(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 113689835 B
(45)授权公告日 2024.12.24

(21)申请号 202010421429.3

(22)申请日 2020.05.18

(65)同一申请的已公布的文献号
申请公布号 CN 113689835 A

(43)申请公布日 2021.11.23

(73)专利权人 微软技术许可有限责任公司
地址 美国华盛顿州

(72)发明人 吴先超 王程元 雷沁颖 夏培军 徐元春

(74)专利代理机构 永新专利商标代理有限公司 72002
专利代理师 张立达

(51)Int.Cl.
G10H 1/00(2006.01)

(56)对比文件

Jean-Pierre Briot 等.Deep Learning Techniques for Music Generation -- A Survey.康奈尔大学网络文献.2017,77、103、136.

Yu-Siang Huang 等.Pop Music Transformer: Generating Music with Rhythm and Harmony.康奈尔大学网络文献.2020,1-7.

Yu-Siang Huang 等.Pop Music Transformer: Generating Music with Rhythm and Harmony.康奈尔大学网络文献.2020,1-7.

Ashish Vaswani 等.Attention is you need.康奈尔大学网络文献.2017,1-15.

审查员 唐松柏

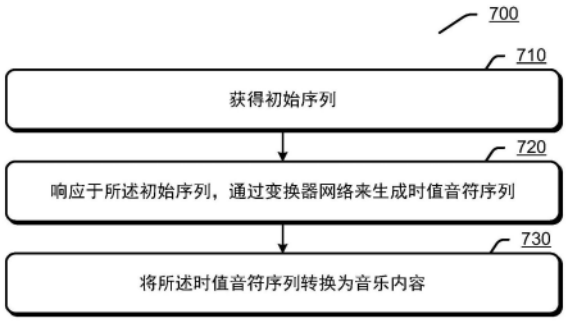
权利要求书3页 说明书13页 附图7页

(54)发明名称

用于自动音乐生成的方法和装置

(57)摘要

本公开提供了用于自动音乐生成的方法和装置。可以获得初始序列。可以响应于所述初始序列,通过变换器网络来生成时值音符序列。可以将所述时值音符序列转换为音乐内容。



CN 113689835 B

1.一种用于自动音乐生成的方法,包括:

获得初始序列;

响应于所述初始序列,通过变换器网络来生成时值音符序列;以及

将所述时值音符序列转换为音乐内容,其中:

所述时值音符序列包括:与从前一音符开始到当前音符开始的时值对应的第一序列,与从当前音符开始到当前音符关闭的时值对应的第二序列,与当前音符的音高对应的第三序列,以及与当前音符的力度对应的第四序列,

所述变换器网络包括:与所述第一序列对应的第一变换器子网络,与所述第二序列对应的第二变换器子网络,与所述第三序列对应的第三变换器子网络,以及与所述第四序列对应的第四变换器子网络,并且

所述第一序列和所述第二序列中至少之一还被输入到所述第三变换器子网络和所述第四变换器子网络中至少之一。

2.如权利要求1所述的方法,其中,

所述初始序列是随机产生的、接收到的、或者根据音乐片段所生成的。

3.如权利要求1所述的方法,其中,所述生成时值音符序列包括迭代地执行以下操作:

通过所述变换器网络,至少基于当前时值音符序列来预测下一个音符。

4.如权利要求1所述的方法,其中,

所述变换器网络是至少基于超长变换器来构建的。

5.如权利要求1所述的方法,其中,所述时值音符序列中的每个音符是以四元组来表示的,所述四元组包括:

从前一音符开始到当前音符开始的时值;

从当前音符开始到当前音符关闭的时值;

当前音符的音高;以及

当前音符的力度。

6.如权利要求1所述的方法,其中,

所述第三序列还被输入到所述第四变换器子网络。

7.如权利要求1所述的方法,还包括:

接收关于情感的指示和/或关于音乐风格的指示;

生成与所述情感对应的情感嵌入表示以及与所述音乐风格对应的风格嵌入表示;以及

将所述情感嵌入表示和/或所述风格嵌入表示输入到所述第一变换器子网络、所述第二变换器子网络、所述第三变换器子网络和所述第四变换器子网络中至少之一。

8.如权利要求1所述的方法,还包括:

接收关于音乐参数的指示,并且

其中,所述将所述时值音符序列转换为音乐内容是进一步基于所述音乐参数来执行的。

9.如权利要求8所述的方法,其中,

所述音乐参数包括以下至少之一:速度、节拍、以及长度。

10.如权利要求1所述的方法,还包括:

接收对所述音乐内容中的至少一个音符的调整指示;以及

响应于所述调整指示,更新所述音乐内容。

11.如权利要求1所述的方法,其中,

所述音乐内容是音乐设备数字接口(MIDI)文件。

12.一种用于自动音乐生成的装置,包括:

初始序列获得模块,用于获得初始序列;

时值音符序列生成模块,用于响应于所述初始序列,通过变换器网络来生成时值音符序列;以及

转换模块,用于将所述时值音符序列转换为音乐内容,其中:

所述时值音符序列包括:与从前一音符开始到当前音符开始的时值对应的第一序列,与从当前音符开始到当前音符关闭的时值对应的第二序列,与当前音符的音高对应的第三序列,以及与当前音符的力度对应的第四序列,

所述变换器网络包括:与所述第一序列对应的第一变换器子网络,与所述第二序列对应的第二变换器子网络,与所述第三序列对应的第三变换器子网络,以及与所述第四序列对应的第四变换器子网络,并且

所述第一序列和所述第二序列中至少之一还被输入到所述第三变换器子网络和所述第四变换器子网络中至少之一。

13.如权利要求12所述的装置,其中,所述生成时值音符序列包括迭代地执行以下操作:

通过所述变换器网络,至少基于当前时值音符序列来预测下一个音符。

14.如权利要求12所述的装置,其中,所述时值音符序列中的每个音符是以四元组来表示的,所述四元组包括:

从前一音符开始到当前音符开始的时值;

从当前音符开始到当前音符关闭的时值;

当前音符的音高;以及

当前音符的力度。

15.一种用于自动音乐生成的装置,包括:

至少一个处理器;以及

存储器,其存储计算机可执行指令,当所述计算机可执行指令被执行时使所述至少一个处理器:

获得初始序列,

响应于所述初始序列,通过变换器网络来生成时值音符序列,以及

将所述时值音符序列转换为音乐内容,其中:

所述时值音符序列包括:与从前一音符开始到当前音符开始的时值对应的第一序列,与从当前音符开始到当前音符关闭的时值对应的第二序列,与当前音符的音高对应的第三序列,以及与当前音符的力度对应的第四序列,

所述变换器网络包括:与所述第一序列对应的第一变换器子网络,与所述第二序列对应的第二变换器子网络,与所述第三序列对应的第三变换器子网络,以及与所述第四序列对应的第四变换器子网络,并且

所述第一序列和所述第二序列中至少之一还被输入到所述第三变换器子网络和所述

第四变换器子网络中至少之一。

用于自动音乐生成的方法和装置

背景技术

[0001] 音乐是一种得到广泛应用的艺术形式。在人们的生活中存在对大量高质量音乐的需求。音乐创作对于专业作曲者而言是具有挑战性的工作。近些年来,例如深度学习算法等人工智能(AI)技术被越来越多地用于自动音乐创作或生成。已经提出了一些AI音乐生成模型来自动地生成音乐内容,例如音乐设备数字接口(MIDI)文件等。

发明内容

[0002] 提供本发明内容以便介绍一组概念,这组概念将在以下的具体实施方式中做进一步描述。本发明内容并非旨在标识所保护主题的关键特征或必要特征,也不旨在用于限制所保护主题的范围。

[0003] 本公开的实施例提出了用于自动音乐生成的方法和装置。可以获得初始序列。可以响应于所述初始序列,通过变换器网络来生成时值音符序列。可以将所述时值音符序列转换为音乐内容。

[0004] 应当注意,以上一个或多个方面包括以下详细描述以及权利要求中具体指出的特征。下面的说明书及附图详细提出了所述一个或多个方面的某些说明性特征。这些特征仅仅指示可以实施各个方面的原理的多种方式,并且本公开旨在包括所有这些方面和其等同变换。

附图说明

[0005] 以下将结合附图描述所公开的多个方面,这些附图被提供用以说明而非限制所公开的多个方面。

[0006] 图1示出了根据实施例的自动音乐生成的示例性过程。

[0007] 图2示出了根据实施例的基于时值的音符表示的示例。

[0008] 图3示出了超长变换器(Transformer-XL)的示例性架构。

[0009] 图4示出了根据实施例的变换器网络的示例性架构。

[0010] 图5示出了根据实施例的变换器网络的示例性架构。

[0011] 图6示出了根据实施例的更新音乐内容的示例性过程。

[0012] 图7示出了根据实施例的用于自动音乐生成的示例性方法的流程。

[0013] 图8示出了根据实施例的用于自动音乐生成的示例性装置。

[0014] 图9示出了根据实施例的用于自动音乐生成的示例性装置。

具体实施方式

[0015] 现在将参考多种示例性实施方式来讨论本公开。应当理解,这些实施方式的讨论仅仅用于使得本领域技术人员能够更好地理解并从而实施本公开的实施例,而并非教导对本公开的范围的任何限制。

[0016] 在现有的AI音乐生成中,主流的方式包括通过借用自然语言处理领域的语言建模

的思想来对音符序列进行建模,以便在音乐生成过程中预测音符序列。

[0017] DeepJ被提出用于特定于风格的音乐生成,其可以针对例如巴洛克时期、古典时期、浪漫主义时期等的音乐风格,使用钢琴卷(roll)音符表示来训练双向长短期记忆(LSTM)网络。DeepJ能够以作曲风格的特定混合为条件来创作音乐。在DeepJ中将音符的钢琴卷表示用作MIDI音乐的密集表示。假设一首音乐是N╳T矩阵,其中,N是可弹奏的音符的数量,T是时间步骤的数量。考虑到在保持一个音符与重新弹奏一个音符之间的差异,音符弹奏和音符重新弹奏被联合地用于定义音符表示,其中,音符重新弹奏指在音符结束后立即重新弹奏该音符,而在连续的弹奏之间没有时间步骤。然而,钢琴卷实际上并非是密集的,这是因为在弹奏/重新弹奏矩阵中存在大量的零,在每个时间步骤期间只有少量音符被弹奏而所有其它音符是零。此外,难以采用钢琴卷音符表示来进行序列学习以对音符有效地建模。至少由于对音符表示的上述局限,基于DeepJ的音乐生成将是非常耗时的。

[0018] 变换器(transformer)技术也被提出用于音乐生成,其中,变换器是基于自注意力机制的序列模型,其在需要维持长范围相关性的任务中具有良好的性能。与例如LSTM的循环神经网络(RNN)相比,变换器在训练和推理阶段都更具并行化且更具可解释性。例如,音乐变换器(Music Transformer)被提出用于采用具有相对注意力机制的变换器来生成钢琴曲。Music Transformer是采用具有时间间隔的单个音符序列来训练的。基于时间或时间间隔对音符的表示将难以用于在一个或多个MIDI文件中对共享相同时值但不同速度的音符的相似度计算。此外,单个序列的使用将限制模型基于乐谱来学习音乐作曲信息,例如节奏模式、音符值等。相应地,在生成长持续时间的音乐时,Music Transformer将具有明显的衰减并且缺乏节奏稳定性。

[0019] 超长变换器(Transformer-XL)被提出用于使得变换器能够学习固定长度之外的依存性,而不干扰时间相关性。Transformer-XL可以通过段(segment)级别循环和相对位置编码来对极长的序列进行建模,使得其能够捕获长期依存性并且解决语言或音乐上下文碎片化问题。然而,现有的用于音乐生成的Transformer-XL仍然是采用具有时间间隔的单个音符序列来训练的,从而面临与上述Music Transformer类似的局限。

[0020] 本公开的实施例所提出的自动音乐生成能够快速自动地创作高质量的音乐并生成相应的音乐内容。在本文中,音乐内容可以广泛地指对音乐的各种实例化呈现,例如MIDI文件、乐谱等。尽管在以下讨论中多处以MIDI文件为例,但是应当理解,本公开的实施例可以通过类似方式而被应用于自动生成各种其它类型的音乐内容。

[0021] 在一个方面,取代基于时间或时间间隔来表示音符,本公开的实施例提出了基于时值(time value)的音符表示。基于时值的音符表示是一种相对音符表示,其在相对长度上对音符进行度量,而并非是在时间上表示音符。例如,可以在自动音乐生成中采用时值音符序列。时值音符序列中的每个音符是以四元组(four-tuple)来表示的。四元组可以包括:从前一音符开始(former note on)到当前音符开始(current note on)的时值;从当前音符开始到当前音符关闭(current note off)的时值;当前音符的音高(pitch);以及当前音符的力度(velocity)。基于上述的四元组定义,时值音符序列可以进一步被划分为四个序列:与从前一音符开始到当前音符开始的时值对应的序列,简化表示为On2On序列;与从当前音符开始到当前音符关闭的时值对应的序列,简化表示为On2Off序列;与当前音符的音高对应的序列,简称为音高序列;以及与当前音符的力度对应的序列,简称为力度序列。

[0022] 在一个方面,本公开的实施例提出了至少基于Transformer-XL构建的变换器网络,以用于预测时值音符序列。该变换器网络可以包括分别基于Transformer-XL来构建的四个变换器子网络。所述四个变换器子网络可以分别对应于时值音符序列所包括的On2On序列、On2Off序列、音高序列以及力度序列,并且是基于这四个序列而联合训练的。

[0023] 在一个方面,本公开的实施例可以在用户指定的生成条件下执行音乐生成。例如,用户可以为所要生成的音乐指定期望的情感和/或音乐风格。相应地,变换器网络可以在考虑情感和/或音乐风格的情况下预测时值音符序列。例如,用户可以提供关于音乐参数的指示,使得在将所预测的时值音符序列转换为音乐内容时,可以至少考虑用户所指定的音乐参数。

[0024] 在一个方面,本公开的实施例提出了对所生成的音乐内容的更新机制。由于人类在作曲时是基于音符时值来创作的,并且本公开的实施例也采用基于时值的音符表示,因此,所生成的音乐内容是容易理解且容易修改的。用户可以通过特定的音乐内容编辑平台来对所生成的音乐内容进行后编辑或后创作,从而实现音乐内容的更新。

[0025] 本公开的实施例可以快速自动地生成更高质量的、更长持续时间的音乐内容。所生成音乐的节奏稳定且丰富。所生成音乐的音符密度在长时间范围上是稳定的,而不会随着时间而出现明显的衰减,其中,音符密度可以指例如在每个时间窗口大小中的音符的数量。所生成音乐的音高分布更接近于人类创作的真实音乐。此外,本公开的实施例可以适用于生成单乐器多声部的音乐,并且所生成的音乐可以具有自动产生的和弦进行的特性。

[0026] 图1示出了根据实施例的自动音乐生成的示例性过程100。

[0027] 可以首先获得初始序列102。初始序列102是用于触发自动音乐生成的种子序列,其可以采用时值音符序列的形式。例如,初始序列102可以是一小段时值音符序列,以作为所要生成的音乐的起始。初始序列102可以基于如前所述的四元组来表示音符。可以通过各种方式来获得初始序列。在一种实现方式中,可以随机地产生初始序列102。在一种实现方式中,可以从例如用户处直接接收指定的初始序列102。在一种实现方式中,可以间接地获得初始序列102。例如,假设获得了一个音乐片段并且期望将该音乐片段作为所生成的音乐的起始,则可以在接收到该音乐片段后,从该音乐片段中提取出时值音符序列以作为初始序列102。

[0028] 可以将初始序列102提供给变换器网络110,以触发对时值音符序列的预测。变换器网络110可以包括与用于表示音符的四元组分别对应的四个变换器子网络,例如,用于处理On2On序列的子网络、用于处理On2Off序列的子网络、用于处理音高序列的子网络、用于处理力度序列的子网络等。每个子网络都可以是基于Transformer-XL来构建的。变换器网络110可以迭代地基于当前时值音符序列来预测下一个音符。

[0029] 在一种实现方式中,可选地,变换器网络110还可以在考虑情感和/或音乐风格的情况下预测时值音符序列。例如,可以接收关于情感和/或音乐风格104的指示。情感可以指期望所生成的音乐所表现出的情感类型,例如,高兴的、悲伤的等。音乐风格可以指所生成的音乐所属于的风格。在音乐领域可能存在各种对音乐风格的划分,例如,巴洛克、古典、浪漫主义、乡村、爵士等。本公开的实施例适用于任何音乐风格。可以将情感和/或音乐风格104转换为对应的潜在空间表示,例如情感嵌入表示和/或风格嵌入表示106,并且将情感嵌入表示和/或风格嵌入表示106提供给变换器网络110。从而,在预测每一个音符时,变换器

网络110可以将情感嵌入表示和/或风格嵌入表示106作为当前时值音符序列之外的附加输入。

[0030] 假设变换器网络110最终输出时值音符序列120。过程100可以进一步将时值音符序列120转换为音乐内容140。例如,可以通过转换模块130来执行该转换。转换模块130可以是基于预定规则132来运行的。预定规则132包括从时值音符序列向特定类型的音乐内容的映射关系。以音乐内容140是MIDI文件为例,预定规则132可以包括预先确定的从时值音符序列向MIDI文件的映射关系。转换模块130可以基于预定规则132,将从时值音符序列120中读取的信息映射为MIDI文件中的信息,从而最终输出MIDI文件。不同类型的音乐内容可能对应不同的预定规则,本公开的实施例不受到任何具体的预定规则的限制。

[0031] 在一种实现方式中,可选地,转换模块130还可以参考音乐参数108来执行转换。音乐参数108可以包括音乐创作所涉及的各种参数,例如,速度(tempo)、节拍(metre)、长度等。速度可以指拍子(beat)的速率,其可以被表示为bpm值,即,每分钟的拍子的数量。节拍可以指固定的强弱音循环重复的序列,其可以由例如节拍号来表示,例如,2/4、4/4等。长度可以指所期望的音乐的总长度。应当理解,以上仅列出了音乐参数的几个示例,本公开的实施例也适用于任何其它音乐参数。音乐参数108可以是用户根据自己的喜好而指定的,或者是默认设置的。过程100可以接收关于音乐参数的指示,并且将其提供给转换模块130。转换模块130在将时值音符序列120转换为音乐内容140时可以参考音乐参数108。例如,假设音乐参数指定了节拍为4/4,则转换模块130可以按照节拍“4/4”来将时值音符序列120转换为例如MIDI文件。

[0032] 应当理解,以上过程100仅仅是示例性的,取决于具体的应用需求和设计,可以对过程100进行任意形式的修改并且这些修改都将由本公开所涵盖。

[0033] 图2示出了根据实施例的基于时值的音符表示的示例。

[0034] 在视图200中示出了一个示例性的音乐片段。在视图200中,x轴表示时间,y轴表示音高。该音乐片段是在开启延音踏板的情况下以琶音演奏的C大调,其起始力度为80。音符202、音符204和音符206依次在第0秒、第0.5秒、第1.0秒处弹奏,其中,音符202的音高为60,音符204的音高为64,音符206的音高为67。在第2.0秒处,释放延音踏板,并且音符202、音符204和音符206结束。在第2.5秒处,以力度100弹奏音符208并持续0.5秒,音符208的音高为65。

[0035] 为了进行比较,在图2中示出了以Music Transformer中基于时间偏移的音符表示方式来对上述音乐片段进行表示的音符序列210。在Music Transformer中定义了用于音符表示的四种演奏事件(performance event),包括:“设置_力度(SET_VELOCITY)”,其表示对音符的力度的设置;“音符_开始(NOTE_ON)”,其表示音符的起始;“时间_偏移(TIME_SHIFT)”,其表示从前一事件到下一事件的时间间隔;以及“音符_关闭(NOTE_OFF)”,其表示音符的结束。因此,音符序列210也可以被视为演奏事件序列。

[0036] 作为示例,在音符序列200中的“设置_力度<80>,音符_开始<60>,时间_偏移<500>,音符_开始<64>…”表示:在第0秒处为后续的音符设置力度“80”,在第0秒处音高为“60”的音符202起始,在前一事件“音符_开始<60>”之后的500毫秒(即,第0.5秒处)出现下一事件“音符_开始<64>”,即,音高为“64”的音符204起始。作为示例,在音符序列200中的“…时间_偏移<1000>,音符_关闭<60>,音符_关闭<64>,音符_关闭<67>…”表示:在前一事件“音

符_开始<67>”之后的1000毫秒处(即,在第2.0秒处),音高为“60”的音符202、音高为“64”的音符204和音高为“67”的音符206都结束。作为示例,在音符序列200中的“…时间_偏移<500>,设置_力度<100>,音符_开始<65>…”表示:在前一事件“音符_关闭<67>”之后的500毫秒处(即,在第2.5秒处),为后续的音符设置力度“100”,并且音高为“67”的音符208起始。

[0037] 应当理解,Music Transformer中采用的“时间_偏移”可能引起时值信息的丢失。当使用时间间隔来表示音符时,在不同的速度下,一个音符将对应到不同的时间间隔,并且一个时间间隔也将对应到不同的音符。在演奏事件序列中,各种音符的音符_开始和音符_关闭混合在一起,其破坏了音符的独立性并且引起时值信息的丢失。音符_开始和音符_关闭本应当像括号一样在序列中配对,但是这并不能被基于事件长度或时间间隔的序列分段所保证,因为每个音符的起始时间和结束时间是通过基于时间_偏移计算音符_开始和音符_关闭来获得的。这样的计算过程将引起例如不稳定的节奏、快速的衰减等问题。

[0038] 在图2中示出了以根据本公开实施例的基于时值的音符表示方式来对音乐片段进行表示的时值音符序列220。在时值音符序列220中,以四元组来表示每个音符。例如,四元组“(0,1.0,60,80)”表示音符202,四元组“(0.25,0.75,64,80)”表示音符204,四元组“(0.25,0.5,67,80)”表示音符206,并且四元组“(0.75,0.25,65,100)”表示音符208。以用于表示音符202的四元组“(0,1.0,60,80)”为例,“0”表示从前一音符开始到当前音符(即,音符202)开始的时值为0,“1.0”表示从当前音符开始到当前音符关闭的时值为1.0,“60”表示当前音符的音高为60,并且“80”表示当前音符的力度为“80”。以用于表示音符204的四元组“(0.25,0.75,64,80)”为例,“0.25”表示从前一音符(即,音符202)开始到当前音符(即,音符204)开始的时值为0.25,“0.75”表示从当前音符开始到当前音符关闭的时值为0.75,“64”表示当前音符的音高为64,并且“80”表示当前音符的力度为“80”。

[0039] 时值可以是目标量相对于全音符的比率,所述目标量例如为从前一音符开始到当前音符开始、从当前音符开始到当前音符关闭等。在一种实现方式中,可以基于“时值=(速度╳持续时间)/(60╳4)”来计算时值,其中,速度表示bpm值,持续时间表示目标量持续了多少秒,“60”表示60秒以用于将以分钟度量的速度转换为以秒度量的速度,并且“除以4”用于转变成相对全音符的比率。作为示例,假设在图2中速度为120bpm,通过基于上述公式来计算“(120╳2)/(60╳4)”,可以确定在音符202的四元组中用于表示从当前音符开始到当前音符关闭的时值为“1.0”。作为示例,通过基于上述公式来计算“(120╳0.5)/(60╳4)”,可以确定在音符204的四元组中用于表示从前一音符开始到当前音符开始的时值为“0.25”。

[0040] 在图2中还示出了用于对音乐片段进行表示的时值音符序列230,其是对时值音符序列220的变型。如前所述,时值音符序列220采用了时值的浮点表示,例如,1.0、0.25等,而时值音符序列230采用了时值的整数表示。可以通过将时值的浮点表示乘以整数化倍数来得到时值的整数表示。示例性地,可以通过“整数化倍数=量化粒度╳3”来计算整数化倍数,其中,量化粒度的值可以是例如对应于128分音符的“128”、对应于64分音符的“64”等,并且数字“3”是在考虑三连音情况下的因子。应当理解,取决于具体的应用,上述计算整数化倍数的公式可以以任意形式进行改变,例如,可以采用任意的量化粒度、可以考虑除三连音之外的任何其它因子等。以量化粒度为128为例,可以计算出整数化倍数为384,相应地,可以将时值音符序列220中的时值“1.0”转换为时值音符序列230中的时值“384”、将时值音

符序列220中的时值“0.25”转换为时值音符序列230中的“96”、等等。通过采用时值的整数表示,可以进一步提高数据计算和处理效率。

[0041] 应当理解,图2中的时值音符序列220和230可以进而被划分或转换为0n20n序列、0n20ff序列、音高序列和力度序列。0n20n序列可以由时值音符序列中每个音符的四元组的第一个项形成,0n20ff序列可以由时值音符序列中每个音符的四元组的第二个项形成,音高序列可以由时值音符序列中每个音符的四元组的第三个项形成,并且力度序列可以由时值音符序列中每个音符的四元组的第四个项形成。以时值音符序列220为例,其可以被划分为0n20n序列{0,0.25,0.25,0.75}、0n20ff序列{1.0,0.75,0.5,0.25}、音高序列{60,64,67,65}和力度序列{80,80,80,100}。此外,应当理解,以上列出的四元组中的四个项的顺序是示例性的,本公开的实施例可以涵盖这四个项的任何其它排列顺序。

[0042] 根据本公开实施例的转换器网络可以是部分地基于已知的Transformer-XL来构建的。Transformer-XL可以通过段级别循环和相对位置编码来捕获长期依存性并且解决语言或音乐上下文碎片化问题。

[0043] 在N层的Transformer-XL中,假设$s_{\tau-1}=[x_{\tau-1,1},\ldots,x_{\tau-1,L}]$是具有长度L的段,例如自然语言中的L个词语或音乐中的L个音符,并且$\mathbf{h}_{\tau-1}^{n-1}\in\mathbb{R}^{L\times d}$是与$s_{\tau-1}$对应的第n-1层的隐藏状态序列,其中d是隐藏层的维度,$n\in N$。对于下一个段s_{τ},对应的第n个隐藏层的隐藏状态可以计算为:

[0044] $$\tilde{\mathbf{h}}_{\tau}^{n-1}=[\mathrm{SG}(\mathbf{h}_{\tau-1}^{n-1})\circ\mathbf{h}_{\tau}^{n-1}] \qquad \text{公式(1)}$$

[0045] $$\mathbf{q}_{\tau}^{n},\mathbf{k}_{\tau}^{n},\mathbf{v}_{\tau}^{n}=\mathbf{h}_{\tau}^{n-1}\mathbf{W}_{q}^{\top},\tilde{\mathbf{h}}_{\tau}^{n-1}\mathbf{W}_{k,(E)}^{\top},\tilde{\mathbf{h}}_{\tau}^{n-1}\mathbf{W}_{v}^{\top} \qquad \text{公式(2)}$$

[0046] $$\mathbf{h}_{\tau}^{n}=\text{Transformer-Layer}(\mathbf{q}_{\tau}^{n},\mathbf{k}_{\tau}^{n},\mathbf{v}_{\tau}^{n}) \qquad \text{公式(3)}$$

[0047] 其中,函数SG(•)表示停止梯度,即,$\mathbf{h}_{\tau-1}^{n-1}$的梯度不会基于下一个段而更新,$(\mathbf{h}_{u}\circ\mathbf{h}_{v})$表示两个隐藏序列沿着长度维度的级联,W.表示可训练的模型参数,并且Transformer-Layer(•)表示经由变换器中的层的处理。

[0048] 与传统的变换器相比,此处主要的更新包括将前一个段的第n-1层的隐藏状态序列$\mathbf{h}_{\tau-1}^{n-1}$用于计算中间序列$\tilde{\mathbf{h}}_{\tau}^{n-1}$,并且将$\tilde{\mathbf{h}}_{\tau}^{n-1}$进一步用于计算将要利用查询序列$\mathbf{q}_{\tau}^{n}$来检索的扩展上下文增强序列$\mathbf{k}_{\tau}^{n}$和$\mathbf{v}_{\tau}^{n}$。

[0049] 以上公式示出的由Transformer-XL所应用的循环机制类似于针对每两个连续的段所执行的$\mathbf{h}_{\tau}^{n}=\text{Recurrent}(\mathbf{h}_{\tau-1}^{n-1},\mathbf{h}_{\tau}^{n-1})$,其中Recurrent(•)表示循环机制。这实际上创建了在各个变换器层的隐藏状态中的段级别循环。从而,允许上下文信息被应用到两个段之外。

[0050] 在标准的变换器中,查询向量$q_{i}=(\mathbf{E}_{x_{i}}+\mathbf{U}_{i})$是嵌入向量$\mathbf{E}_{x_{i}}$与第i个绝对位置编码$U_{i}$的和,键(key)向量$k_{j}=(\mathbf{E}_{x_{j}}+\mathbf{U}_{j})$是嵌入向量$\mathbf{E}_{x_{j}}$与第j个绝对位置编码$U_{j}$的和,并且在同一个段内在查询向量与键向量之间的注意力分数可以被分解为:

[0051] $$\mathbf{A}_{i,j}^{\text{abs}} = q_i^{\top} k_j = (\mathbf{E}_{x_i}^{\top} + \mathbf{U}_i^{\top})\mathbf{W}_q^{\top}\mathbf{W}_k(\mathbf{E}_{x_j} + \mathbf{U}_j) \qquad \text{公式（4）}$$

[0052] 其中,“abs”是绝对位置编码的缩写。绝对位置编码的缺点在于其并不能区分同一位置在不同段中呈现的差异。

[0053] 遵循相对位置编码的思想,在Transformer-XL中引入了相对距离R_{i-j}以描述q_i和k_j之间的相对位置嵌入。此处,R是不具有可学习参数的正弦(sinusoid)编码矩阵。相对注意力分数可以计算为:

[0054] $$\mathbf{A}_{i,j}^{\text{rel}} = (\mathbf{E}_{x_i}^{\top}\mathbf{W}_q^{\top} + u^{\top}:v^{\top})(\mathbf{W}_{k,E}\mathbf{E}_{x_j} + \mathbf{W}_{k,R}\mathbf{R}_{i-j}) \qquad \text{公式（5）}$$

[0055] 其中,“rel”是相对位置编码的缩写,两个可训练的向量$u, v \in \mathbb{R}^d$被用于替代$\mathbf{U}_i^{\top}\mathbf{W}_q^{\top}$,并且被分别用于与$\mathbf{W}_{k,E}\mathbf{E}_{x_j}$和$W_{k,R}R_{i-j}$相乘。此外,$W_k$被故意地分成两个权重矩阵$W_{k,E}$和$W_{k,R}$,以便分别与基于内容和基于位置的键向量相乘。

[0056] 从而,公式(3)中采用了相对位置编码机制的第n个Transformer-Layer可以被计算为:

[0057] $$\mathbf{A}_{\tau,i,j}^{\text{rel},n} = (\mathbf{q}_{\tau,i}^{n\top} + u^{\top}:v^{\top})(\mathbf{k}_{\tau,j}^{n} + \mathbf{W}_{k,R}^{n}\mathbf{R}_{i-j}) \qquad \text{公式（6）}$$

[0058] $$\mathbf{a}_{\tau}^{n} = \text{Masked-Softmax}(\mathbf{A}_{\tau}^{\text{rel,n}})\mathbf{v}_{\tau}^{n} \qquad \text{公式（7）}$$

[0059] $$\mathbf{o}_{\tau}^{n} = \text{LayerNorm}(\text{Linear}(\mathbf{a}_{\tau}^{n}) + \mathbf{h}_{\tau}^{n-1}) \qquad \text{公式（8）}$$

[0060] $$\mathbf{b}_{\tau}^{n} = \text{Positionwise-Feed-Forward}(\mathbf{o}_{\tau}^{n}) \qquad \text{公式（9）}$$

[0061] $$\mathbf{h}_{\tau}^{n} = \text{LayerNorm}(\text{Linear}(\mathbf{b}_{\tau}^{n}) + \mathbf{b}_{\tau}^{n}) \qquad \text{公式（10）}$$

[0062] 其中,Masked-Softmax(•)表示执行遮盖(mask)和Softmax处理,Linear(•)表示执行线性变换,LayerNorm(•)表示正则层的处理,并且Positionwise-Feed-Forward(•)表示执行位置前馈。

[0063] 图3示出了Transformer-XL的示例性架构300。

[0064] 输入302可以被传递到嵌入层310以获得嵌入表示。在320处,可以执行相对位置编码。在322处的叠加的输出可以被提供给Transformer-XL的记忆敏感模块330。模块330可以被重复N次。每个模块330可以进而包括:具有记忆的遮盖相对多头注意力模块332,其可以基于例如公式(1)、(2)、(5)、(6)、(7)等的处理;叠加和正则层334,其可以基于例如公式(8)的处理;前馈层336,其可以基于例如公式(9)的处理;以及叠加和正则层338,其可以基于例如公式(10)的处理。N个模块330的输出可以被提供到线性层340以执行线性映射。线性层340的输出可以被提供给Softmax层350,以便获得所预测序列的概率304。

[0065] 应当理解,架构300仅仅是示例性的,基于具体的需求和设计,可以对架构300中的任意模块、层等进行任何方式的修改。

[0066] 根据本公开实施例的变换器网络可以是至少基于例如图3所示的Transformer-XL来构建的。图4示出了根据实施例的变换器网络的示例性架构400。

[0067] 在架构400中,变换器网络可以包括分别基于Transformer-XL来构建的四个变换

器子网络。所述四个变换器子网络可以分别处理时值音符序列410所包括的0n20n序列420、0n20ff序列430、音高序列440以及力度序列450。

[0068] 在对应于0n20n序列420的变换器子网络中,可以首先通过嵌入层421来获得0n20n序列的嵌入表示。该0n20n序列的嵌入表示可以进而被传递通过424处的相对位置编码、N个记忆敏感模块425、线性层426、Softmax层427等。在424处的相对位置编码可以对应于图3中的320处的相对位置编码,记忆敏感模块425可以对应于图3中的模块330,线性层426可以对应于图3中的线性层340,并且Softmax层427可以对应于图3中的Softmax层350。Softmax层427可以输出用于下一个音符的0n20n候选时值的概率。可以选择具有最高概率的0n20n候选时值作为下一个音符的0n20n时值。可选地,也可以从概率排名最高的多个0n20n候选时值中随机地选择下一个音符的0n20n时值。

[0069] 在对应于0n20ff序列430的变换器子网络中,可以首先通过嵌入层431来获得0n20ff序列的嵌入表示。该0n20ff序列的嵌入表示可以进而被传递通过434处的相对位置编码、N个记忆敏感模块435、线性层436、Softmax层437等。在434处的相对位置编码可以对应于图3中的320处的相对位置编码,记忆敏感模块435可以对应于图3中的模块330,线性层436可以对应于图3中的线性层340,并且Softmax层437可以对应于图3中的Softmax层350。Softmax层437可以输出用于下一个音符的0n20ff候选时值的概率。可以选择具有最高概率的0n20ff候选时值作为下一个音符的0n20ff时值。可选地,也可以从概率排名最高的多个0n20ff候选时值中随机地选择下一个音符的0n20ff时值。

[0070] 在对应于音高序列440的变换器子网络中,可以首先通过嵌入层441来获得音高序列的嵌入表示。根据本公开的实施例,还可以将0n20n序列和0n20ff序列中至少之一输入到对应于音高序列440的变换器子网络,使得下一个音符的音高可以是在0n20n序列和0n20ff序列中至少之一的影响下进行预测的。例如,在442处,可以将音高序列的嵌入表示与影响因子414进行级联,该影响因子414可以包括通过嵌入层421所获得的0n20n序列的嵌入表示和/或通过嵌入层431所获得的0n20ff序列的嵌入表示。应当理解,在442处也可以通过任何其它方式来进行多个嵌入表示之间的组合,例如,通过叠加等。此外,可选地,可以通过线性层443来对442处的级联的输出进行维度变换。线性层443的输出可以进而被传递通过444处的相对位置编码、N个记忆敏感模块445、线性层446、Softmax层447等。在444处的相对位置编码可以对应于图3中的320处的相对位置编码,记忆敏感模块445可以对应于图3中的模块330,线性层446可以对应于图3中的线性层340,并且Softmax层447可以对应于图3中的Softmax层350。Softmax层447可以输出用于下一个音符的候选音高的概率。可以选择具有最高概率的候选音高作为下一个音符的音高。可选地,也可以从概率排名最高的多个候选音高中随机地选择下一个音符的音高。

[0071] 在对应于力度序列450的变换器子网络中,可以首先通过嵌入层451来获得力度序列的嵌入表示。根据本公开的实施例,还可以将0n20n序列和0n20ff序列中至少之一输入到对应于力度序列450的变换器子网络,使得下一个音符的力度可以是在0n20n序列和0n20ff序列中至少之一的影响下进行预测的。此外,可选地,还可以将音高序列输入到对应于力度序列450的变换器子网络,使得下一个音符的力度可以进而是在音高序列的影响下进行预测的。例如,在452处,可以将力度序列的嵌入表示与影响因子415进行级联,该影响因子415可以包括通过嵌入层421所获得的0n20n序列的嵌入表示和/或通过嵌入层431所获得的

0n20ff序列的嵌入表示,可选地,影响因子415还可以包括通过嵌入层441所获得的音高序列的嵌入表示。应当理解,在452处也可以通过任何其它方式来进行多个嵌入表示之间的组合。可选地,可以通过线性层453来对452处的级联的输出进行维度变换。线性层453的输出可以进而被传递通过454处的相对位置编码、N个记忆敏感模块455、线性层456、Softmax层457等。在454处的相对位置编码可以对应于图3中的320处的相对位置编码,记忆敏感模块455可以对应于图3中的模块330,线性层456可以对应于图3中的线性层340,并且Softmax层457可以对应于图3中的Softmax层350。Softmax层457可以输出用于下一个音符的候选力度的概率。可以选择具有最高概率的候选力度作为下一个音符的音高。可选地,也可以从概率排名最高的多个候选力度中随机地选择下一个音符的力度。

[0072] 通过四个转换器子网络所确定的0n20n时值、0n20ff时值、音高和力度可以一起组成四元组,以表示所预测的下一个音符460。进而,该预测的音符460可以被添加到时值音符序列410中以形成经更新的时值音符序列。该经更新的时值音符序列可以再次通过架构400来预测下一个音符。通过迭代地基于架构400来进行预测,可以最终获得与所要生成的音乐对应的时值音符序列。

[0073] 在架构400中,通过将0n20n序列和0n20ff序列中至少之一输入到用于处理音高序列的变换器子网络和用于处理力度序列的变换器子网络中至少之一,可以使得音高和/或力度的预测也是至少考虑到时值信息的。

[0074] 在训练过程中,可以计算每个变换器子网络的交叉熵损失,并且将四个变换器子网络的各自的交叉熵损失组合为全局损失,以便进行目标损失优化。从而,四个变换器子网络既保持相对独立性,又存在相互影响。在一个方面,该相互影响存在于输入端,例如,经由442处的级联和/或452处的级联进行的嵌入表示组合,在另一个方面,该相互影响存在于输出端的损失的组合。

[0075] 应当理解,以上架构400仅仅是示例性的,取决于具体的应用需求和设计,可以对架构400进行任意形式的修改并且这些修改都将由本公开所涵盖。

[0076] 图5示出了根据实施例的变换器网络的示例性架构500。架构500是对图4中的架构400的变型。在图5和图4中,相同的参考标号对应于相同的组件。在架构500中,在基于变换器网络预测时值音符序列的过程中进一步考虑了音乐风格和/或情感的因素,使得所预测的时值音符序列可以遵循特定的音乐风格和/或情感。

[0077] 可以获得指定的音乐风格502,并且通过嵌入层504获得与音乐风格502对应的风格嵌入表示。可以获得指定的情感506,并且通过嵌入层508获得与情感506对应的情感嵌入表示。然后,可以将风格嵌入表示和/或情感嵌入表示提供到四个变换器子网络中至少之一。

[0078] 在对应于0n20n序列的变换器子网络中,可以在522处对0n20n序列的嵌入表示与影响因子512进行级联,该影响因子512可以包括风格嵌入表示和情感嵌入表示中至少之一。然后,可以通过可选的线性层523来对522处的级联的输出进行维度变换,并将线性层523的输出提供给后续的处理。

[0079] 在对应于0n20ff序列的变换器子网络中,可以在532处对0n20ff序列的嵌入表示与影响因子513进行级联,该影响因子513可以包括风格嵌入表示和情感嵌入表示中至少之一。然后,可以通过可选的线性层533来对532处的级联的输出进行维度变换,并将线性层

533的输出提供给后续的处理。

[0080] 在对应于音高序列的变换器子网络中,影响因子514除了可以包括图4中的影响因子414外,还可以包括风格嵌入表示和情感嵌入表示中至少之一。

[0081] 在对应于力度序列的变换器子网络中,影响因子515除了可以包括图4中的影响因子415外,还可以包括风格嵌入表示和情感嵌入表示中至少之一。

[0082] 应当理解,以上架构500仅仅是示例性的,取决于具体的应用需求和设计,可以对架构500进行任意形式的修改并且这些修改都将由本公开所涵盖。

[0083] 以上结合图4和图5讨论了根据本公开实施例的变换器网络的示例性架构。在根据本公开实施例的自动音乐生成中,由变换器网络所输出的最终时值音符序列可以进而通过例如由图1的转换模块130所执行的转换操作而被转换为音乐内容。本公开的实施例可以支持对所生成的音乐内容进行更新,例如,由于根据本公开实施例所生成的音乐内容是容易辨识和理解的,因此可以方便地对所生成的音乐内容进行编辑或修改。

[0084] 图6示出了根据实施例的更新音乐内容的示例性过程600。

[0085] 假设针对初始序列602,通过在610处执行自动音乐生成而创建了音乐内容604。在610处的自动音乐生成可以是根据上述的本公开的实施例来执行的。

[0086] 音乐内容604可以被提供给音乐内容编辑平台620。音乐内容编辑平台620可以是支持对音乐内容604进行呈现、修改等操作的应用软件、网站等。假设音乐内容604是MIDI文件,则音乐内容编辑平台620可以是例如能够对MIDI文件进行创作和编辑的应用软件。音乐内容编辑平台620可以包括与用户进行交互的用户界面。通过该用户界面,可以向用户提供和呈现音乐内容604,并且可以接收用户对于音乐内容的至少一部分的调整指示606。例如,如果用户对于音乐内容604中的一部分不满意或者想要对一个或多个音符进行修改,则用户可以通过用户界面输入调整指示604。调整指示604可以包括对音乐中涉及的各种参数的修改或设置,所述参数可以包括例如速度、音高、力度等。

[0087] 以下以MIDI文件为例。现有的Music Transformer或基于单个音符序列的Transformer-XL是基于时间或时间间隔来对音符进行表示的,因此在所生成的MIDI文件中,音符并不能被有效地量化到网格(grid)或与网格对齐,从而用户难以在MIDI文件中辨识特定的音符,并且难以设置MIDI控制器和编排音乐。与此不同,由于本公开的实施例是基于时值来表示音符的,因此在所生成的MIDI文件中,音符将恰当地量化到网格,从而用户可以容易地辨识音符并且进行相应的参数修改。

[0088] 假设通过音乐内容编辑平台620接收到针对至少一个音符的调整指示606,则可以响应于该调整指示606来对该音符进行调整,以获得经调整的音符632。可以利用经调整的音符632来替代音乐内容604中原来的音符,以便形成更新的音乐内容608。

[0089] 基于调整指示进行调整的操作可以被迭代地执行,从而实现对所生成的音乐内容的不断更新。应当理解,过程600中的任何步骤和处理都是示例性的,取决于具体的应用需求和设计,可以对过程600进行任意形式的修改并且这些修改都将由本公开所涵盖。

[0090] 图7示出了根据实施例的用于自动音乐生成的示例性方法700的流程。

[0091] 在710处,可以获得初始序列。

[0092] 在720处,可以响应于所述初始序列,通过变换器网络来生成时值音符序列。

[0093] 在730处,可以将所述时值音符序列转换为音乐内容。

[0094] 在一种实现方式中,所述初始序列可以是随机产生的、接收到的、或者根据音乐片段所生成的。

[0095] 在一种实现方式中,所述生成时值音符序列可以包括迭代地执行以下操作:通过所述变换器网络,至少基于当前时值音符序列来预测下一个音符。

[0096] 在一种实现方式中,所述变换器网络可以是至少基于超长变换器来构建的。

[0097] 在一种实现方式中,所述时值音符序列中的每个音符可以是以四元组来表示的。所述四元组可以包括:从前一音符开始到当前音符开始的时值;从当前音符开始到当前音符关闭的时值;当前音符的音高;以及当前音符的力度。

[0098] 所述时值音符序列可以包括:与从前一音符开始到当前音符开始的时值对应的第一序列;与从当前音符开始到当前音符关闭的时值对应的第二序列;与当前音符的音高对应的第三序列;以及与当前音符的力度对应的第四序列。

[0099] 所述变换器网络可以包括:与所述第一序列对应的第一变换器子网络;与所述第二序列对应的第二变换器子网络;与所述第三序列对应的第三变换器子网络;以及与所述第四序列对应的第四变换器子网络。

[0100] 所述第一序列和所述第二序列中至少之一还可以被输入到所述第三变换器子网络和所述第四变换器子网络中至少之一。

[0101] 所述第三序列还可以被输入到所述第四变换器子网络。

[0102] 方法700还可以包括:接收关于情感的指示和/或关于音乐风格的指示;生成与所述情感对应的情感嵌入表示以及与所述音乐风格对应的风格嵌入表示;以及将所述情感嵌入表示和/或所述风格嵌入表示输入到所述第一变换器子网络、所述第二变换器子网络、所述第三变换器子网络和所述第四变换器子网络中至少之一。

[0103] 在一种实现方式中,方法700还可以包括:接收关于音乐参数的指示。所述将所述时值音符序列转换为音乐内容可以是进一步基于所述音乐参数来执行的。

[0104] 所述音乐参数可以包括以下至少之一:速度、节拍、以及长度。

[0105] 在一种实现方式中,方法700还可以包括:接收对所述音乐内容中的至少一个音符的调整指示;以及响应于所述调整指示,更新所述音乐内容。

[0106] 在一种实现方式中,所述音乐内容可以是MIDI文件。

[0107] 应当理解,方法700还可以包括根据上述本公开实施例的用于自动音乐生成的任何步骤/过程。

[0108] 图8示出了根据实施例的用于自动音乐生成的示例性装置800。

[0109] 装置800可以包括:初始序列获得模块810,用于获得初始序列;时值音符序列生成模块820,用于响应于所述初始序列,通过变换器网络来生成时值音符序列;以及转换模块830,用于将所述时值音符序列转换为音乐内容。

[0110] 在一种实现方式中,所述生成时值音符序列可以包括迭代地执行以下操作:通过所述变换器网络,至少基于当前时值音符序列来预测下一个音符。

[0111] 在一种实现方式中,所述时值音符序列中的每个音符可以是以四元组来表示的。所述四元组可以包括:从前一音符开始到当前音符开始的时值;从当前音符开始到当前音符关闭的时值;当前音符的音高;以及当前音符的力度。

[0112] 所述时值音符序列可以包括:与从前一音符开始到当前音符开始的时值对应的第

一序列;与从当前音符开始到当前音符关闭的时值对应的第二序列;与当前音符的音高对应的第三序列;以及与当前音符的力度对应的第四序列。

[0113] 所述变换器网络可以包括:与所述第一序列对应的第一变换器子网络;与所述第二序列对应的第二变换器子网络;与所述第三序列对应的第三变换器子网络;以及与所述第四序列对应的第四变换器子网络。

[0114] 此外,装置800还可以包括执行根据上述本公开实施例的用于自动音乐生成的方法的步骤的任何其它模块。

[0115] 图9示出了根据实施例的用于自动音乐生成的示例性装置900。

[0116] 装置900可以包括:至少一个处理器910;以及存储器920,其存储计算机可执行指令,当所述计算机可执行指令被执行时使所述至少一个处理器910:获得初始序列;响应于所述初始序列,通过变换器网络来生成时值音符序列;以及将所述时值音符序列转换为音乐内容。此外,处理器910还可以执行根据上述本公开实施例的用于自动音乐生成的方法的任何其它步骤/过程。

[0117] 本公开的实施例可以实施在非暂时性计算机可读介质中。该非暂时性计算机可读介质可以包括指令,当所述指令被执行时,使得一个或多个处理器执行根据上述本公开实施例的用于自动音乐生成的方法的任何操作。

[0118] 应当理解,以上描述的方法中的所有操作都仅仅是示例性的,本公开并不限制于方法中的任何操作或这些操作的顺序,而是应当涵盖在相同或相似构思下的所有其它等同变换。

[0119] 还应当理解,以上描述的装置中的所有模块都可以通过各种方式来实施。这些模块可以被实施为硬件、软件、或其组合。此外,这些模块中的任何模块可以在功能上被进一步划分成子模块或组合在一起。

[0120] 已经结合各种装置和方法描述了处理器。这些处理器可以使用电子硬件、计算机软件或其任意组合来实施。这些处理器是实施为硬件还是软件将取决于具体的应用以及施加在系统上的总体设计约束。作为示例,本公开中给出的处理器、处理器的任意部分、或者处理器的任意组合可以实施为微处理器、微控制器、数字信号处理器(DSP)、现场可编程门阵列(FPGA)、可编程逻辑器件(PLD)、状态机、门逻辑、分立硬件电路、以及配置用于执行在本公开中描述的各种功能的其它适合的处理部件。本公开给出的处理器、处理器的任意部分、或者处理器的任意组合的功能可以实施为由微处理器、微控制器、DSP或其它适合的平台所执行的软件。

[0121] 软件应当被广泛地视为表示指令、指令集、代码、代码段、程序代码、程序、子程序、软件模块、应用、软件应用、软件包、例程、子例程、对象、运行线程、过程、函数等。软件可以驻留在计算机可读介质中。计算机可读介质可以包括例如存储器,存储器可以例如为磁性存储设备(如,硬盘、软盘、磁条)、光盘、智能卡、闪存设备、随机存取存储器(RAM)、只读存储器(ROM)、可编程ROM(PROM)、可擦除PROM(EPROM)、电可擦除PROM(EEPROM)、寄存器或者可移动盘。尽管在本公开给出的多个方面中将存储器示出为是与处理器分离的,但是存储器也可以位于处理器内部(如,缓存或寄存器)。

[0122] 以上描述被提供用于使得本领域任何技术人员可以实施本文所描述的各个方面。这些方面的各种修改对于本领域技术人员是显而易见的,本文限定的一般性原理可以应用

于其它方面。因此,权利要求并非旨在被局限于本文示出的方面。关于本领域技术人员已知或即将获知的、对本公开所描述各个方面的元素的所有结构和功能上的等同变换,都将由权利要求所覆盖。

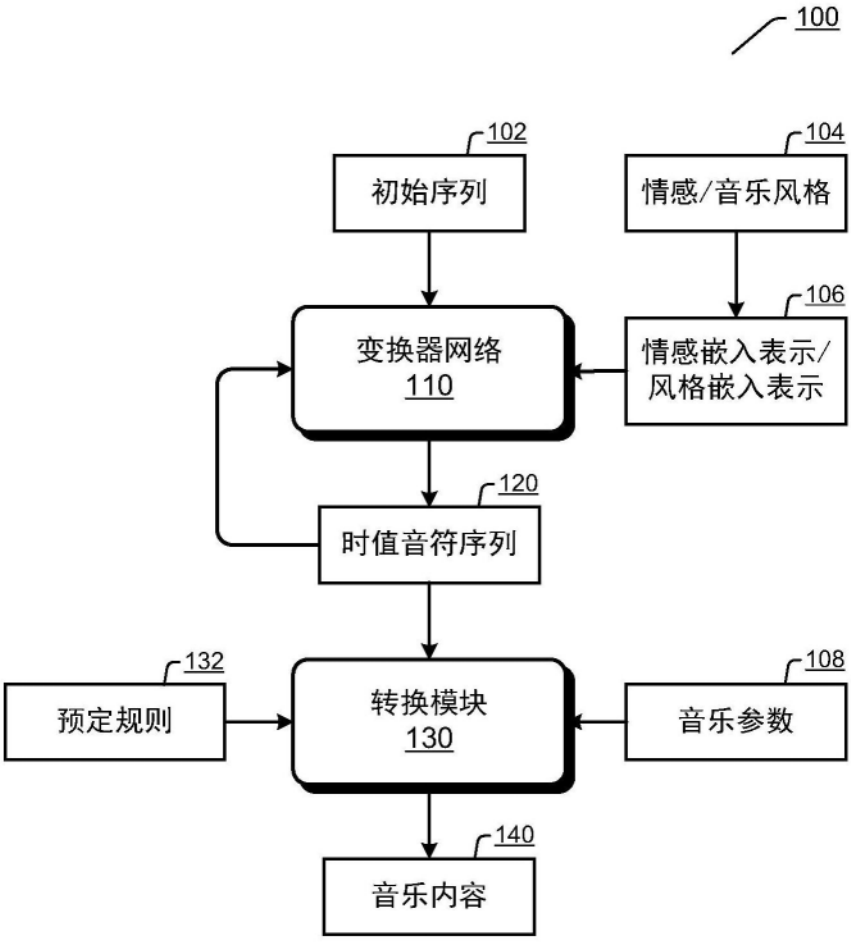
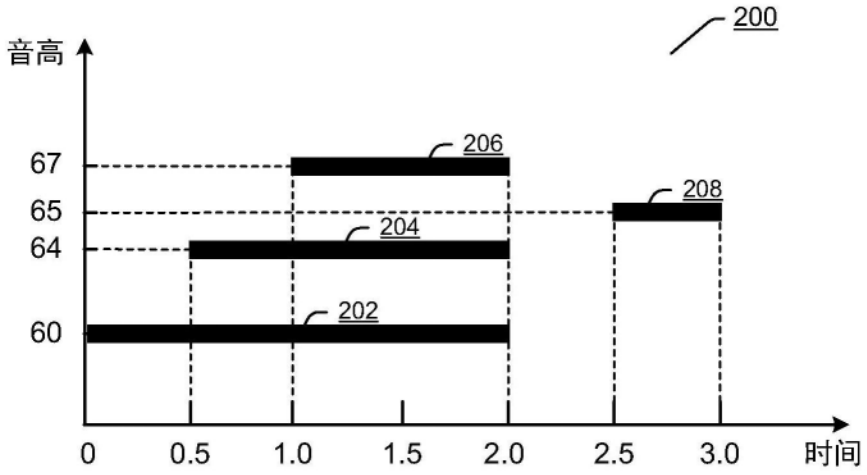

100
102
初始序列
104
情感/音乐风格
106
变换器网络
110
情感嵌入表示/
风格嵌入表示
120
时值音符序列
132
预定规则
转换模块
130
108
音乐参数
140
音乐内容

图1



210

设置_力度<80>, 音符_开始<60>, 时间_偏移<500>, 音符_开始<64>, 时间_偏移<500>, 音符_开始<67>, 时间_偏移<1000>, 音符_关闭<60>, 音符_关闭<64>, 音符_关闭<67>, 时间_偏移<500>, 设置_力度<100>, 音符_开始<65>, 时间_偏移<500>, 音符_关闭<65>

220

(0, 1.0, 60, 80) (0.25, 0.75, 64, 80) (0.25, 0.5, 67, 80) (0.75, 0.25, 65, 100)

230

(0, 384, 60, 80) (96, 288, 64, 80) (96, 192, 67, 80) (288, 96, 65, 100)

图2

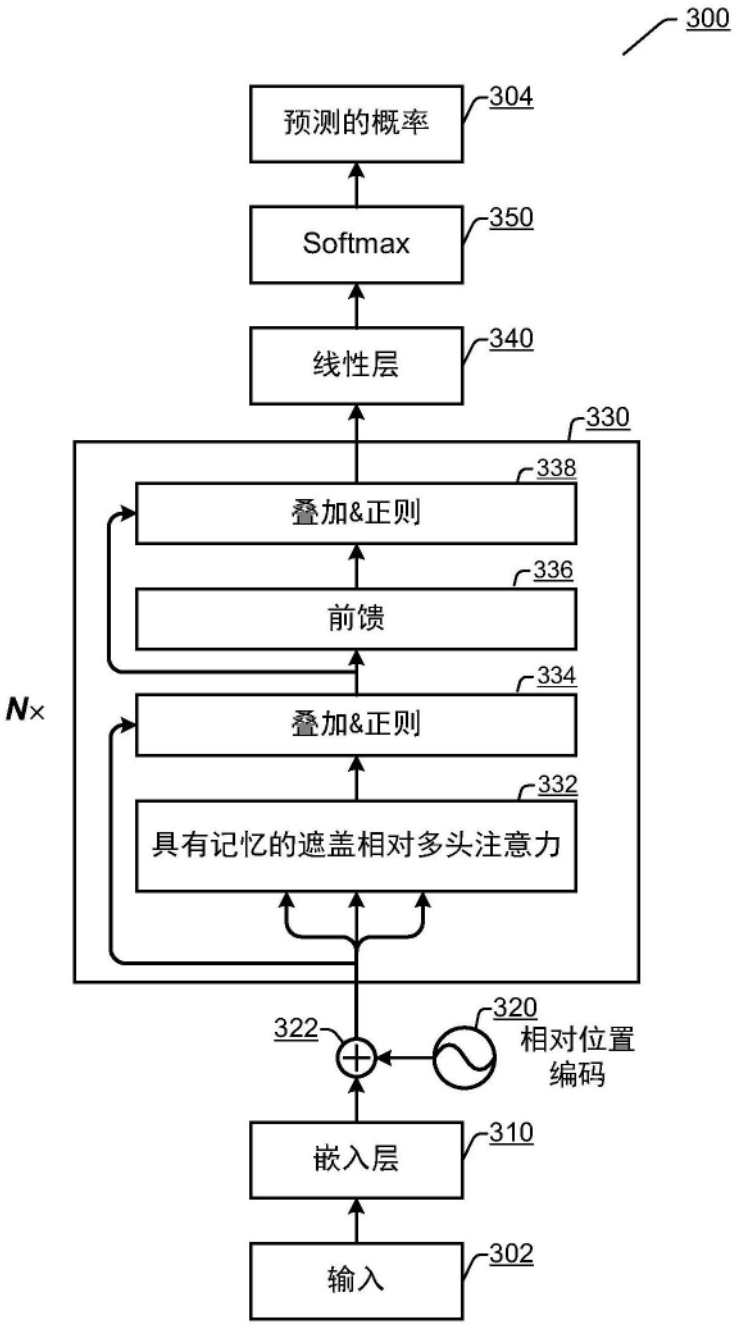

300
预测的概率
304
Softmax
350
线性层
340
330
338
叠加&正则
336
前馈
334
叠加&正则
N×
332
具有记忆的遮盖相对多头注意力
320
322
相对位置
编码
嵌入层
310
输入
302

图3

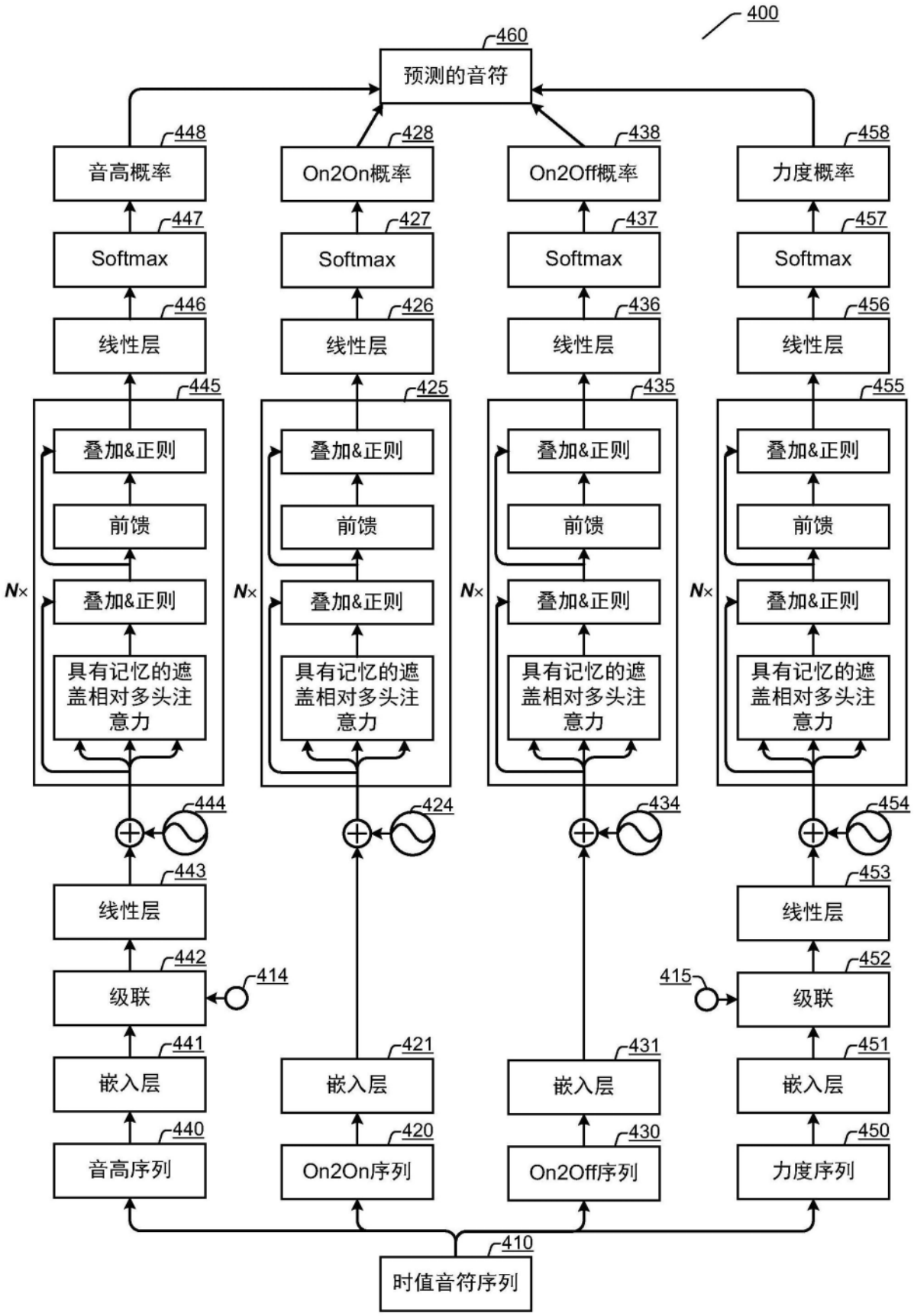

400
460
预测的音符
448
音高概率
428
On2On概率
438
On2Off概率
458
力度概率
447
Softmax
427
Softmax
437
Softmax
457
Softmax
446
线性层
426
线性层
436
线性层
456
线性层
445
425
435
455
叠加&正则
前馈
叠加&正则
具有记忆的遮盖相对多头注意力
N×
444
424
434
454
443
线性层
453
线性层
442
级联
414
415
452
级联
441
嵌入层
421
嵌入层
431
嵌入层
451
嵌入层
440
音高序列
420
On2On序列
430
On2Off序列
450
力度序列
410
时值音符序列

图4

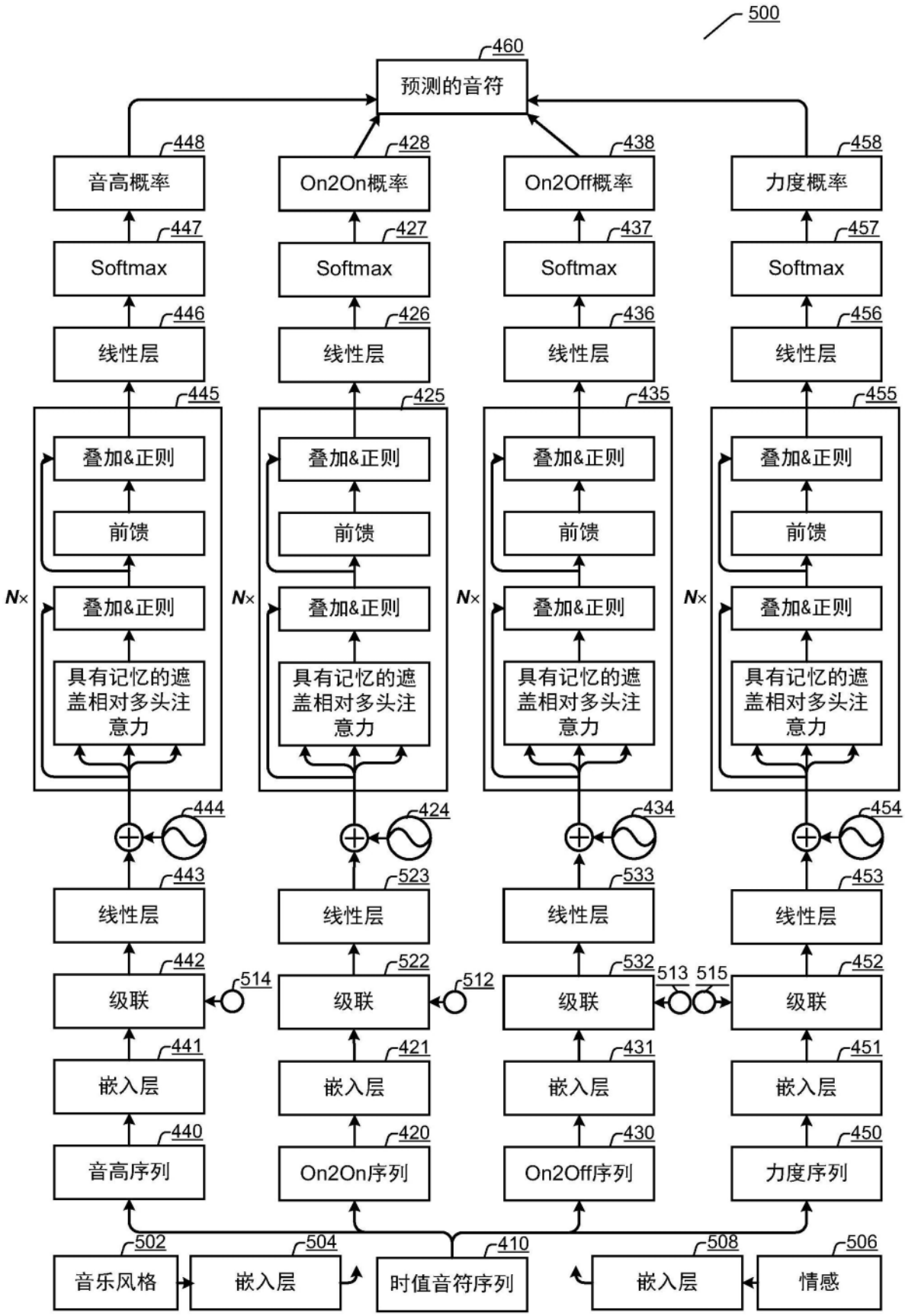

500
460
预测的音符
448
音高概率
428
On2On概率
438
On2Off概率
458
力度概率
447
Softmax
427
Softmax
437
Softmax
457
Softmax
446
线性层
426
线性层
436
线性层
456
线性层
445
425
435
455
叠加&正则
前馈
叠加&正则
具有记忆的遮盖相对多头注意力
N×
444
424
434
454
443
线性层
523
线性层
533
线性层
453
线性层
442
级联
514
522
级联
512
532
级联
513
515
452
级联
441
嵌入层
421
嵌入层
431
嵌入层
451
嵌入层
440
音高序列
420
On2On序列
430
On2Off序列
450
力度序列
502
音乐风格
504
嵌入层
410
时值音符序列
508
嵌入层
506
情感

图5

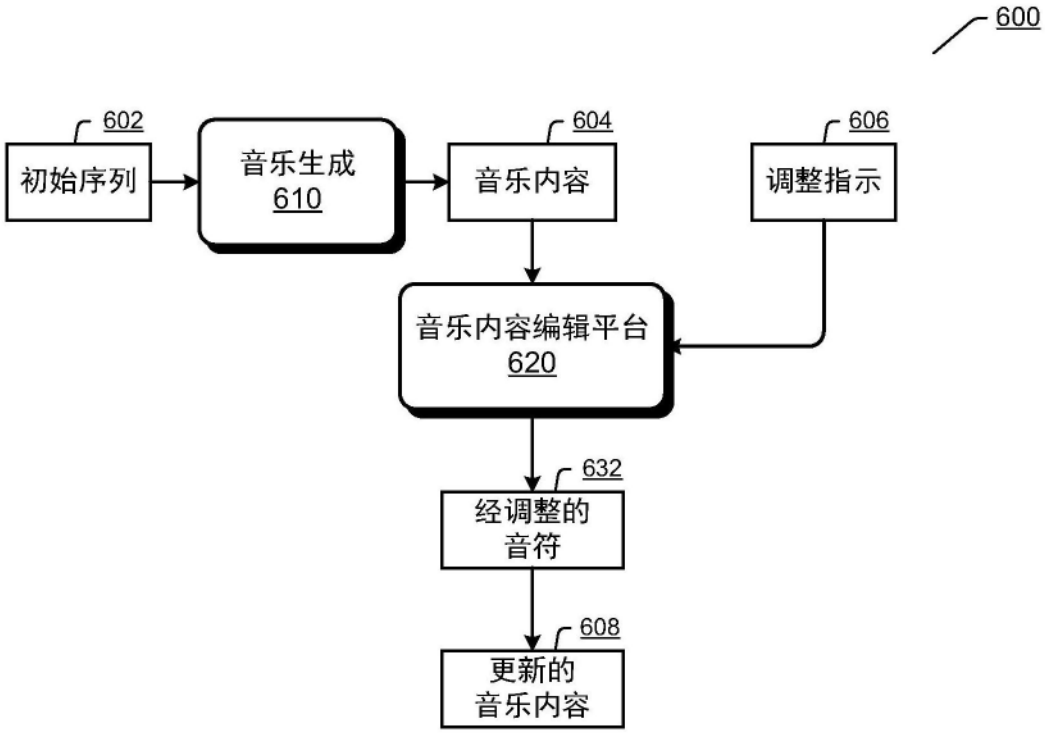

600
602
初始序列
音乐生成
610
604
音乐内容
606
调整指示
音乐内容编辑平台
620
632
经调整的
音符
608
更新的
音乐内容

图6

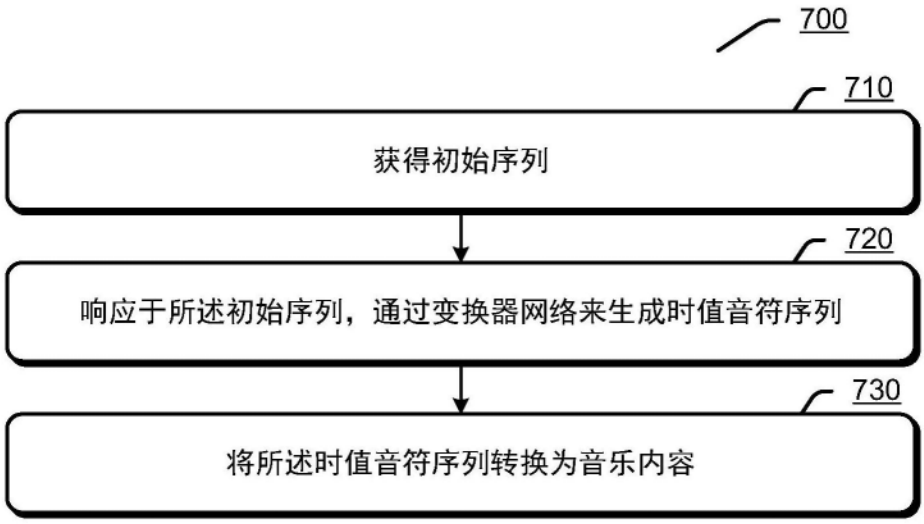

700
710
获得初始序列
720
响应于所述初始序列，通过变换器网络来生成时值音符序列
730
将所述时值音符序列转换为音乐内容

图7

图8

图9