文献翻译

摘要

神经语音合成模型最近展示了为文本到语音和压缩应用合成高质量语音的能力。这些新模型通常需要强大的 GPU 才能实现实 时操作，因此能够降低它们的复杂度将开辟许多新应用的道路。我们提出了 LPCNet，一种结合线性预测和循环神经网络的 WaveRNN 变体，以显著提高语音合成的效率。我们证明 LPCNet 在相同网络规模下可以实现比 WaveRNN 显著更高的质量，并且高质量 LPCNet 语音合成可以在复杂度低于 3GFLOPS 下实现。这使得在低功耗设备上部署神经合成应用变得更加容易，例如嵌入式系统和手机。索引词—神经音频合成，参数编码，WaveRNN

1.引言

神经语音合成算法最近使得合成高质量语音[1,2,3],和以极低比特率[4]编码高质量语音成为可能。这些算法的第一代，通常基于 WaveNet[5],等算法，在高端 GPU 上提供了所需的数十亿浮点运算每秒(GFLOPS)，在实时中取得了令人鼓舞的结果。我们希望在移动手机等终端用户设备上进行合成，这些设备没有强大的 GPU 且电池容量有限。近期工作[6,7]专注于寻找更高效的模型，以降低语音合成的复杂性。在这项工作中，我们继续这一方向，提供更多效率改进，并使在更慢的 CPU 上合成语音变得更容易，且对电池寿命的影响有限。低复杂度参数化合成模型，如低比特率声码器，已经存在很长时间[8,9],，但它们的质量一直受到严重限制。虽然它们通常使用线性预测高效地对语音的频谱包络（声道响应）进行建模，但没有类似的简单模型用于激励。尽管在建模激励信号方面取得了一些进展[10,11,12],，但将其保持为一种挑战。在这项工作中，我们提出了 LPCNet 模型，该模型将频谱包络建模的负担从神经合成网络中移除，使其大部分能力可以用于建模频谱平坦的激励。这使得能够在更少的神经元下匹配最先进神经合成系统的质量，显著降低了复杂度。从第 2 节中总结的 WaveRNN 算法出发，我们进行了改进，以降低模型复杂度，具体细节见第 3 节。在第 4 节，我们基于所提出的模型，在无说话人依赖的语音合成环境中评估了 LPCNet 的质量和复杂度。我们在第 5 节得出结论。

2.WAVERNN

[7]中提出的 WaveRNN 架构以先前的音频样本 st−1 以及调节参数 f 作为输入，并为输出样本生成离散概率分布 P(st)。尽管它被提出为一个 16 位模型（分为 8 位粗略精度和 8 位精细精度），但在本摘要中我们省略了粗略/精细分割，既为了清晰度，也因为我们在这项工作中没有使用它。WaveRNN 模型主要由一个门控循环单元（GRU）[13],和两个全连接层组成，最后以 softmax 激活结束。它被计算为

$$
\begin{aligned}
\mathbf{x}_t &= [s_{t-1}; \mathbf{f}] \\
\mathbf{u}_t &= \sigma\left(\mathbf{W}^{(u)}\mathbf{h}_{t-1} + \mathbf{U}^{(u)}\mathbf{x}_t\right) \\
\mathbf{r}_t &= \sigma\left(\mathbf{W}^{(r)}\mathbf{h}_{t-1} + \mathbf{U}^{(r)}\mathbf{x}_t\right) \\
\widetilde{\mathbf{h}}_t &= \tanh\left(\mathbf{r}_t \circ \left(\mathbf{W}^{(h)}\mathbf{h}_{t-1}\right) + \mathbf{U}^{(h)}\mathbf{x}_t\right) \qquad (1) \\
\mathbf{h}_t &= \mathbf{u}_t \circ \mathbf{h}_{t-1} + (1 - \mathbf{u}_t) \circ \widetilde{\mathbf{h}}_t \\
P\left(s_t\right) &= \operatorname{softmax}\left(\mathbf{W}_2 \operatorname{relu}\left(\mathbf{W}_1 \mathbf{h}_t\right)\right) ,
\end{aligned}
$$

其中 W(·)和 U(·)矩阵是 GRU 权重，$\sigma(x) = \frac{1}{1+e^{-x}}$是 sigmoid 函数，而◦表示逐元素向量乘法。在本篇论文中，为了清晰度，省略了偏差。合成的输出样本 st 是通过从概率分布 P(st)中采样获得的。为了降低复杂性，GRU 使用的矩阵可以变得稀疏，[7]建议使用 4x4 或 16x1 大小的非零块，以确保向量化仍然可能且高效。

3.LPCNet

本节介绍了 LPCNet 模型，这是我们针对 WaveRNN 提出的改进方案。图 1 展示了其架构概述，更多细节将在本节中解释。它包含一个工作在 16kHz 的样本率网络，以及一个处理 10ms 帧（160 个样本）的帧率网络。在本工作中，我们将合成的输入限制为仅 20 个特征：18 个 Bark 尺度[14]倒谱系数，以及 2 个音调参数（周期、相关性）。对于低比特率编码应用，倒谱和音调参数将被量化[4],，而对于文本到语音，它们将使用另一个神经网络[1]从文本中计算得出。

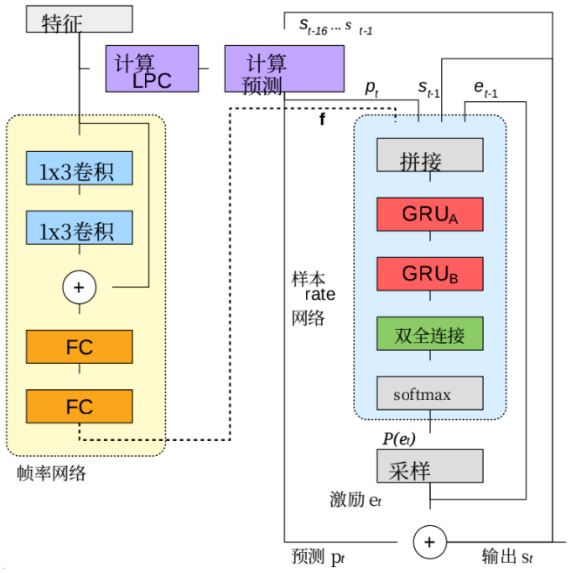

图 1.LPCNet 算法概述。网络（黄色）的左半部分每帧计算一次，其结果在整个帧内保持不变，供右侧（蓝色）的样本速率网络使用。计算预测模块根据先前样本和线性预测系数预测时间 t 处的样本。为简洁起见，省略了 μ-律和线性之间的转换。去加重滤波器应用于输出 st。

3.1 条件参数

作为帧率网络的一部分，20 个特征首先通过两个滤波器大小为 3（conv3x1）的卷积层，从而产生一个 5 帧（前两帧和后两帧）的感受野。两个卷积层的输出与残差连接相加，然后通过两个全连接层。帧率网络输出一个 128 维的条件向量 f，该向量随后被样本率网络使用。向量 f 在每一帧的持续时间内保持不变。

3.2 预加重和量化

一些合成模型，如 WaveNet[5]，使用 8 位 μ-律量化[15]将可能的输出样本值数量减少到仅 256 个。由于语音信号的能量往往主要集中在中低频段，μ-律白量化噪声在高频段通常可闻，特别是对于频谱倾斜度更高的 16kHz 信号。为了避免这个问题，一些方法将输出扩展到 16 位[7]。相反，我们建议对训练数据简单地应用一阶预加重滤波器 $E(z) = 1 - \alpha z^{-1}$，α=0.85 可获得良好结果。然后可以使用逆（去加重）滤波器对合成输出进行滤波:

$$D(z) = \frac{1}{1 - \alpha z^{-1}}\ , \tag{2}$$

有效塑造噪声，使其在奈奎斯特速率下的功率降低 16dB。这显著降低了感知噪声（参见第 4.3 节），并使 8 位 μ-law 输出适用于高质量合成。

3.3 线性预测

许多神经语音合成方法中的神经网络[4,5,6,7,16]需要建模整个语音产生过程，包括声门脉冲、噪声激励以及声道响应。尽管其中一些确实非常难以建模，但我们知道声道响应可以通过简单的全极点线性滤波器[17]合理地表示。令 st 为时间 t 的信号，其基于先前样本的线性预测是

$$p_t = \sum_{k=1}^{M} a_k s_{t-k}\ , \tag{3}$$

其中，a_k 是当前帧的 M^{th} 阶线性预测系数（LPC）。预测系数 a_k 的计算方法是：首先将 18 频带的梅尔频率倒谱转换为线性频率功率谱密度（PSD）。然后将 PSD 转换为自相关，使用逆 FFT。从自相关中，使用 Levinson-Durbin 算法计算预测器。从倒谱计算预测器可以确保（在语音编码上下文中）不需要传输额外信息，或（在文本到语音上下文中）不需要合成。尽管这种方式的 LPC 分析不如在输入信号上计算的准确（由于倒谱的分辨率较低），但输出受影响很小，因为网络能够学习进行补偿。这比开环滤波方法[12]是一个优势。

作为使用线性预测器帮助神经网络的明显扩展，我们也可以让网络直接预测激励（预测残差），而不是样本值。这不仅使网络的任务稍微容易一些，而且由于激励的幅度通常比预加重信号小，还稍微减少了 μ-律量化噪声。网络将先前采样的激励 e_{t-1}、过去信号 s_{t-1} 和当前预测 p_t 作为输入。我们仍然包括 s_{t-1} 和 p_t，因为我们发现仅基于 e_{t-1}

的开环合成会产生低质量的语音（见第 3.8 节）

3.4 输出层

为了更容易地计算输出概率，同时又不显著增加前一层的大小，我们将两个全连接层结合，使用逐元素加权求和。我们将这个层称为双全连接（或 DualFC），其定义为

$$\mathrm{dual\_fc}(\mathbf{x}) = \mathbf{a}_1 \circ \tanh\left(\mathbf{W}_1\mathbf{x}\right) + \mathbf{a}_2 \circ \tanh\left(\mathbf{W}_2\mathbf{x}\right) \ , \qquad (4)$$

其中 W1 和 W2 是权重矩阵，而 a1 和 a2 是加权向量。虽然对于所提出的方案能否工作并非严格必要，但我们发现，在复杂度相当的情况下，与常规全连接层相比，DualFC 层能略微提高质量。DualFC 层背后的直觉是，判断一个值是否落在某个特定范围内（在本例中为 μ-law 量化区间）需要进行两次比较，而每个全连接的 tanh 层实现的相当于一次比较。可视化训练好的网络中的权重支持这一直觉。DualFC 层的输出与 softmax 激活函数一起使用，用于计算每个可能的激励值 e_t 对应的概率 $P(e_t)$。

3.5 稀疏矩阵

为了保持复杂度较低，我们使用稀疏矩阵来表示最大的 GRU（图 1 中的 GRUA）。我们不是允许通用的逐元素稀疏性——这会阻止高效的向量化——而是使用[7]中提出的块稀疏矩阵。训练从密集矩阵开始，具有最低幅度的块会逐步被强制置零，直到达到所需的稀疏度。我们发现 16x1 块能提供良好的精度，同时便于向量化乘积。除了非零块之外，我们还包含稀疏矩阵中的所有对角线项，因为它们最有可能是非零的。尽管它们在水平或垂直方向上不对齐，但对角线项仍然很容易向量化，因为它们会导致与向量操作数进行逐元素乘法。包含对角线项可以避免仅为了对角线上的单个元素而强制使用 16x1 非零块。

3.6 嵌入和代数简化

我们没有在将标量样本值输入网络之前将它们缩放到固定范围，而是利用 μ-律值的离散性来学习嵌入矩阵 E。嵌入将每个 μ-律级别映射到一个向量，本质上是在学习要应用于 μ-律值的非线性函数集。通过可视化训练网络的嵌入矩阵，我们已经能够确认嵌入学习到了——除其他之外——将 μ-律尺度转换为线性的函数。只要嵌入直接发送到 GRU，就可以通过预先计算嵌入矩阵与 GRU 的非循环权重（U(·)）的相应子矩阵的乘积来避免增加复杂性。设 U(u)的子矩阵 U(u,s)由应用于 st−1 输入样本嵌入的列组成，我们可以推导出一个新的嵌入矩阵 V(u,s)=U(u,s)E，它直接将样本 st−1 映射到更新门计算的非循环项。相同的转换适用于所有门（u、r、h）和所有嵌入输入（s、p、e），总共 9 个预先计算的 V(·,·)矩阵。这样，嵌入贡献可以简化为每个门、每个嵌入输入仅一个加法操作。由于每个样本每个嵌入矩阵仅使用一个条目，即使这些矩阵不适合缓存，它们的大小也不是问题。与嵌入类似，帧条件向量 f 的贡献也可以简化，因为它在整个帧内是恒定的。每帧一次，我们可以计算 $g^{(\cdot)}=U^{(\cdot)}f$，即 f 对每个 GRU 门贡献的值。上述简化实质上使主 GRU 所有非循环输入的计算成本可以忽略不计，因此公式(1)中的计算变为

$$
\begin{aligned}
\mathbf{u}_t =& \sigma\left(\mathbf{W}_u \mathbf{h}_t + \mathbf{v}_{s_{t-1}}^{(u,s)} + \mathbf{v}_{p_{t-1}}^{(u,p)} + \mathbf{v}_{e_{t-1}}^{(u,e)} + \mathbf{g}^{(u)}\right) \\
\mathbf{r}_t =& \sigma\left(\mathbf{W}_r \mathbf{h}_t + \mathbf{v}_{s_{t-1}}^{(r,s)} + \mathbf{v}_{p_{t-1}}^{(r,p)} + \mathbf{v}_{e_{t-1}}^{(r,e)} + \mathbf{g}^{(r)}\right) \\
\widetilde{\mathbf{h}}_t =& \tanh\left(\mathbf{r}_t \circ (\mathbf{W}_h \mathbf{h}_t) + \mathbf{v}_{s_{t-1}}^{(h,s)} + \mathbf{v}_{p_{t-1}}^{(h,p)} + \mathbf{v}_{e_{t-1}}^{(h,e)} + \mathbf{g}^{(h)}\right) \\
\mathbf{h}_t =& \mathbf{u}_t \circ \mathbf{h}_{t-1} + (1 - \mathbf{u}_t) \circ \widetilde{\mathbf{h}}_t \\
P(e_t) =& \operatorname{softmax}\left(\text{dual\_fc}\left(\text{GRU}_\text{B}\left(\mathbf{h}_t\right)\right)\right),
\end{aligned}
\tag{5}
$$

在 $v^{(\cdot,\cdot)}_i$ 向量中，它们是列向量 i 在相应 $V^{(\cdot,\cdot)}$矩阵中的查找，而 $\text{GRU}_\text{B}(\cdot)$是一个常规的、非稀疏的 GRU，用于替代(1)中的全连接层（带 ReLU 激活）。

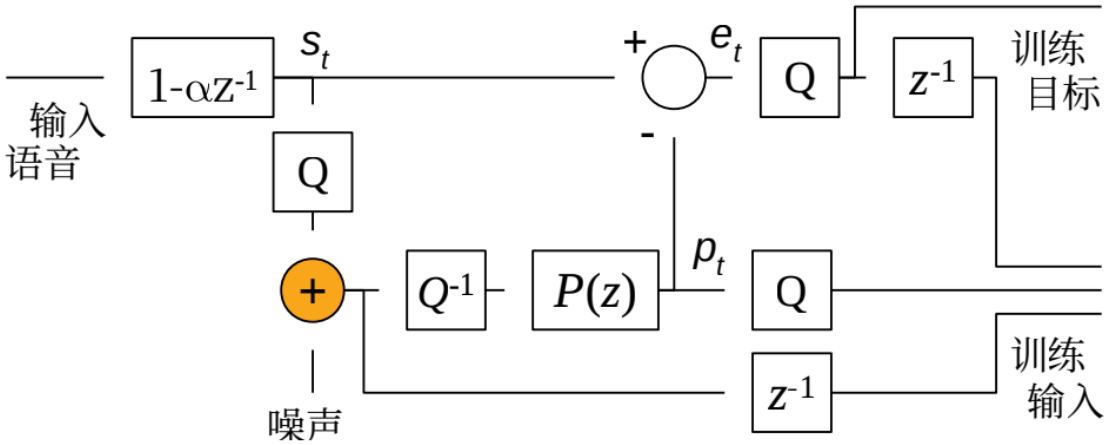

图 2。训练过程中的噪声注入，其中 Q 表示 μ-律量化，Q^{-1} 表示从 μ-律转换为线性。预测滤波器 $P(z) = \sum_{k=1}^{M} a_k z^{-k}$ 应用于噪声量化输入。激励值计算为干净未量化输入与预测值之间的差值。注意噪声是在 μ-律域中添加的。

3.7 从概率分布中采样

直接从输出分布中采样有时会导致过度的噪声。这在[6]中通过将 logits 乘以一个常数 c=2（用于语音声音）来解决，这相当于降低采样过程的“温度”。我们不是做一个二元的发声决定，而是设置

$$
c = 1 + \max\left(0, 1.5 g_p - 0.5\right), \tag{6}
$$

其中 g_p 是音调相关性($0<g_p<1$)。作为第二步，我们从分布中减去一个常数，以确保任何低于该常数阈值 T 的概率都变为零。这可以防止由低概率引起的脉冲噪声。修改后的概率分布变为

$$
P'(e_t) = \mathcal{R}\left(\max\left[\mathcal{R}\left(\left[P(e_t)\right]^c\right) - T, 0\right]\right), \tag{7}
$$

在 $\mathcal{R}(\cdot)$算子将分布重新归一化为 1，包括两个步骤之间以及最终结果上。我们发现 T=0.002 在减少脉冲噪声和保留语音自然度之间提供了良好的平衡。

3.8 训练噪声注入

在合成语音时，网络在不同于训练的条件下列行，因为生成的样本不同于训练时使用的样本（更不完美）。这种不匹配可能会放大并导致合成中产生过度失真。为了使网络对不匹配更鲁棒，我们在训练时向输入添加噪声，正如[6]所建议的。线性预测的使用使得噪声注入的细节尤为重要。在信号中注入噪声，但在干净的激励上训练网络时，我们发现系统产生的伪影类似于合成分析器时代，其中噪声的形状与合成滤波器 $\frac{1}{1-P(z)}$ 相同。相反，我们发现通过如图 2 所示添加噪声，网络有效地学习在信号域中最小化误差，因为尽管其输出是预测残差，但其中一个输入是用于计算该残差的相同预测。这类似于 CELP[18,19]的分析合成效果，并大大减少了合成语音中的伪影。为使噪声与信号幅度成正比，我们直接在 μ-律域中注入噪声。我们将其分布调整至训练数据中，从无噪声到在[−3,3]范围内均匀分布。

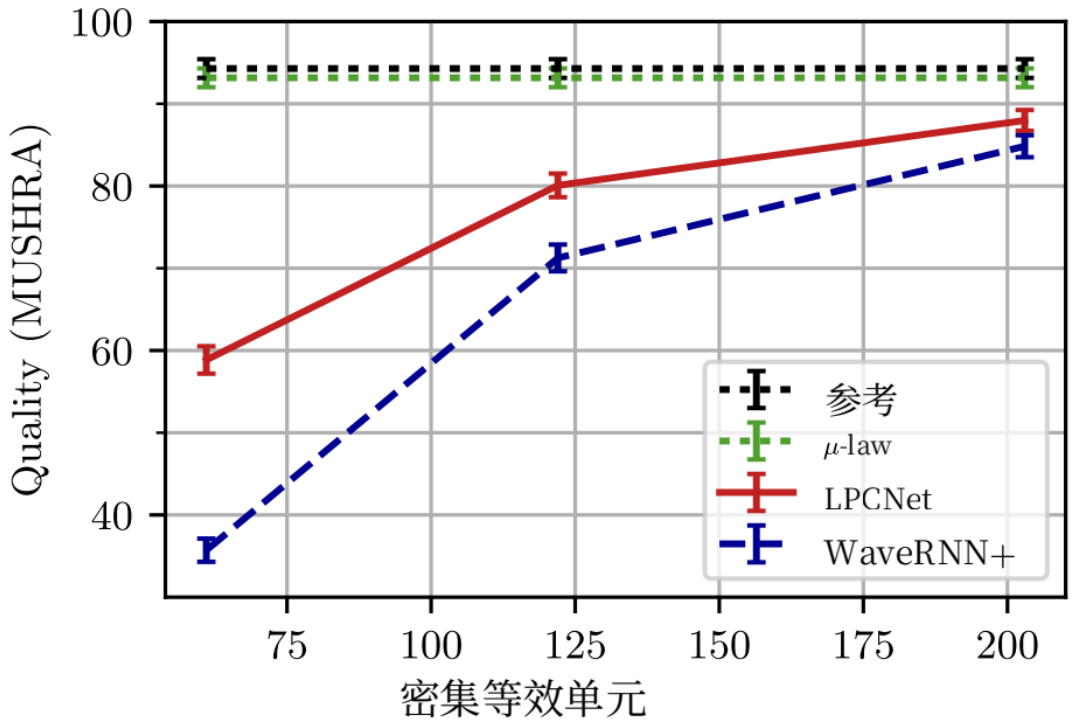

图 3。主观质量（MUSHRA）结果作为 GRUA 中密集等效单元数量的函数。

4.评估

4.1 复杂度

所提出的 LPCNet 模型的复杂度主要来自两个 GRU 以及双全连接层。它对应于每个样本产生的每个权重（一个加法，一个乘法）的两个操作，表示为

$$C = \left(3dN_A^2 + 3N_B\left(N_A + N_B\right) + 2N_BQ\right) \cdot 2F_s\,, \quad (8)$$

其中 N_A 和 N_B 是两个 GRU 的大小，d 是稀疏 GRU 的密度，Q 是 μ-律层数量，Fs 是采样速率。使用 N_A=384、N_B=16 和 Q=256 对宽带语音(Fs=16000)进行处理，并考虑被忽略项（偏差、条件网络、激活函数等）的约 0.5GFLOPS 复杂度，我们得到总复杂度约

为 2.8GFLOPS。可以在 AppleA8（iPhone6）的单个核心或 2.4GHzIntelBroadwell 核心的 20%上实现实时合成。作为对比，该说话人相关的 FFTNet 模型——声称比原始 WaveNet[5]算法复杂度更低——其复杂度约为 16GFLOPS[6]。原始 WaveRNN 的复杂度——作为说话人相关模型进行评估——并未明确说明，但我们对 WaveRNN[7]论文中提供的数据解读表明，稀疏、移动版本的复杂度约为 10GFLOPS。SampleRNN 的复杂度也未明确说明，但从论文中我们估计其复杂度约为 50GFLOPS（主要由于 1024 单元的 MLP 层）。

4.2 实验设置

尽管所提出的系统可以是说话人相关的，也可以是说话人无关的，但我们评估它在更具挑战性的说话人无关的上下文中。为了隔离编码器本身的质量，我们直接从录制的语音样本中计算特征。倒谱使用与[20]相同的频带布局，而音调估计器基于开环互相关搜索。训练数据仅包含来自 NTT 多语言语音数据库（电话测量，21 种语言）的 4 小时语音，其中我们排除了所有在测试中使用的说话人的样本。每个网络训练了 120 个 epoch（230k 次更新），批量大小为 64，每个序列由 15 个 10-ms 帧组成。训练是在 NvidiaGPU 上使用 Keras[1]/Tensorflow[2] 进行的，使用了 CuDNNGRU 实现和 AMSGrad[21]优化方法（Adam 变体），步长为$\alpha = \frac{\alpha_0}{1+\delta \cdot b}$，其中 α0=0.001、δ=5×10^{-5} 和 b 是批次数。

我们将 LPCNet 与 WaveRNN 的改进版本（记为 WaveRNN+）进行比较，后者包含了第 3 节中的所有改进，但除 LPC 部分外，即 WaveRNN+仅从 s_{t-1} 和条件参数预测样本 s_t。每个模型均使用一个主 GRU，其大小 NA 分别为 192、384 和 640 单元，且具有非零密度 d=0.1。这些模型中的非零权重数量与等效的密集 GRU（大小分别为 61、122 和 203）相同（这些 GRU 大小与[7]中的“等效”大小匹配）。在所有情况下，第二个 GRU 的大小为 NB=16。对于每个测试样本，我们从原始音频（真实值）计算输入特征，然后使用待测模型合成新的音频。我们还评估了预加重 μ-律量化的影响，将其作为可达到质量的上限。

4.3 质量评估

正如[4],中报告的，客观质量指标如 PESQ 和 POLQA 无法充分评估非波形、神经声码器。我们采用 MUSHRA 衍生方法[22],进行主观听音测试，其中 8 个语音片段（2 个男性和 2 个女性说话人）由 100 名参与者进行评估。图 3 的结果表明，在相同复杂度下，LPCNet 的质量显著优于 Wave-RNN+。或者，它表明在显著降低复杂度的情况下可以实现相同质量。结果还验证了我们的假设：预加重使得 μ-律量化噪声与合成伪影相比可以忽略不计。仅能计算单个 256 值分布也降低了复杂度，相比原始 16 位 WaveRNN 而言。生成的样本（来自 WaveRNN+和 LPCNet）中的主要可听伪影是由于基频谐波之间的噪声造成的粗糙感。一种可能的补救措施——我们尚未研究——包括使用后去噪技术，正如[6]中建议的那样。用于听音测试的样本子集可在 https://people.xiph.org/~jm/demo/lpcnet/获取。

5.结论

这项工作证明，通过将更新的神经合成技术与远为古老的线性预测技术相结合，可以提高说话人无关语音合成的效率。除了将线性预测与 WaveRNN 相结合的主要贡献外，我们还做出了其他贡献，例如信号值的嵌入、改进的采样，以及在 μ-律量化之前的预加重。我们相信，所提出的模型同样适用于文本到语音和低比特率语音编码。LPCNet 的未来工作包括研究长期（音调）预测是否可以作为一种进一步降低复杂性的方法。