### 基于LSTM的图像自然语言描述

窦珊[[1]](#footnote-1) 马平烁[[2]](#footnote-2) 汪洁[[3]](#footnote-3)

清华大学自动化系，北京 100084

摘要：自动描述图像内容(Image Caption)是计算机视觉和自然语言处理的一个交叉问题，本文使用CNN模型抽象图像，并用LSTM模型生成流畅的自然语言。在此基础上，本文将Attention机制加入LSTM模型，作为第二种模型进行对比。最终我们在9000张图片的中文标注数据集上进行训练，证实了模型的可行性和优越性。

1 引言

全排列问题是组合数学中的经典问题1。目前已经有很多算法解决了这个问题，包括字典序法、递增进位制数法、递减进位制数法和邻位对换法。在这四种算法中，字典序法最为简单和灵活，其思想是对给出的一个元素的序列，假设其满足，那么算法会依据字典序产生{}的全部排列。比如说，对序列，算法输出的序列是：

字典序虽然简单，但其算法中有很多不必要的计算，使得其运行缓慢，存在很大的改进空间2。本文基于此，通过降低改进字典序算法复杂度中的常系数，提高了算法的运行速度。

2 模型和方法

2.1

字典序法生成全排列的算法如下，设是的全排列，下一个排列的生成算法如下。

* [寻找 *j.*] 从排列的右端开始，找出第一个比右边数字小的数字的序号（从左端开始计算），即
* [增大 *pj.*] 在的右边的数字中，找出所有比大的数字中最小的数字，即（右边的数字从右至左是递增的，因此是所有大于的数字中序号最大者）
* 交换
* [逆转*.*]将倒转得到排列 ，这就是排列的下一个排列。

要生成的全排列，那么算法的初始序列为，终止序列为。

2.2

对于数列和，假设和前个数均相同，若，则，若，则。特别的，若，则。

要证明字典序的正确性，需要证明每次生成的下一个排序恰好比当前排列大的最小的一个序列即可。

* 第一步找出的序列是一个递减序列。
* 第二步和第三步得到。因为，这就保证了新的序列大于原来的序列。
* 第四步通过将后边的序列逆转（原序列是递增序列），从而得到比当前序列大的最小的序列。

2.3

算法总共要进行次，算法复杂度取决于每次平均进行多少次交换和比较。不妨对每个步骤进行分析：

第一步中，寻找*j*的过程，需要比较的次数是的概率是,其中表示,也即是.所以均值为：

第二步中，增大 *pj.*的过程中，总共发生*k*次比较的概率是：

所以，均值是;

第四步中，平均交换数目是

所以整体来讲，复杂度为,其中是常数。我们接下来的工作就是优化这个常系数。

3 结果

3.1 算法改进思路

在字典序中，每次查找下一个序列会用到次交换。事实上，在个全排列中，存在个排列满足,这种情况下只需要用到1次交换。同理，的情况只有，所以我们对此进行优化，对和的情况分别进行特殊处理。经过这个优化，在第二步中的运算量降低了.

3.2 改进算法流程

改进的字典序全排列生成算法如下，假设，同样设是的一个排列，下一个排列的生成算法为

* [第一种简单情况]判断和的大小，如果,则交换和，即为当前条件下的下一个排列；如果，则进入下一步。
* [第二种简单情况]判断和的大小

如果且，则原来的换到，原来的换到，原来的换到，即为当前条件下的下一个排列；

如果且，则原来的换到，原来的换到，原来的换到，即为当前条件下的下一个排列；如果，则进入下一步。

* [寻找 *j.*] 从开始向左扫描，直到找到比右边数字小的数字的序号。如果，则不存在下一个排列。
* [第三种简单情况]判断和的大小,如果，则原来的换到，原来的换到，原来的换到，然后进入最后一步。而如果，则进入下一步。
* 从开始向左扫描，直到找到比大的数字，序号为；交换和，交换和。
* 最后将倒转，即得到下一个排列。

4 讨论

为了检验本文改进算法的效率和正确性，我们将本文方法和传统方法进行对比。实验部分在内存为8GB的MacOS系统上进行，CPU是Inter Core i5，编程语言是C++。为了保证时间记录的准确性，表4.1省略了算法中输出字典序的时间，仅仅记录了计算时间。

表4.1 改进的字典序法与传统方法的效率对比（单位:ms）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 递增进位 | 递减进位 | 邻位对换 | 传统字典序 | 改进字典序 |
| n=9 | 115 | 124 | 7.456 | 7.70 | 5.09 |
| n=10 | 1294 | 1470 | 76.2 | 61.6 | 41.9 |
| n=11 | 15847 | 18561 | 827 | 688 | 427 |

从图4.1中可以看出四种算法中递增、递减进位制数法由于使用了中介数，所以效率比其他两种方法低很多。传统的字典序法和邻位对换法的效率不相上下，而改进的字典序法相比传统的字典序法提升了大概40%的效率。

图4.1：改进的字典序法与传统方法的效率对比（单位:秒）

参考文献

[1] 卢开澄; 卢华明, *组合数学*. 清华大学出版社: 1991; Vol. 67.

[2] Donald, E. K., The art of computer programming. *Sorting and searching* **1999,** *3*, 426-458.

1. 窦珊：; [↑](#footnote-ref-1)
2. 马平烁, 2016211151, mps16@mails.tsinghua.edu.cn [↑](#footnote-ref-2)
3. 汪洁, 2016211087; [↑](#footnote-ref-3)