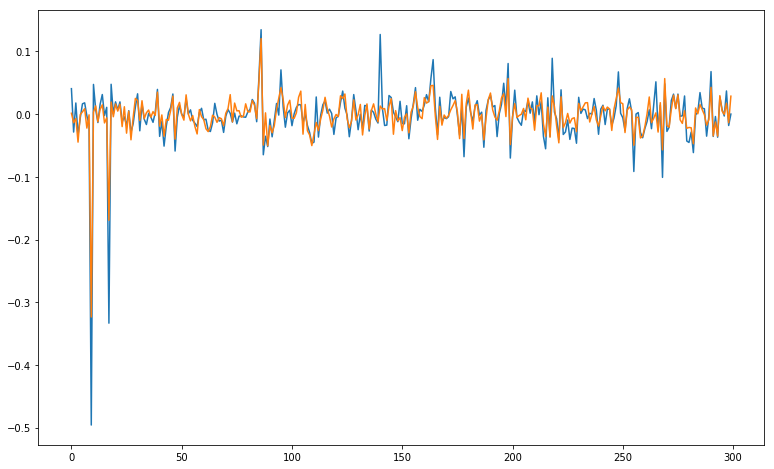
下面是一些根据这个标准的得分最高和最低的单词的例子。（为了夸大分数，以下是基于未规范化的词向量和我们的转换向量之间的点积）。

我们现在的做法是：

* 从50000个最常见的单词中选出得分最高的单词（加上一些其他过滤器）作为我们的正确拼写列表。
* 应用反向转换并查找最近的邻居，通过排除明显不同的单词来筛选这些候选词，以构建一组大量正确 - 不正确拼写对的示例。
* 计算一个修正的变换向量作为这些正确 - 不正确对的字向量的平均差。

初始向量和拼写向量的比较

新向量本质上是原始向量的一个更极端的版本。两个向量之间的相关性都在90%以上，但新向量的方差要高得多。下图以蓝色绘制新矢量，以橙色绘制原始矢量：



应用到我们最初的17个例子的测试集，现在它们都很好了 - 甚至是“法老”！

应用到14000对左右，我们建立了转换向量，通过一些额外的技巧和调整，我们最终得到了超过90%的正确率。通过一些错误的例子，很明显，这些错误中的许多代表了训练集中的错误，这是我们构建数据集的自动化方式的结果。

为什么拼写错误会聚集在向量空间？

我们现在可以更好地解释为什么所有的拼写错误都被分流到一个平行的空间？最明显的线索是上面列出的拼写最好和最差的单词。拼写最好的词是那些你可能在严肃的新闻文章中找到的词；拼写最差的词主要是非正式语言的例子。

您可以在web上找到两组截然不同的文本内容：

1. 出版前经过仔细校对、编辑和/或拼写检查的内容。
2. 来自论坛、电子邮件、twitter等的未筛选用户生成的内容。

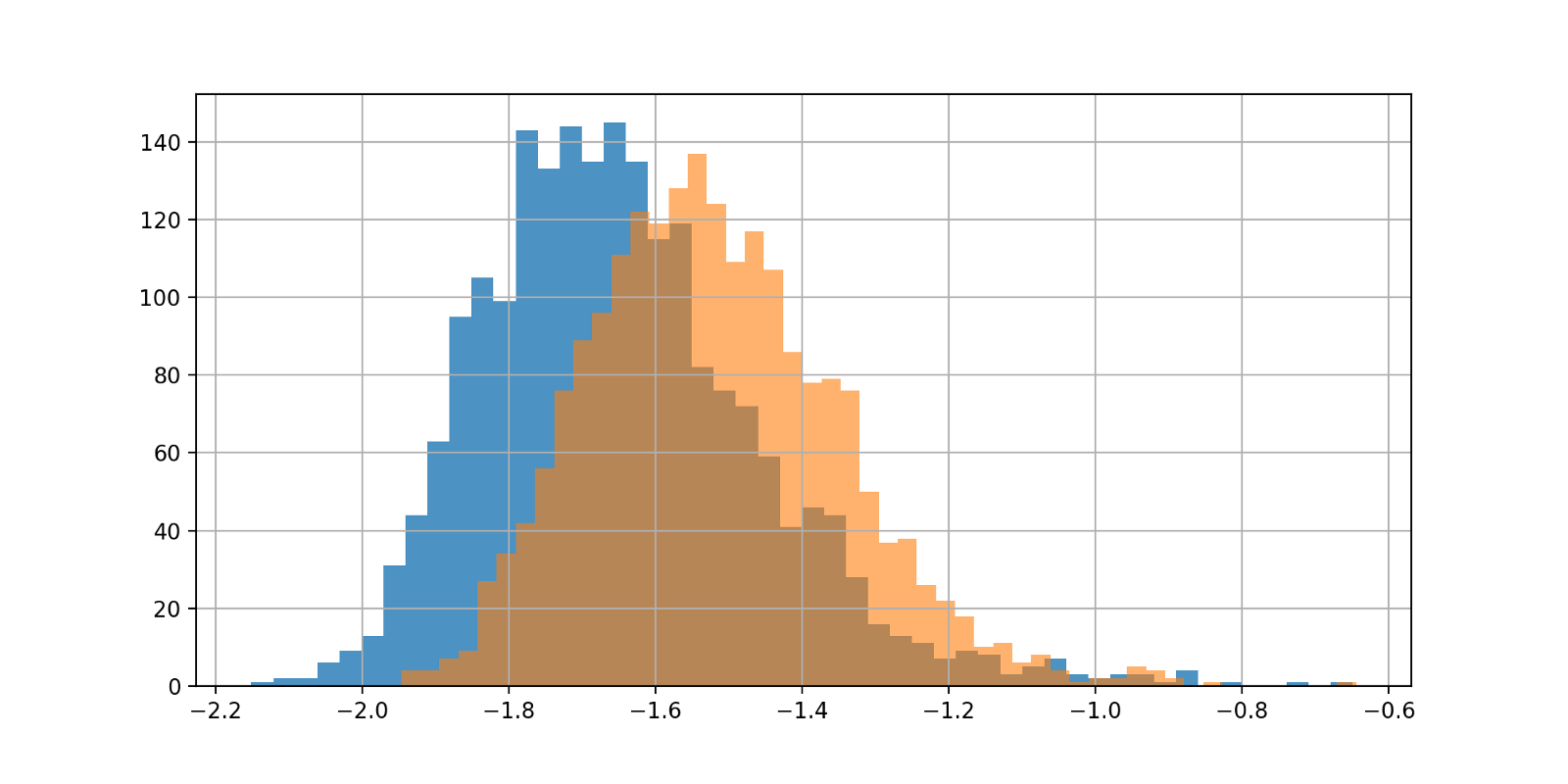
第二组文本更可能包含拼写错误。看到像plz或ROFL这样的词将是一个非常有力的指标，你正在看用户生成的内容。这样，你就更有可能看到拼写错误和拼写错误。同样，当我们开始阅读百分比、经济、政府等信息时，有理由猜测文本来自新闻来源，因此拼写正确。

另一个支持性的证据是，如果我们试图使用在维基百科垃圾场训练过的手套词向量和Gigaword新闻语料库数据，这些数据将包含更少的用户生成内容，那么这种方法根本不起作用。拼写错误仍然有一个区域/方向，但它的定义要差得多，拼写错误的例子更少。

如果上面列出的论点是正确的，那么我们的“拼写分数”最好被看作是一个“正式”分数，表明一个单词来自已发布源而不是用户的可能性有多大。

为了测试这个想法，看看这个新指标是否有实际用途，我构建了一组数据，其中包括来自的BBC商业新闻，并将其与来自的一组样本进行了比较。每一个句子或评论都会根据文本中单词的平均“正式程度”得分。

结果并不理想，但该方法确实能够分离出至少一些文本，如下图所示。英国广播公司的新闻报道得分形成蓝色分布，IMDB评论为橙色。



最后的想法

除了是单词向量的一个有趣特性外，拼写错误/非正式语言的并行世界允许我们创建一个简单的查找表来纠正常见的拼写错误。这应该有许多应用程序，但作为清理原始文本数据的非常简单的第一步，它应该特别有用。

然而，值得承认的是它的局限性：

* 即使它工作得很好，它也只能纠正出现在预先训练过的词向量词汇表中的拼写错误。
* 它不能纠正单词中部分出现的拼写错误。它只能在一个单词标记上操作。
* 同样，如果两个单词缺少一个空格，它也无法更正错误。
* 虽然单词向量是建立在上下文的基础上的，但是我们试图独立地纠正单词。因此，该方法无法解决更困难的情况，即拼写错误或拼写错误会创建不同（但拼写正确）的单词，如我们在上面的amd示例中所述。
* 如上所述，我们倾向于没有较长单词的排版示例，可能是因为它们发生的频率不够。因此，我们可以纠正错误，但不能纠正错误。

为了满足这些限制，有很多方法可以在这里概述的基本方法的基础上进行构建。例如：

* 使用字符rnn从我们在上面的过程中发现的常见拼写错误的例子中学习模式，然后可以用于词汇表外的单词。这里有一个有趣的想法，你可能想按单词长度来划分词汇表，然后训练简短的单词示例来学习基于QWERTY的打字模式。
* 使用语言模型构建新的词向量来扩展词汇。
* 使用更传统的拼写检查技术，查看是否可以将拼写/形式分数较差的单词拆分为两个或多个拼写正确的单词。即使在这里，向量空间这个词的结构也有帮助：例如，download的最近邻居是and，yourand，toand，ofand。

最后一句话就是感谢雷切尔·托马斯和杰里米·霍华德的创作。讲座是一个极好的资源，现在它已经有了一个不断发展，热情和支持的社区。如果，像我一样，你刚开始学习机器学习，但由于某些原因还没有接触到他们的课程，我不能高度推荐。

|尽管到目前为止似乎还没有人采取措施将此功能转变为实用的词典/拼写检查器，但至少机器学习/NLP社区中的一些人知道词向量空间内的关系。例如，Jake Mannix提到了它。

#### More where this came from

This story is published in [Noteworthy](http://blog.usejournal.com), where thousands come every day to learn about the people & ideas shaping the products we love.

Follow our publication to see more product & design stories featured by the [Journal](https://usejournal.com/?/utm_source=usejournal.com&utm_medium=blog&utm_campaign=guest_post) team.