**摘要**

我们的目标是能够从深层的神经网络架构中建立一个生成模型，试图创造出既和谐又有旋律的音乐，并且可以作为人类作曲的音乐。以前在音乐创作方面的工作主要集中在创作一首单一的曲子。最近围绕**时间序列的概率密度的估计**的复调音乐建模工作取得了一些成功。特别是，已经有很多的工作，基于结合了受限玻尔兹曼机（RBM）的递归神经网络（RNN）和其他类似的性能量模型。然而，我们的方法是仅用深层神经网络来完成端到端的学习和生成。

**介绍**

音乐是终极语言。历史上许多令人惊叹的作曲家创作了既有创意又有刻意的作品。像巴赫这样的作曲家以制作具有大量**底层音乐结构**的作品而著称。那么，计算机也有可能学会创造这样的音乐结构吗？

灵感来自一篇博客，写到能够创作出似乎带有旋律并且相对和谐的**复调音乐（和弦乐？）**，我们决定处理同样的问题。我们试着回答两个主要问题：

1、有没有一种有意义的方法把音符作为矢量来表达？或者说，是否有一种方法来表达单词的意思就像**word2vec[[1]](#footnote-1)**对音乐的翻译一样？

2、我们可以建立有趣的生成神经网络架构，有效地表达和谐和旋律的概念吗？大部分的作品都有一段主旋律，它可以扩展，我们的神经网络也能做到吗？

**背景及相关工作**

Chen et al写的最早的一部关于深度学习产生音乐的论文，产生了一首只有一个旋律却没有和声的音乐。作者还省略了音符、休止符和所有和弦。他们提到的主要问题之一是音乐缺乏总体结构。

这给我们提供了两个主要的方向去提高它：

1、使用音乐的节奏、更复杂的结构、并利用所有类型的音符：包括点音符、更长的和弦、和休止符，来创造音乐。

2、创建一个学习长期结构的能力模型，并有能力建立一个旋律，并返回到整个作品。

Liu et al处理同样的问题，但无法克服任何挑战。他们声明，他们的音乐表现没有正确地区分（对应的）旋律和作品其他部分的旋律，此外也不能解决大多数经典作品中的全部复杂性。他们引用了两篇论文，试图解决上述的问题。

eck et al使用两种不同的LSTM（Long-Short Term Memory，是一种时间递归神经网络）网络：一个学习和弦结构和局部音符结构；另一个学习长期依赖，以此来尝试学习旋律并在这个过程中保留它。这使作者可以产生的音乐，却不需要分离原和弦的旋律。然而，这种体系结构训练了一组和弦，却不能创造出更加多样化（新的）的音符组合。

另一方面，Boulanger Lewandowski等人，试图解决音乐中学习复杂复调结构的挑战。他们使用一个“递归的时间限制Boltzmann机（RTRBM）”，以此建立一个不受现代复调音乐。RTRBM结构的使用使他们可以表现一个复杂的分配对**每个时间步长**，而不是大多数字符语言模型中一个单一的符号。这使他们能够处理生成音乐中的复调问题。

在我们的项目中，我们将主要解决学习复杂的结构和节奏的问题，比较我们与Boulanger Lewandowski等人的结果。

**数据**

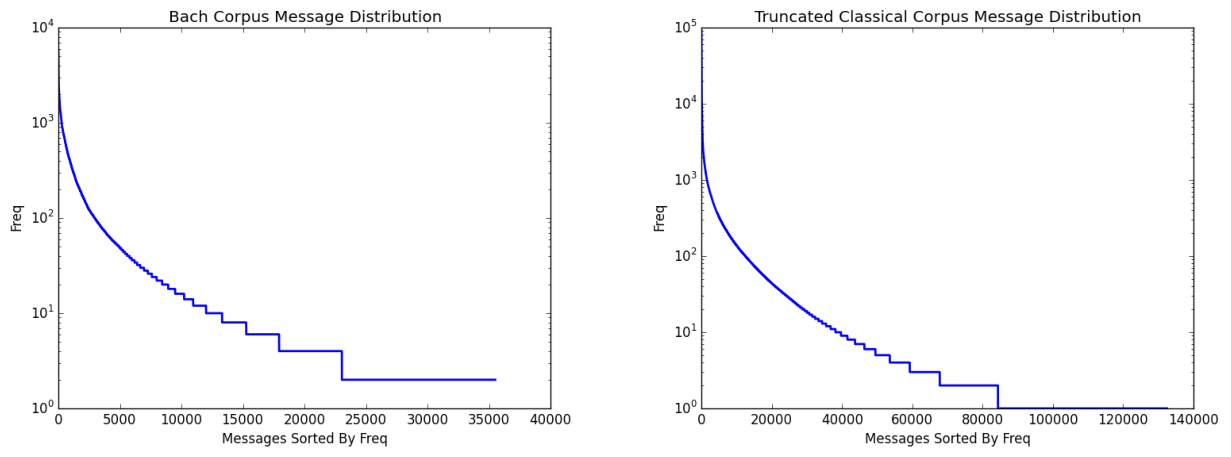
音乐生成训练模型的主要挑战之一是选择正确的数据表示。我们选择了两种主要类型：最小预处理的**MIDI**[[2]](#footnote-2)文件和MIDI文件的“钢琴滚动”表示。

**midi数据**

MIDI文件被构造成一系列并发磁道，每个磁道包含一列元消息和其他信息。我们提取音符及其持续时间相关的消息，并将整个消息编码为唯一的标记。例如，“note-on-60-0”后接“note-off-60-480”会转化成两个独立的信息或标记。同时，这两个消息将指示MIDI播放器播放“中音C”480次，这意味着大多数midi时间尺度会转换为四分音符。我们使音轨变得平滑，以便于一个被分开的单独音轨的标记将被连接为“端到端”。

我们开始从musedata（对于巴赫语料库的数据可以直接从HTTP的musedata网站：/ / musedata.org /.下载）中下载整个巴赫文集，因为巴赫是那个网站中相对多产的作曲家。在我们的巴赫语料库中，总共有417件共被标记1663576个编码标记。我们还确保每片节拍正常化。然而，我们没有把每一块都换成同一个键，这一点被证明可以提高性能。

图一：信息或标记被分为两部分：巴赫语料库（左）和经典音乐语料库的截断版本（右）的：



此外，我们还从其他不同古典作曲家的其他**联机存储库**[[3]](#footnote-3)中提取了额外的MIDI文件。这将我们的语料从大约100万个标记扩展到大约2500万个标记。由于我们模型的内存限制，我们主要对包含2000件的数据集的截短版本进行操作。

|  |  |  |
| --- | --- | --- |
| 数据集 | 词 | 独一无二的标记 |
| 巴赫 | 1663576 | 35509 |
| 完全的古典 | 24654390 | 175467 |
| 截断的古典（部分） | 11413884 | 132437 |

我们还比较了巴赫唯一的MIDI语料库和整个经典MIDI语料库的标记分布，如图1所示。我们发现在这两个数据集中有许多消息的频率非常低。事实上，对于这两个数据集，超过三分之二的独有标记发生少于10次。

然而，更重要的是，直接编码MIDI信息的缺点是，它不能有效地保留多个音符同时被多条音轨同时播放的概念。因为我们连接轨道的端到端的。我们认为，让我们的模型在相同时间段去学习不同轨道的相同位置上的各种音符是很困难的。

3.2钢琴手卷数据

为了解决上面列出的缺点，我们转向用不同的数据表示。我们没有将标记按磁道分割，而是将每个MIDI文件表示为一系列时间步骤，其中每一步都是一个正在播放的音符ID列表。

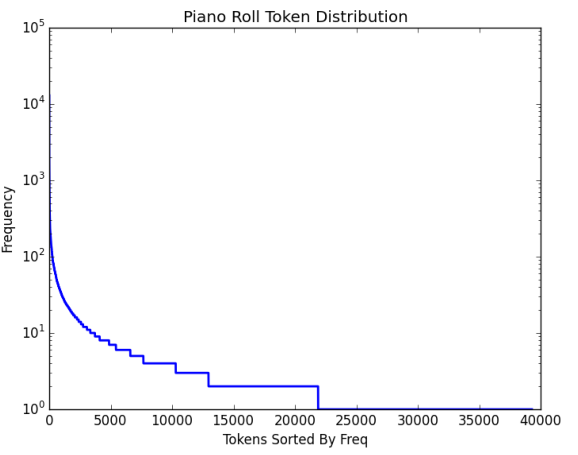


图2：“所有音乐” 钢琴滚动数据 集中所有标记的频率分布。

我们在**musedata**[[4]](#footnote-4)中检索到的所有作品用钢琴卷表示。数据集是由每个MIDI文件在第八音符间隔采样处创建的；碎片也调换到C大调和小调。所提供的训练集共有524件，共245202个时间步骤。通过把各个音符IDs连在一起，让我们编译的每个时间步长形成一个标记（如C大调和弦将表示为“60-64-67”）。此外，当我们关注唯一标记的数量时，如果在任何特定的时间步长超过4，我们就随机选择3个音符。

4.方法

我们用一个2层的“长短期记忆网络（LSTM）”“递归神经网络（RNN）”架构，产生字符级模型来预测序列中的下一个音符。

在我们的MIDI数据实验中，我们将每个MIDI信息视为单独的记号，而在我们的**钢琴卷（piano roll）**实验中，我们将通过所有时间步长得到的音符的所有独特组合作为一个单独的标记来处理。

我们创建一个嵌入矩阵，将每个标记映射成一个学习向量表示。每组的标记序列被连接到一个嵌入向量列表中，这些向量构成进入到LSTM的时间序列输入。

LSTM的输出馈入softmax层的所有标记。损失 对应于 我们对每个预测的时间步长和实际音符演奏的时间步长的交叉熵误差。

我们的架构允许用户设置各种参数如层数、隐层单元的大小、序列长度、批量大小、和学习速率。我们修剪我们的梯度，以防止我们的梯度爆炸。当我们看到训练误差减小的速度减慢时，我们也会降低学习速率。

我们通过在训练模型中输入一个短的种子序列来生成音乐。我们产生新的标记从我们的softmax输出分布和饲料新标记回到我们的模型。我们使用两种不同的抽样方案相结合：一个选择最大预测概率的标记，另一个从整个SoftMax分布中选择一个标记。我们在**AWS [[5]](#footnote-5)**g2.2xlarge中进行实例实验。我们深入学习的实现是在TensorFlow中完成的。

5.实验

5.1 基线

对于MIDI基线，我们有未经训练的模型生成序列。如图3所示，我们的模型无法学习MIDI消息的“开关”结构，从而导致许多休止。对于我们的钢琴卷的基线，我们从我们的钢琴卷表示中抽取随机的和弦，由它们在我们的语料库中出现的频率加权。



图3：（上面）：从未经训练的模型生成基线MIDI文件。（下面）：钢琴卷表示的标记加权样本。

我们看到，钢琴卷的音乐是很不和谐的，尽管每个和弦听起来合理，但从和弦到和弦是没有局部结构的。

5.2巴赫MIDI实验

我们首先在“巴赫”MIDI数据集上训练我们的模型。我们训练了大约50个**epochs（时代？）**，大约花了4个小时在GPU上训练。

隐藏状态128

标记嵌入大小128

批量大小50

序列长度为50



图4：只从巴赫数据库中产生的音乐

5.3古典的MIDI实验

由于时间限制，我们使用了与巴赫MIDI实验相同的架构，即“截断经典”数据集。15个epochs在GPU上花了22个小时。此外，由于AWS的g2.2xlarge设备内存的限制，我们被迫减少批量大小和序列长度。

隐藏状态128

标记嵌入大小128

批量大小25

序列长度为25



图5：由“截断经典”数据集生成的音乐

5.4讨论

有趣的是，我们发现由“巴赫唯一”数据训练的模型所产生的序列比那些在不同古典作品的抓取袋上训练的序列更美观。我们认为，相对于巴赫模型，经典MIDI模型的字符集的相对大小严重阻碍了它有效学习的能力。

我们也对字符模型使用**t-SNE[[6]](#footnote-6)，**以可视化我们的嵌入向量作为成功的标准。我们可以在图6中看到结果。o表示MIDI打开消息，x代表MIDI关闭消息。这些数字代表MIDI音符ID（较低的数字表示较低的频率），也由蓝色编码为红色（分别为低到高）。

由于我们的模型中有这么多标记，所以我们过滤我们的可视化，只显示演奏60次的音符。

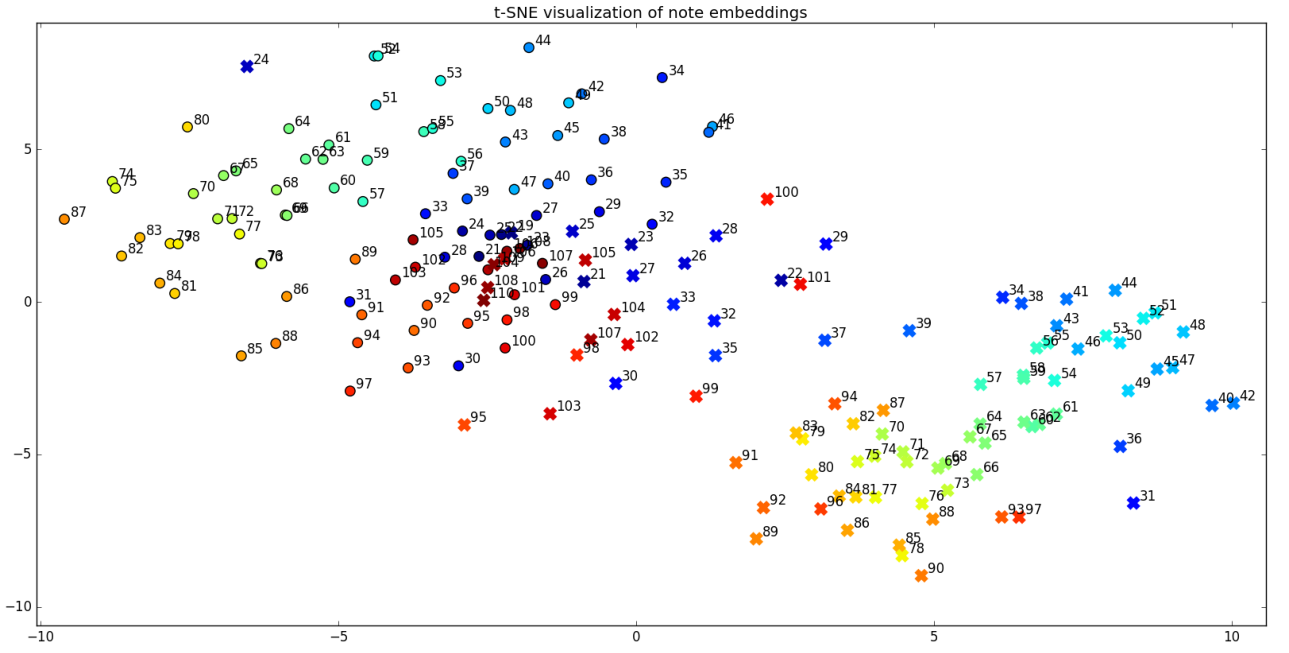


图6：从经典MIDI实验看嵌入载体T-SNE可视化

注意，有明显的集群在断断续续的中频音符之间（这是最常播放的音符），而罕见的低和高的音符都聚集在模糊的云的中心。

此外，该模型似乎学习将相似的音高组合在一起，并从低音高到高音高有某种线性的进展。在音符和非音符都有一个普遍的模式：较低的音在右上角，较高的音在左下角。

5.5 钢琴卷实验

我们用相同的参数来运行这个实验，并称他为“巴赫MIDI实验”。我们跑了800epochs，历时7小时在AWS g2.2xlarge实例。我们也运行相同的配置，在100个时代的截断数据集上，CPU花费了7个小时。



图7↑：由muse钢琴卷数据生成的音乐。前4行来自“muse\_all”数据集，后两行来自“muse\_截断”数据集

我们为新模型再次使用T-SNE去可视化我们的嵌入载体，如图8所示。由于我们有成千上万种不同的音符组合，我们首先来看编码单个音符被播放的记号。

在这里，我们看到了类似的结果，该模型能够清楚地区分在较低和较高的音阶。然而我们也看到，最低和最高的音阶也组合在一起，或者轻微的和其他音符相分开。

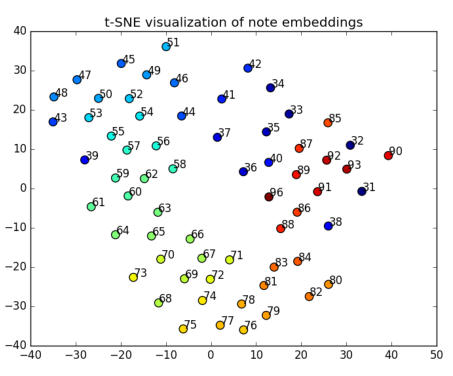


图8↑：从钢琴卷实验得到的单个音符嵌入向量的t-SNE可视化

6 评估

评价我们模型质量的主要挑战之一是融入音乐美学的概念。也就是说，我们的模型最终产生的音乐有多好？因此，我们设计了一个盲人实验，我们让26名志愿者对3个生成的音乐样本发表意见。

* 我们要求他们背对背听3个样品。
* 我们要求他们按比例从1到10打分。

1的评级是“完全随机噪音”。

5的评级是“音乐上好像真实的”。

10的评分是“由一个初学作曲的人作曲”。

样本的识别如下：

样品1 ：10秒的剪辑的“巴赫MIDI”模型

样品2 ：16秒的从库[[7]](#footnote-7)中导出的剪辑的“7 rnn-nade序列”

样本3 ：11秒的“钢琴卷”模型训练的“muse\_all”数据集

我们选择了对我们的序列与库中RNN神经自回归分布估计（rnn-nade）序列进行比较，因为它取得了类似的结果，其他常用的技术如rnn-rbm和rtrbm和robust作为一个分布估计。

我们的研究结果表明，事实上我们制作的音乐，至少相当于审美素质的rnn-nade序列模型做的。事实上，在图9中，我们看到，26名志愿者只有3说他们更喜欢从rnn-nade出来的声音。（另外，这3个人说他们喜欢那个序列和我们制造的序列是差不多的。）也就是说，在图9中的直方图，我们看到的3个样品的平均收视率分别为7±1.87、5.3±1.7和6.2±2.4，这表明我们的样本规模过小，不足以区分不同的样本统计。

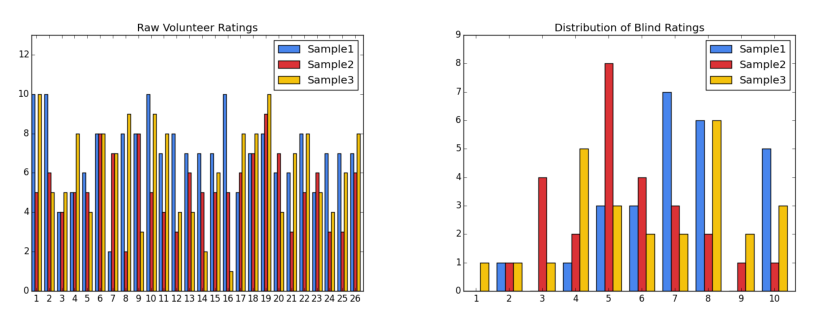


图9↑：（左）：26个志愿者的每个序列的原始投票值。（右）：评级直方图。

7.结论 和 未来工作

我们能够表明，多层LSTM、字符级语言模型应用于两个独立的数据表示的是能够产生音乐，至少这种比较在相当复杂的时间序列的概率密度的技术文献中是常见的。我们发现，我们的模型能够学习有意义的音乐结构。

这篇论文是在一个有趣的时间里产生的。在上周，谷歌宣布其Magenta program 5，一个tensorflow支持生成艺术的机器学习平台。谷歌还发布了一个90秒的电脑制作旋律和伴奏的鼓线。

考虑到最近对机器学习艺术的热情，我们希望通过引入更复杂的模型和数据表示来继续我们的工作，从而有效地捕捉底层的旋律结构。此外，我们认为，在制定一个更好的作品质量评估指标方面可以做更多的工作，只有这样，我们才能培养出真正能够作曲的模型！

1. word2vec 是 Google 于 2013 年开源推出的一个用于获取 word vector 的工具包，它简单、高效，因此引起了很多人的关注（原文解释：向量空间中词语表示的有效估计） [↑](#footnote-ref-1)
2. MIDI(Musical Instrument Digital Interface)乐器数字接口 ，是20 世纪80 年代初为解决电声乐器之间的通信问题而提出的。MIDI是编曲界最广泛的音乐标准格式，可称为“计算机能理解的乐谱”。它用音符的数字控制信号来记录音乐。 [↑](#footnote-ref-2)
3. 我们提取其他MIDI文件是从这2个网站：http://classicalmidiresource.com/,http://midiworld.com/, and http://piano-midi.de/ [↑](#footnote-ref-3)
4. [↑](#footnote-ref-4)
5. AWS - 亚马逊公司旗下云计算服务平台，为全世界各个国家和地区的客户提供一整套基础设施和云解决方案。 [↑](#footnote-ref-5)
6. t-SNE(t-distributed stochastic neighbor embedding)是用于**降维**的一种机器学习算法，是由 Laurens van der Maaten 和 Geoffrey Hinton在08年提出来。此外，t-SNE 是一种非线性降维算法，非常适用于高维数据降维到2维或者3维，进行可视化。t-SNE是由SNE(Stochastic Neighbor Embedding, SNE; Hinton and Roweis, 2002)发展而来。我们先介绍SNE的基本原理，之后再扩展到t-SNE。最后再看一下t-SNE的实现以及一些优化。 [↑](#footnote-ref-6)
7. 序列可以从这里下载：http://www-etud.iro.umontreal.ca/ ˜ boulanni/icml2012. 点击“mp3样本”链接 [↑](#footnote-ref-7)