

解码方法、装置和计算机可读存储介质

相关申请的交叉引用

本申请是以 CN 申请号为 202110229441.9, 申请日为 2021 年 3 月 2 日的申请为基础, 并主张其优先权, 该 CN 申请的公开内容在此作为整体引入本申请中。

技术领域

本公开涉及计算机技术领域, 特别涉及一种解码方法、装置和计算机可读存储介质。

背景技术

随着人工智能的高速发展, 人工智能客服机器人的应用越来越广泛, 涉及语音识别技术, 而语音识别又依赖说实时的音频流的输入, 作为语音识别的前置条件。通常在人工智能客服领域, 需要对用户同机器人讲的话进行识别, 用户讲的话会作为音频流实时传入系统中, 因此在对音频流的实时解码成为一个要解决的问题。

对音频流进行实时解码需要获取音频的格式、参数等, 这些信息一般包含在头部信息中。

发明内容

根据本公开的一些实施例, 提供一种解码方法, 包括: 对接收的数据流的一个或多个流片段进行缓存, 其中, 数据流包括音频流; 对缓存的流片段进行解析, 直至解析得到头部信息; 将头部信息进行保存; 根据头部信息对接收的各个流片段中音频流的流片段进行解码, 直至完成对音频流的解码。

在一些实施例中, 对缓存的流片段进行解析, 直至解析得到头部信息包括: 确定当前缓存的所有流片段的总的数据长度是否达到预设帧长度; 在当前缓存的所有流片段的总的数据长度达到预设帧长度的情况下, 对流片段中从起始数据开始到满足预设帧长度的数据进行解析; 确定是否成功解析得到头部信息; 在未成功解析得到头部信息的情况下, 将预设帧长度增加预设值, 更新预设帧长度; 重复执行上述步骤, 直至解析得到头部信息。

在一些实施例中, 对缓存的流片段进行解析, 直至解析得到头部信息还包括: 在

当前缓存的所有流片段的总的长度未达到预设帧长度的情况下，等待接收到下一个流片段进行缓存后，重新执行确定缓存的所有流片段的总的长度是否达到预设帧长度。

5 在一些实施例中，根据头部信息对接收的各个流片段中音频流的流片段进行解码包括：根据头部信息确定音频帧的长度；根据音频帧的长度，对接收到的各个流片段中音频流的流片段区分不同的音频帧进行解码。

10 在一些实施例中，根据音频帧的长度，对接收到的各个流片段中音频流的流片段区分不同的音频帧进行解码包括：根据音频帧的长度，按照数据封装格式的顺序对音频流的当前的流片段，进行划分，得到一个或多个完整的音频帧；对一个或多个完整的音频帧进行解码；确定音频流的当前的流片段的尾部数据是否属于不完整的音频帧；在音频流的当前的流片段的尾部数据属于不完整的音频帧的情况下，将不完整的音频帧缓存；等待接收到音频流的下一个流片段后，将下一个流片段与不完整的音频帧进行拼接，得到拼接后的流片段；将拼接后的流片段作为音频流的当前的流片段，重复执行上述步骤，直至完成对音频流的最后一个流片段的解码。

15 在一些实施例中，根据头部信息对接收的各个流片段中音频流的流片段进行解码，直至完成对音频流的解码包括：在根据头部信息对音频流的当前的流片段进行解码失败的情况下，对当前的流片段或者当前的流片段以及当前的流片段之后的流片段进行解析，直至解析得到新的头部信息；根据新的头部信息对接当前的流片段之后的流片段进行解码，直至完成对音频流的解码。

20 在一些实施例中，对缓存的流片段进行解析，直至解析得到头部信息包括：调用 **FFmpeg** 中的 **Open avformat** 方法对缓存的流片段进行解析，直至解析得到头部信息。

25 在一些实施例中，根据头部信息对接收的各个流片段中音频流的流片段进行解码包括：根据头部信息确定数据流是否包括音频流以外的其他数据流；在数据流包括音频流以外的其他数据流的情况下，将其他数据流与音频流分离；根据头部信息确定音频流的格式信息；根据音频流的格式信息将音频流的各个流片段转码为原始音频流；将原始音频流按照预设码率重新采样。

30 在一些实施例中，调用 **FFmpeg** 中的分离流 **Separate stream** 方法将其他数据流与音频流分离；调用 **FFmpeg** 中的解析格式 **Parse format** 方法根据头部信息确定音频流的格式信息，根据音频流的格式信息将音频流的各个流片段转码为原始音频流，并将原始音频流按照预设码率重新采样。

根据本公开的另一些实施例，提供一种解码装置，包括：缓存模块，用于对接收的数据流的一个或多个流片段进行缓存，其中，数据流包括音频流；头部信息解析模块，用于对缓存的流片段进行解析，直至解析得到头部信息；头部信息保存模块，用于将头部信息进行保存；解码模块，用于根据头部信息对接收的各个流片段中音频流的流片段进行解码，直至完成对音频流的解码。

根据本公开的又一些实施例，提供一种解码装置，包括：处理器；以及耦接至处理器的存储器，用于存储指令，指令被处理器执行时，使处理器执行如前述任意实施例的解码方法。

根据本公开的再一些实施例，提供一种非瞬时性计算机可读存储介质，其上存储有计算机程序，其中，该程序被处理器执行时实现前述任意实施例的解码方法。

通过以下参照附图对本公开的示例性实施例的详细描述，本公开的其它特征及其优点将会变得清楚。

附图说明

此处所说明的附图用来提供对本公开的进一步理解，构成本申请的一部分，本公开的示意性实施例及其说明被配置为解释本公开，并不构成对本公开的不当限定。

图 1 示出本公开的一些实施例的解码方法的流程示意图。

图 2 示出本公开的一些实施例的音频流的结构示意图。

图 3 示出本公开的另一些实施例的解码方法的流程示意图。

图 4 示出本公开的一些实施例的解码装置的结构示意图。

图 5 示出本公开的另一些实施例的解码装置的结构示意图。

图 6 示出本公开的又一些实施例的解码装置的结构示意图。

具体实施方式

下面将结合本公开实施例中的附图，对本公开实施例中的技术方案进行清楚、完整地描述，显然，所描述的实施例仅仅是本公开一部分实施例，而不是全部的实施例。以下对至少一个示例性实施例的描述实际上仅仅是说明性的，决不作为对本公开及其应用或使用的任何限制。基于本公开中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都属于本公开保护的范围。

发明人发现：实际人工智能客服的电话场景中，音频需要进行流式处理，即将音

频文件分为音频流片段进行传输，此时第一个流片段或前几个流片段中会包含音频编码时产生的头部信息。之后的流片段中不包含头部信息，尤其在使用 FFmpeg 工具针对不同的流片段进行解码的过程中，会因为大部分流片段不包含头部信息无法解码而返回错误信息，无法满足实时音频流在人工智能客服场景中实时解码的需求。

5 本公开所要解决的一个技术问题是：如何实现对音频流的实时解码。

本公开提供一种解码方法可以用于人工智能客服场景中对音频流实时解码，下面结合图 1~图 3 进行描述。

图 1 为本公开解码方法一些实施例的流程图。如图 1 所示，该实施例的方法包括：步骤 S102~S108。

10 在步骤 S102 中，对接收的数据流的一个或多个流片段进行缓存。

数据流包括音频流，还可以包括音频流之外的其他数据流，例如，视频流等非音频数据流，音频流和其他数据流混合的情况下，后续步骤中需要进行不同流的分离，具体在之后的实施例中描述。数据流在传输过程中被划分为多个流片段，每个流片段可以被封装成数据包（Package）进行传输。解码装置（执行本公开解码方法的装置）

15 接收到数据包后，对数据包进行解析得到流片段，对流片段进行缓存。

可以基于 FFmpeg API 实现本公开的方案。首先可以初始化 avformat 和 avio context 两个模块（Init avformat/Init avio context），分别用来进行后续的头部信息解析以及音频流的读取，对流片段进行缓存可以调用 Buffer stream 方法。

在步骤 S104 中，对缓存的流片段进行解析，直至解析得到头部信息。

20 头部（Header）信息例如包括：音频流的格式信息和至少一个参数，至少一个参数例如包括：采样率、位深、通道数、压缩比等至少一项，不限于所举示例。由于流片段的划分是不确定的，有可能一个流片段中就包含完整的头部信息，也有可能一个流片段只包含部分头部信息，需要多个流片段才能得到完整的头部信息。在一些实施例中，每次缓存一个流片段之后，尝试对之前缓存的所有流片段进行解析，确定是否
25 成功解析得到头部信息，在未成功解析得到头部信息的情况下，继续对下一个流片段进行缓存，重复上述过程，直至成功解析得到头部信息。

在另一些实施例中，确定当前缓存的所有流片段的总的数据长度是否达到预设帧长度；在当前缓存的所有流片段的总的数据长度达到预设帧长度的情况下，对当前缓存的流片段中从起始数据开始到满足预设帧长度的数据进行解析；确定是否成功解析
30 得到头部信息；在未成功解析得到头部信息的情况下，将预设帧长度增加预设值，更

新预设帧长度；重复执行上述步骤，直至解析得到头部信息。

预设帧长度可以根据历史音频流中头部信息的长度统计得到的。每次缓存一个流片段之后可以判断当前缓存的所有流片段的总的长度是否达到预设帧长度。在当前缓存的所有流片段的总的长度未达到预设帧长度的情况下，等待接收到下一个流片段进行缓存后，重新执行确定缓存的所有流片段的总的长度是否达到预设帧长度的步骤。直至当前缓存的所有流片段的总的长度达到预设帧长度，则尝试对当前缓存的流片段中从起始数据开始到满足预设帧长度的数据进行解析。

例如，预设帧长度为 200 字节，则由最开始缓存的流片段的第一个字节开始至 200 字节的长度的数据，作为待解析数据，对待解析数据进行解析确定是否成功解析得到头部信息。如果成功解析得到头部信息，则停止头部信息的解析过程。如果解析头部信息失败，预设帧长度增加预设值，更新预设帧长度，例如，将 200 字节增加至 300 字节。之后由确定当前缓存的所有流片段的总的长度是否达到预设帧长度的步骤开始重新执行。

可以调用 FFmpeg 中的 **Open avformat** 方法对缓存的流片段进行解析，直至解析得到头部信息。上述方法中通过持续对缓存的流片段进行头部解析处理的尝试，避免了在头部信息被划分至不同流片段的情况下，无法成功解析的问题。通过预设帧长度的判断和修正，减少了解析次数，提高效率。

在步骤 S106 中，将头部信息进行保存。

在步骤 S108 中，根据头部信息对接收的各个流片段中音频流的流片段进行解码，直至完成对音频流的解码。

在数据流只包含音频流的情况下，利用头部信息直接对接收到的每个流片段进行解码。在数据流包含音频流和其他数据流（非音频数据流）的情况下，需要进行流的分离操作。在一些实施例，根据头部信息确定数据流是否包括音频流以外的其他数据流；在数据流包括音频流以外的其他数据流的情况下，将其他数据流与音频流分离。例如，调用 FFmpeg 中的 **Separate stream**（分离流）方法将其他数据流与音频流分离。

将接收的每个流片段中的音频流的流片段分离出来后，利用头部信息对音频流的流片段进行解码。在一些实施例，根据头部信息确定音频流的格式信息；根据音频流的格式信息将音频流的各个流片段转码为原始音频流；将原始音频流按照预设码率重新采样。重新采样的码率符合播放设备的码率，便于播放。例如，调用 FFmpeg 中的 **Parse format**（解析格式）方法根据头部信息确定音频流的格式信息，根据音频流

的格式信息将音频流的各个流片段转码为原始音频流，并将原始音频流按照预设码率重新采样。

在音频流只包含一个完整的音频文件的情况下，利用保存的头部信息可以实现对整个音频流的正确解码。在音频流包含多个完整的音频文件的情况下，不同音频文件的头部信息可能会不同，在解码过程中出现失败。针对该问题，在一些实施例中，在根据头部信息对音频流的当前的流片段进行解码失败的情况下，对当前的流片段或者当前的流片段以及当前的流片段之后的流片段进行解析，直至解析得到新的头部信息；根据新的头部信息对接当前的流片段之后的流片段进行解码，直至完成对音频流的解码。

解析获取新的头部信息的方法可以参考前述实施例中解析头部信息的方法。将新的头部信息进行保存，可以删除原来保存的头部信息，利用新的头部信息对之后接收的流片段进行解码，直至完成对音频流的解码。

上述实施例的方法首先对接收的数据流的一个或多个流片段进行缓存，对缓存的流片段持续进行解析，直至解析得到头部信息，将头部信息进行保存，利用头部信息对后续接收的各个流片段中音频流的流片段进行解码，直至完成对音频流的解码。上述实施例的方法可以实现音频流的实时解码，满足实时音频流在人工智能客服场景中实时解码的需求。

尤其针对使用 **FFmpeg** 工具实现音频解码的场景，上述实施例的方法通过音频流 **buffer** 缓存流片段，提取头部信息（包含音频流的格式信息和参数等）解析后进行保存，根据头部信息可以解析出音频流的格式信息和至少一个参数，通过音频流的格式信息可以获取到解码器类型，并在之后接收的音频流的流片段中，使用之前缓存的解码器类型链接对应的解码器引擎，并根据音频流的至少一个参数去对之后的流片段进行解码尝试，这种情况下，对于使用标准的音频编码器生成的音频流，可以做到实时解码，解决了 **FFmpeg** 工具由于大部分流片段中不包含头部信息而无法解码的问题。

音频流传输过程中，如果传输的流片段并非按照音频帧长度的整数倍进行切分，可能存在不完整音频帧的问题。如图 2 所示，音频流的流片段 1 包含音频帧（**Frame**）1、音频帧 2 以及音频帧 3 的一部分，而流片段 2 包含音频帧 3 的另一部分，此时使用解码器根据头部信息对流片段 1 和 2 解码时会报错。针对上述问题，本公开还提供一种解决方法。在一些实施例中，根据头部信息确定音频帧的长度；根据音频帧的长度，对接收到的各个流片段中音频流的流片段区分不同的音频帧进行解码。可以根据

头部信息包含的参数确定音频帧的长度，例如，根据采样率、位深、通道数等确定音频帧的长度，可以参考现有技术，具体不再赘述。

进一步，如图 3 所示，根据头部信息对接收的各个流片段中音频流的流片段进行解码包括：步骤 S302~S316。

5 在步骤 S302 中，根据头部信息确定音频帧的长度。

在步骤 S304 中，如果头部信息所在的流片段还包含音频数据，则将该流片段作为音频流的当前的流片段。

在步骤 S306 中，对当前的流片段，根据音频帧的长度，按照数据封装格式的顺序划分音频帧，即根据音频帧的长度，按照数据封装格式的顺序对当前的流片段进行
10 划分，得到一个或多个完整的音频帧。

例如，数据在流片段中按照从左到右或者从前到后的顺序排列。如图 2 所示，流片段 1 划分音频帧后，尾部数据属于不完整的音频帧 3。

在步骤 S308 中，对所述一个或多个完整的音频帧进行解码。

在步骤 S310 中，确定当前的流片段是否为最后一个流片段，如果是，则停止，
15 否则执行步骤 S312。

在步骤 S312 中，确定音频流的当前的流片段的尾部数据是否属于不完整的音频帧。如果是，则执行步骤 S314，否则执行步骤 S313。

在步骤 S313 中，等待接收到音频流的下一个流片段后，将下一个流片段作为当前的流片段，返回步骤 S306 重新开始执行。

20 在步骤 S314 中，将不完整的音频帧缓存。

在步骤 S316 中，等待接收到音频流的下一个流片段后，将下一个流片段与不完整的音频帧进行拼接，得到拼接后的流片段，作为当前的流片段，并返回步骤 S306 重新开始执行

如图 2 所示，将流片段 2 与流片段 1 中音频帧 3 的前半部分进行拼接，组成完整的
25 的帧。

上述实施例的方法，考虑对不完整的帧信息缓存到下个流片段接收后再进行拼接处理，解决了流片段中包含不完整的音频帧的情况下，无法正确解码的问题。

本公开还提供一种解码装置，下面结合 4 进行描述。

图 4 为本公开解码装置的一些实施例的结构图。如图 4 所示，该实施例的装置
30 包括：缓存模块 410，头部信息解析模块 420，头部信息保存模块 430，解码模块 440。

缓存模块 410 用于对接收的数据流的流片段进行缓存,其中,数据流包括音频流。

头部信息解析模块 420 用于对缓存的一个或多个流片段进行解析,直至解析得到头部信息。

5 在一些实施例中,头部信息解析模块 420 用于确定当前缓存的所有流片段的总的
数据长度是否达到预设帧长度;在当前缓存的所有流片段的总的的数据长度达到预设帧
长度的情况下,对流片段中从起始数据开始到满足预设帧长度的数据进行解析;确定
是否成功解析得到头部信息;在未成功解析得到头部信息的情况下,将预设帧长度增
加预设值,更新预设帧长度;重复执行上述步骤,直至解析得到头部信息。

10 在一些实施例中,头部信息解析模块 420 用于在当前缓存的所有流片段的总的数
据长度未达到预设帧长度的情况下,等待接收到下一个流片段进行缓存后,重新执行
确定缓存的所有流片段的总的的数据长度是否达到预设帧长度。

在一些实施例中,头部信息解析模块 420 用于调用 FFmpeg 中的 **Open_avformat**
方法对缓存的流片段进行解析,直至解析得到头部信息。

头部信息保存模块 430 用于将头部信息进行保存。

15 解码模块 440 用于根据头部信息对接收的各个流片段中音频流的流片段进行解码,
直至完成对音频流的解码。

在一些实施例中,解码模块 440 用于根据头部信息确定音频帧的长度;根据音频
帧的长度,对接收到的各个流片段中音频流的流片段区分不同的音频帧进行解码。

20 在一些实施例中,解码模块 440 用于根据音频帧的长度,按照数据封装格式的顺
序对音频流的当前的流片段,进行划分,得到一个或多个完整的音频帧;对一个或多
个完整的音频帧进行解码;确定音频流的当前的流片段的尾部数据是否属于不完整的
音频帧;在音频流的当前的流片段的尾部数据属于不完整的音频帧的情况下,将不完
整的音频帧缓存;等待接收到音频流的下一个流片段后,将下一个流片段与不完整的
音频帧进行拼接,得到拼接后的流片段;将拼接后的流片段作为音频流的当前的流片
25 段,重复执行上述步骤,直至完成对音频流的最后一个流片段的解码。

在一些实施例中,解码模块 440 用于在根据头部信息对音频流的当前的流片段进
行解码失败的情况下,对当前的流片段或者当前的流片段以及当前的流片段之后的流
片段进行解析,直至解析得到新的头部信息;根据新的头部信息对接当前的流片段之
后的流片段进行解码,直至完成对音频流的解码。

30 在一些实施例中,解码模块 440 用于根据头部信息确定数据流是否包括音频流以

外的其他数据流；在数据流包括音频流以外的其他数据流的情况下，将其他数据流与音频流分离；根据头部信息确定音频流的格式信息；根据音频流的格式信息将音频流的各个流片段转码为原始音频流；将原始音频流按照预设码率重新采样。

5 在一些实施例中，解码模块 440 用于调用 FFmpeg 中的分离流 **Separate stream** 方法将其他数据流与音频流分离；调用 FFmpeg 中的解析格式 **Parse format** 方法根据头部信息确定音频流的格式信息，根据音频流的格式信息将音频流的各个流片段转码为原始音频流，并将原始音频流按照预设码率重新采样。

本公开的实施例中的解码装置可各由各种计算设备或计算机系统来实现，下面结合图 5 以及图 6 进行描述。

10 图 5 为本公开解码装置的一些实施例的结构图。如图 5 所示，该实施例的装置 50 包括：存储器 510 以及耦接至该存储器 510 的处理器 520，处理器 520 被配置为基于存储在存储器 510 中的指令，执行本公开中任意一些实施例中的解码方法。

15 其中，存储器 510 例如可以包括系统存储器、固定非易失性存储介质等。系统存储器例如存储有操作系统、应用程序、引导装载程序（**Boot Loader**）、数据库以及其他程序等。

图 6 为本公开解码装置的另一一些实施例的结构图。如图 6 所示，该实施例的装置 60 包括：存储器 610 以及处理器 620，分别与存储器 510 以及处理器 520 类似。还可以包括输入输出接口 630、网络接口 640、存储接口 650 等。这些接口 630，640，650 以及存储器 610 和处理器 620 之间例如可以通过总线 660 连接。其中，输入输出接口 20 630 为显示器、鼠标、键盘、触摸屏等输入输出设备提供连接接口。网络接口 640 为各种联网设备提供连接接口，例如可以连接到数据库服务器或者云端存储服务器等。存储接口 650 为 SD 卡、U 盘等外置存储设备提供连接接口。

本领域内的技术人员应当明白，本公开的实施例可提供为方法、系统、或计算机程序产品。因此，本公开可采用完全硬件实施例、完全软件实施例、或结合软件和硬件 25 件方面的实施例的形式。而且，本公开可采用在一个或多个其中包含有计算机可用程序代码的计算机可用非瞬时性存储介质（包括但不限于磁盘存储器、**CD-ROM**、光学存储器等）上实施的计算机程序产品的形式。

本公开是参照根据本公开实施例的方法、设备（系统）、和计算机程序产品的流程图和 / 或方框图来描述的。应理解为可由计算机程序指令实现流程图和 / 或方框图 30 中的每一流程和 / 或方框、以及流程图和 / 或方框图中的流程和 / 或方框的结合。可

提供这些计算机程序指令到通用计算机、专用计算机、嵌入式处理机或其他可编程数据处理设备的处理器以产生一个机器，使得通过计算机或其他可编程数据处理设备的处理器执行的指令产生被配置为实现在流程图一个流程或多个流程和 / 或方框图一个方框或多个方框中指定的功能的装置。

- 5 这些计算机程序指令也可存储在能引导计算机或其他可编程数据处理设备以特定方式工作的计算机可读存储器中，使得存储在该计算机可读存储器中的指令产生包括指令装置的制造品，该指令装置实现在流程图一个流程或多个流程和 / 或方框图一个方框或多个方框中指定的功能。

- 10 这些计算机程序指令也可装载到计算机或其他可编程数据处理设备上，使得在计算机或其他可编程设备上执行一系列操作步骤以产生计算机实现的处理，从而在计算机或其他可编程设备上执行的指令提供被配置为实现在流程图一个流程或多个流程和 / 或方框图一个方框或多个方框中指定的功能的步骤。

以上所述仅为本公开的较佳实施例，并不用以限制本公开，凡在本公开的精神和原则之内，所作的任何修改、等同替换、改进等，均应包含在本公开的保护范围之内。

权 利 要 求

1. 一种解码方法，包括：

对接收的数据流的一个或多个流片段进行缓存，其中，所述数据流包括音频流；

对缓存的流片段进行解析，直至解析得到头部信息；

将所述头部信息进行保存；

根据所述头部信息对接收的各个流片段中所述音频流的流片段进行解码，直至完成对所述音频流的解码。

2. 根据权利要求 1 所述的解码方法，其中，所述对缓存的流片段进行解析，直至解析得到头部信息包括：

确定当前缓存的所有流片段的总的数据长度是否达到预设帧长度；

在当前缓存的所有流片段的总的数据长度达到所述预设帧长度的情况下，对所述流片段中从起始数据开始到满足所述预设帧长度的数据进行解析；

确定是否成功解析得到所述头部信息；

在未成功解析得到所述头部信息的情况下，将所述预设帧长度增加预设值，更新所述预设帧长度；

重复执行上述步骤，直至解析得到头部信息。

3. 根据权利要求 2 所述的解码方法，其中，所述对缓存的流片段进行解析，直至解析得到头部信息还包括：

在当前缓存的所有流片段的总的数据长度未达到预设帧长度的情况下，等待接收到下一个流片段进行缓存后，重新执行确定缓存的所有流片段的总的数据长度是否达到预设帧长度。

4. 根据权利要求 1 所述的解码方法，其中，所述根据所述头部信息对接收的各个流片段中所述音频流的流片段进行解码包括：

根据所述头部信息确定音频帧的长度；

根据所述音频帧的长度，对接收到的各个流片段中所述音频流的流片段区分不同的音频帧进行解码。

5. 根据权利要求 4 所述的解码方法, 其中, 所述根据所述音频帧的长度, 对接收到的各个流片段中所述音频流的流片段区分不同的音频帧进行解码包括:

根据所述音频帧的长度, 按照数据封装格式的顺序对所述音频流的当前的流片段, 进行划分, 得到一个或多个完整的音频帧;

对所述一个或多个完整的音频帧进行解码;

确定所述音频流的当前的流片段的尾部数据是否属于不完整的音频帧;

在所述音频流的当前的流片段的尾部数据属于不完整的音频帧的情况下, 将所述不完整的音频帧缓存;

等待接收到所述音频流的下一个流片段后, 将所述下一个流片段与所述不完整的音频帧进行拼接, 得到拼接后的流片段;

将拼接后的流片段作为所述音频流的当前的流片段, 重复执行上述步骤, 直至完成对所述音频流的最后一个流片段的解码。

6. 根据权利要求 1 所述的解码方法, 其中, 所述根据所述头部信息对接收的各个流片段中所述音频流的流片段进行解码, 直至完成对所述音频流的解码包括:

在根据所述头部信息对所述音频流的当前的流片段进行解码失败的情况下, 对当前的流片段或者当前的流片段以及当前的流片段之后的流片段进行解析, 直至解析得到新的头部信息;

根据所述新的头部信息对接当前的流片段之后的流片段进行解码, 直至完成对所述音频流的解码。

7. 根据权利要求 1 所述的解码方法, 其中, 所述对缓存的流片段进行解析, 直至解析得到头部信息包括:

调用 FFmpeg 中的 `Open avformat` 方法对缓存的流片段进行解析, 直至解析得到头部信息。

8. 根据权利要求 1 所述的解码方法, 其中, 所述根据所述头部信息对接收的各个流片段中所述音频流的流片段进行解码包括:

根据所述头部信息确定所述数据流是否包括所述音频流以外的其他数据流;

在所述数据流包括所述音频流以外的其他数据流的情况下，将所述其他数据流与所述音频流分离；

根据所述头部信息确定所述音频流的格式信息；

根据所述音频流的格式信息将所述音频流的各个流片段转码为原始音频流；

将所述原始音频流按照预设码率重新采样。

9. 根据权利要求 8 所述的解码方法，其中，

调用 FFmpeg 中的分离流 **Separate stream** 方法将所述其他数据流与所述音频流分离；

调用 FFmpeg 中的解析格式 **Parse format** 方法根据所述头部信息确定所述音频流的格式信息，根据所述音频流的格式信息将所述音频流的各个流片段转码为原始音频流，并将所述原始音频流按照预设码率重新采样。

10. 一种解码装置，包括：

缓存模块，用于对接收的数据流的一个或多个流片段进行缓存，其中，所述数据流包括音频流；

头部信息解析模块，用于对缓存的流片段进行解析，直至解析得到头部信息；

头部信息保存模块，用于将所述头部信息进行保存；

解码模块，用于根据所述头部信息对接收的各个流片段中所述音频流的流片段进行解码，直至完成对所述音频流的解码。

11. 一种解码装置，包括：

处理器；以及

耦接至所述处理器的存储器，用于存储指令，所述指令被所述处理器执行时，使所述处理器执行如权利要求 1-9 任一项所述的解码方法。

12. 一种非瞬时性计算机可读存储介质，其上存储有计算机程序，其中，该程序被处理器执行时实现权利要求 1-9 任一项所述方法的步骤。