# 基于LSTM神经网络架构利用MIDI短音乐段作为输入产生音乐

## 摘要

现如今越来越多的人开始研究自动编曲的方式，但是极少有在研究过程中使用神经网络作为架构来实现自动编曲。

本文研究的是基于长短期记忆网神经网络架构（LSTM），通过输入一段音乐后计算机预测获得一段较为贴近原有音调模式产生新的音乐。训练该神经网络的数据集为74首MIDI格式Led Zeppelin的歌曲。所有的MIDI文件数据都会转换到二维数组内，假设这个二维数组为a[ i ][ j ]，那么这个二维数组的i、j值分别是MIDI文件的音高和MIDI tick[[1]](#footnote-0)。数组的数据通过顺序进行选择最终用于神经网络的训练，目前如何选择数据大致有四种方式，其中一种方式是移除歌曲内无声的部分，通过对最终生成音乐的结构进行分析发现，这四种选择数据的方式都是有效的。通过让参加者听通过LSTM神经网络架构生成的音乐小样并将这些小样以听的时候的愉悦度作为标准进行分级，最终发现移除歌曲内无声的部分作为训练神经网络的数据生成的歌曲是最为成功的。神经网络难以学会如何在音乐方式段落之间进行过渡，在未来的研究中，通过尝试增加数据集的大小等方法来解决这个问题。

## 介绍

作曲自古就是人类的一大爱好，它的历史可以追溯到公元前200年古希腊的Seikilos epitaph，先存历史最悠久的编曲比赛。近代人们开始寻找自动化作曲的实现方式，最近关于自动作曲的研究发表于1960年。作曲的产生是出于人们对于听音乐进行娱乐和通过音乐调整子的情绪的需求。作曲是项耗时耗力的工作，如果能让这项工作变得自动化那么一定会很有价值。

目前有许多自动化作曲的尝试，包括通过语言处理工具利用音乐的语法模式来进行作曲，或者是通过离散数学分析不同程度优秀的音乐。最近在许多科学领域，深度学习取得了成功，基于深度学习的神经网络实现自动作曲是必要的。处于这样的想法，本文将探究如何利用深度学习来实现自动生成音乐。

### 1.1先前的研究

对于利用深度学习实现音乐的自动生成的研究在此之前是非常少的。

Chen和Miikkulainen希望能够找到一个神经网络可以用于查找音乐的结构。为了能够对目标有最好的预测结果，他们使用进化算法来完成这项任务。他们所编写的音调和节奏的进化算法能够完成依照正确的音乐结构创作音乐。但是产生的音乐过于简单，而且系统不能处理多种乐器的情况。

Eck and Schmidhuber研究了当时流行的神经网络架构的问题所在，LSTM神经网络在其他研究领域获得了成功，他们认为其在音乐生成也是可行的。之后他们利用LSTM神经网络输入蓝调的代表性和弦成功生成了蓝调。其重点放在了网络如何让生成的音乐拥有正确的结构和节奏。

Franklin验证了神经网络中过去的数据对于音乐生成的重要性。通过RNN不仅想要完成想要编写特定类型的音乐，而是想要取代复制、创作音乐。通过LSTM，又由于音乐是可再版的，通过重配和声，新的音乐就能被创作出来。通过替代和学习和弦（符合音乐结构）能够创造出新的音乐。

Sak ，Senior 和 Beaufays 三人一起探寻了RNN在语言识别的运用。RNN在语音识别方面做的非常成功。这项成就为神经网络直接处理音乐文件提供了技术支持，人们对于音频的处理不再只局限于基于文频的音乐数据。

Johnston基于LSTM循环网络用许多的音乐来训练神经网络（ABC符号文本形式），并且接收之前所有的元素（音乐文件内）作为下一特征预测的输入。通过循环这样的方式，新的歌曲能够被创作出来，创作出来的每一个元素都将返回到RNN中去。通过对不同结构的音乐输入后测试发现，音乐都可以被产生出来。但是这样的方式只能成功产生一些简单的音乐，随着输入的音乐变得越来越复杂，生成音乐的音调开始变得变那么准确，同时利用ABC符号来代替原有的音频文件也耗时过长，这并不算成功。

这五个基于神经网络简单实现音乐生成的研究为LSTM网络的建立提供了坚实的基础。但是，此次研究也是基于输入音乐的片段来产生音乐，它也不会使用基于语法模式而是用MIDI文件作为替代。与之最为类似的研究仅仅研究单声部的音乐，但是本文中使用的系统能够支持复调的音乐。此次研究也可以作为最近大火的深度机器学习领域的一个优秀的例子，为之后再深度学习方面的研究提供了不错的基础。

### 1.2 研究范围

此次研究的目的是编写一个基于数秒的旋律就能够自动生成音乐的程序。这个程序基于神经网络，并且该网络已经使用其他音乐文件进行过训练。根据输入的旋律，借助神经网络在其他音乐文件中学习到的权值，该程序能够预测这段音乐之后音高最有怎样的可能进行变化。将预测的旋律附加到原有输入的旋律中去，通过重复这样的步骤，音乐就生成了。

值得注意的是，音乐是被编码的，而系统编码时候，大多数被编码的音频文件都会在描述音乐音高的时候加入特殊的时间戳。

虽说有一个足够大的数据集，该网络能够学会如何在系统里工作，但是如果将它转换到一个系统，那么会变得更加容易被训练，也能够大大提高训练的精度、降低训练所需的时间。

如先前所言，许多产生音乐的方式都基于研究中所使用被编码的音乐所定义音乐编曲语法。绝多大多数的研究文献综述资料中也是这样做的，而本文中所使用的编码是MIDI格式，虽然说这种编码的表示学习起来比较困难。但是一种算法如果能够将文件内容转换到系统里去，那么神经网络将会更加容易被训练它产生的结果也会更加准确。

MIDI文件格式将会在2.2进行详细地描述，通过这种编码方式最为重要的是计算机能够知道音乐什么时候在播放，知道音乐每一Tick编码后的音高。这个算法能够将音乐文件转换到一个系统（特别是一个数组），将时间的表示方式从相对表示转换为绝对表示，每一个时间段转换为一个Tick。

本文中使用的神经网络是一个长短期神经网络（LSTM），是递归神经网络的一种特殊形式。在2.3将会被详细地解释。LSTM神经网络允许音乐文件顺序结构的输入输出，也可以只任何计算节点之间。由于音乐一般是顺序的，一段确定的顺序的音高会紧跟着其他的音高，因此做到这一点非常重要，即使是序列模式下这样的情况也存在（例如歌曲转换处或者是副歌部分）。

此篇论文关于数据集的内容在2.1。在解释完MIDI编码系统和神经网络的实现后，对于这个系统的一些变化的部分在2.4，然后紧接着在2.5主要讲的是如何在进程中预测新的音乐片段。后期处理在2.6，对研究结果的评测在3，关于此次研究的讨论和总结分别在4,5。

## 2 方法

### 2.1 数据集

对于此次研究，此次研究所使用的数据集为Led Zeppelin的74首曲子。每一首曲子都被编码为1型 MIDI文件（具体将在下一节进行讨论），文件获得于zeppelinmidi.com，网站上对于如何下载该文件做出了介绍。

对于这些文件所要研究的是歌曲的音轨，因为歌曲的音轨能够很好地体现歌曲的情况。这样能让机器学习具有更强的表达功能，也使得计算机能更好地学习和制作音乐。处于对产生结果准确性的考虑，选择单一的乐队作为数据集的输入。

结束对机器学习的训练后，选择四个不一样的文件来进行歌曲的预测。第一个文件是手工创建只包含四个单音的曲子。第二个文件是Darren Tate vs. Jono Grant一首简单结构多重复的trace曲—— Let the Light Shine In (Arty Remix)。第三个文件稍微复杂些，是一首爵士乐，Washington & Young的Stella by Starlight。最后一个文件是数据集中所包含的音乐——Stairway to Heaven。

### 2.2 MIDI Tick 数组

### 2.2.1 MIDI格式

MIDI将所要演奏的乐曲信息用[字节](https://baike.baidu.com/item/%E5%AD%97%E8%8A%82" \t "https://baike.baidu.com/item/MIDI/_blank)进行描述。譬如在某一时刻，使用什么乐器，以什么音符开始，以什么音调结束，加以什么伴奏等等，MIDI文件本身并不包含[波形](https://baike.baidu.com/item/%E6%B3%A2%E5%BD%A2" \t "https://baike.baidu.com/item/MIDI/_blank)数据。但是这样的格式是没有办法直接输入到机器学习中去，因此需要编写一个新的算法来将这个编码的乐曲转化为二维数组。一个轴所包含的是音乐的每Tick的的信息（用绝对代替相关），另外一个轴包所绘制的是音高。二维数组的值是音高的大小。

MIDI文件存储了多种events，其中包含着关于歌曲信息的Meta events（例如：歌名），非冗余events有NoteEvents, ControlChangeEvents, PitchWheelEvents, SysExEvents, ProgramChangeEvents 还有 EndOfTrackEvents。

NoteEvents可以控制特定音高的速度，还能够控制音高停止的事件。

其他Events控制发出音乐发声的方式（例如让音乐听起来像是其他类型乐器）。但是由于此次所使用来训练神经网络的音乐只有一种乐器，因此其他events所产生的影响应该被忽略。但是为了存储正确的时间结构，Tick值应该被存储下来。

MIDI有三种类型用来处理并行的轨道。一条轨道是多个events的集合，在1型MIDI文件中，每一种乐器都占有一条轨道。0型MIDI文件中不同的乐器都在一条轨道上，然后使用non-NoteEvents来进行控制。2型MIDI文件因为需要使用一个系统来向文件内添加音乐轨道，现在已经很少使用了。但是由于数据集中的文件都是1型MIDI，所以并用在意是不是单种乐器。因为1型MIDI文件是单音乐轨道，所以可以把他认为是单种乐器。解码系统也能使用0型MIDI文件（不在数据集中），但是会创建比1型MIDI文件更为杂乱的数组。

1. 一个四分音符的音长为480 Ticks，则480 Ticks为1秒，单个Tick的时间长度就是（1/480）秒。 [↑](#footnote-ref-0)