arXiv:2303.01086v1 [cs.CL] 2 Mar 2023

LiteG2P：一种快速、轻量且高精度的字素到音素转换模型

Chunfeng Wang， *Peisong Huang*、*Yuxiang Zou*、*Haoyu Zhang*、*Shichao Liu*、*Xiang Yin*、*Zejun Ma*

字节跳动人工智能实验室，中国

摘要

作为自动语音识别（ASR）的关键组件和文本到语音（TTS）的前端，字素到音素（G2P）承担着将字母转换为对应发音的任务。现有方法要么速度较慢，要么性能较差，且应用场景有限，尤其是在端侧推理过程中。本文结合专家知识与基于连接主义时序分类（CTC）的神经网络各自的优势，提出一种名为 LiteG2P 的新方法，该方法快速、轻量，理论上可实现并行处理。得益于精心设计的引导机制，LiteG2P 既可应用于云端，也可在设备端运行。CMU数据集上的实验结果表明，所提方法的性能优于当前最先进的基于CTC的方法，参数量减少至十分之一；其性能甚至可与当前最先进的基于 Transformer 的序列到序列模型相媲美，同时参数量更少、计算量减少至三十三分之一。

索引词— 字素到音素、专家知识、连接时序分类、并行、文本到语音、自动语音识别、LiteG2P

1. 引言

字素到音素（G2P）旨在将单词字母转换为其发音，广泛应用于语音合成 [1, 2, 3] 和语音识别 [4, 5]等语音任务。随着语音应用场景的快速发展，尤其是文本到语音领域，对G2P任务的速度和性能都提出了越来越高的要求。

G2P任务的研究已有数十年的发展历程。在传统的系列方法中， [6] 提出了最早的基于规则的模型，而后来的方法[7, 8]则提出了联合n元语法模型以及加权有限状态转换器（WFST） [9, 10]。

近年来，随着深度神经模型的发展，越来越多的研究人员开始关注端到端解决方案。G2P 任务的关键问题是找到单词字母与音素之间的对齐。基于这一认识，许多研究人员将 G2P 任务视为神经机器翻译任务，从而能够利用现有的序列到序列模型； [11] 提出了一系列带注意力机制的编码器-解码器模型，其编码器和解码器由卷积层或双向长短期记忆网络层组成。[12],将基于 Transformer 的序列到序列模型引入 G2P，结果表明该模型在准确率与模型规模之间取得了更好的平衡。然而，也有一些研究从另一个角度尝试解决这一问题： [13] 首次提出了一种基于连接时序分类（CTC）[14] 的 G2P 方法。CTC 广泛应用于光学字符识别（ OCR ）[15] 和语音识别[4], 领域，这些领域要求输出长度小于或等于输入长度。针对 G2P 任务， [13] 提出了一种延迟策略，使输入长度与输出长度相等。

传统方法[6, 7, 8, 9, 10]性能较差，且需要丰富的专家经验。基于注意力的序列到序列[11, 12] 模型在解码阶段通常采用自回归方式，这限制了计算效率，并存在停止标记坍塌的风险。现有的基于连接时序分类的方法 [13] 需要通过输出长度来实现延迟策略，而且模型过于庞大，限制了应用场景。基于上述分析，我们认为，一种快速、紧凑且准确率高的G2P模型已呼之欲出。据我们所知，我们是首个提出能够同时应用于云端和设备端、且性能达到最先进水平的G2P模型的团队。

我们将所提出的方法称为 LiteG2P。LiteG2P 有以下三项贡献：

1. LiteG2P 是首个将少量专家知识融入数据驱动模型的端到端字素到音素模型，并使用连接时序分类损失函数来寻找对齐。

2. LiteG2P 支持并行处理，轻量且速度比最先进的方法快 30 倍以上，因此易于部署在云端和设备上。

3. LiteG2P 的性能可与最先进的基于 Transformer 的序列到序列模型相媲美。

2. LITEG2P

尽管基于序列到序列的模型在字素到音素任务中表现良好，但我们认为它们在该场景下存在冗余。与神经机器翻译任务不同，字素到音素具有许多固有且富有启发性的特征，例如单调性，以及字母与音素之间有限的映射集合。此外，我们发现，在我们的

实验中，序列到序列模型的性能对规模较为敏感。受上述所有因素的启发，我们决定为字素到音素任务设计一种新的端到端架构。字素到音素是一种序列转换任务，其中输入正字法序列 $W = [g1, g2, ..., gn]$ 通过映射函数 f转换为输出音素序列 $P = [p1, p2, ..., pt]$；在本研究中，该映射函数是一个神经网络。LiteG2P的整体架构如图1所示。

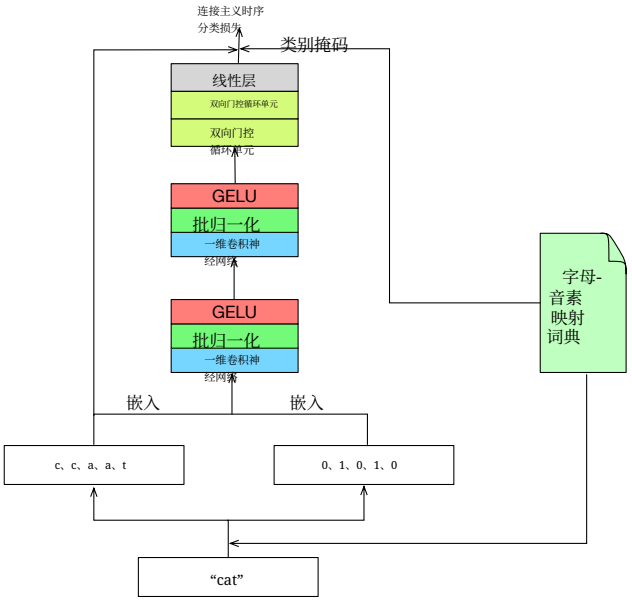


图 1。LiteG2P 的架构

由于序列到序列模型中没有注意力机制的帮助，我们尝试使用 CTC [14] 损失函数来对单词字母与音素之间的对齐进行建模。该函数将网络输出解释为在给定输入数据的条件下，所有可能输出标签序列上的概率分布。CTC目标函数直接最大化正确标注的概率。此外，我们将一个简单的附加词典作为专家知识，见表 1。从左到右，各列分别表示单词字母、对应的最大音素映射长度，以及美式英语中可能映射到的音素。与 [13], 不同，为了确保输出长度小于或等于输入长度，我们在数据预处理时将每个字母扩展到对应的最大音素映射长度。此外，我们发现每个字母只能映射到有限的音素集合，这可以缩小模型的搜索空间，并使模型更好地收敛。因此，我们利用可能映射的音素信息作为模型输出中的掩码，以帮助模型学习。

将单词字母输入模型之前，会先对单词进行处理，得到图 1 所示的两部分特征：第一部分是扩展字母，其扩展长度由词典确定。为了区分重复的字母，还会考虑局部位置嵌入；

表 1. 英语字母与音素映射词典

字母	最大映射长度	可能的音素
a	2	AA0,AA1,AA2, ...
b	1	B,P
c	2	K,CH,S,SH,T
d	1	D,T,JH
e	2	IH0,IH1,IH2,...
...	...	...
x	2	K,S,G,Z,...
y	1	IY0,IY1,IY2,...
z	2	Z,T,S,...

在我们的实验中，我们发现这能带来轻微的性能提升。处理完嵌入后，将这两部分特征拼接起来，作为最终的输入特征。具体而言，局部位置嵌入按如下方式归一化：

$$pos^{i}_{j_{norm}} = \frac{j - len_i}{max(len_i - 1, 1)} \tag{1}$$

其中，i、j、len_i 分别表示原始字母索引、从 1 开始的扩展局部字母索引，以及原始第 i 个字母的扩展长度。

参考光学字符识别 [15] 和自动语音识别 [4], 中最流行的基于连接时序分类的模型，我们设计了 LiteG2P 的模型架构，其中包含一维卷积神经网络（卷积神经网络）和双向门控循环单元（门控循环单元） [16]层。该模型包含两个卷积神经网络模块，每个模块由一维卷积神经网络层、批归一化层 [17] 和高斯误差线性单元（GELU） [18] 激活函数组成。两个卷积神经网络模块之后接有两个双向门控循环单元层。最后，从专家词典中生成一个掩码嵌入矩阵（如图2所示），并将其输入模型的最终输出，以帮助过滤不合理的音素对。矩阵的每个条目填入 0 或 1，表示字母与音素之间是否相关。掩码处理的伪代码如下：

$$output_mask = embedding_lookup(input, matrix_{mask}) \tag{2}$$

$$output = f(input, local_pos) * output_mask \tag{3}$$

$$output[output == 0] = INT_MIN \tag{4}$$

$$output = softmax(output) \tag{5}$$

其中，$input$、$local\ pos$、$matrix\ mask$、f 分别表示扩展字母、局部位置嵌入、掩码嵌入矩阵和模型函数。经过此掩码处理后，每个字母对应的不相关发音的概率都会降低。

	AA0	AA1	AA2	B	P	K	CH	S	SH	T	D	JH	IH0	IH1	IH2	…
a	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	…
b	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	…
c	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	…
d	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	…
e	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	…
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
x	0	0	0	0	0	1	0	1	1	0	0	0	1	0	0	…
Y	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	…
Z	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0	…

图2。由专家词典导出的掩码嵌入矩阵

3. 实验

3.1. 数据集

在接下来的实验中，我们使用 CMU [1] 发音词典来评估性能。该词典是 G2P 任务中公开且权威的数据集。我们按照 [13, 12, 19]的建议，将数据集划分为训练集和测试集，以便直接比较结果。移除多音节词并完成划分后，CMU数据集包含一个由 106,837 个单词组成的训练集和一个由 12,000 个单词组成的测试集。另有 2,670 个单词用作开发（验证）集。该数据集包含 28 个字素（小写字母符号，以及撇号和连字符）和 42 个音素，其中包括连接时序分类中使用的 $blank$ 标签（若将重音和音节边界也考虑在内，则有 84 个音素）。字母到音素映射词典由语言专家设计。

3.2. 软件和硬件详情

训练和推理阶段使用一张 Nvidia V100（32 GB）图形处理器卡，部署在配备 32 GB 内存的 8 核服务器上。训练和评估时，我们使用 Pytorch [2] 深度学习框架作为运行环境。

3.3. 实验配置

在 LiteG2P 中，两个输入特征首先分别嵌入为潜在嵌入，然后输入模型，且两种嵌入的维度均为 64。每个卷积神经网络模块中的一维卷积神经网络的通道数为 128，卷积核大小为 3，步幅为 1。对于双向门控循环单元模块，我们尝试了三组隐藏层大小不同的模型：128、192 和 256，分别对应 LiteG2P-small、LiteG2P-medium 和 LiteG2P-large。此外，每个门控循环单元层的丢弃率设为 0.1。在训练阶段，使用 Adam [**?**] 作为优化器，初始学习率设为 0.001，每 5 个训练轮次乘以 0.5，总共训练 50 个训练轮次。批量大小为 128。

[1]http://www.speech.cs.cmu.edu/cgi-bin/cmudict [2] https://pytorch.org/

表 2。LiteG2P的性能和模型大小

模型	小型	中型	大型
词错误率$_S$(%)	35.4	34.1	**33.6**
词错误率$_{SS}$(%)	36.4	35.1	**34.3**
词错误率(%)	26.5	24.3	**24.0**
size	**0.6M**	1.27M	2.25M

表 3。云设备上不同大小的 LiteG2P模型的耗时。

	Xeon 8260 中央处理器（2.4GHz）	V100 图形处理器
小型模型	**1.6毫秒/词**	**1.2毫秒/词**
中型模型	2.0毫秒/词	1.5毫秒/词
大型模型	3.1毫秒/词	1.6毫秒/词

表 4. LiteG2P-small 模型在移动设备上的耗时。

设备类型	运行时间
A53（1.4GHz）	6.38毫秒/词
A53（2.0GHz）	5.24毫秒/词
A73（1.8GHz）	3.30毫秒/词
A73（2.2GHz）	3.32毫秒/词
A78（2.8GHz）	**0.71毫秒/词**

4. 结果讨论

词错误率（WER）用作评估指标，表示预测的音素序列与参考发音不完全匹配的词所占的百分比，即词错误数除以参考文本中唯一词的总数。对于

表 5。CMU数据集上不同方法的性能、规模和速度比较

方法	词错误率(%)	size	运行时间（中央处理器）	运行时间（图形处理器）
编码器卷积神经网络、解码器双向长短期记忆网络（模型 5）[11]	25.13	14.5M	N/A	N/A
端到端卷积神经网络（模型4）[11]	29.74	7.62M	N/A	N/A
带注意力机制的编码器-解码器LSTM（模型1）[11]	28.44	12.7M	N/A	N/A
Transformer 3x3 [12]	**23.9**	1.49M	66毫秒/词	2.8毫秒/词
DBLSTM-CTC 128 个单元 [13]	27.9	3M	12毫秒/词	N/A
DBLSTM-CTC 512 个单元 [13]	25.8	11M	64毫秒/词	N/A
LiteG2P-medium	24.3	**1.27M**	**2.0毫秒/词**	**1.5毫秒/词**

由于我们的文本转语音系统使用了重音信息和音节边界，在后续实验中，除了词错误率指标外，我们还考虑了 WER_S 和 WER_{SS} 评估指标。WER_S 表示计算时考虑重音但不考虑音节边界的 WER，而 WER_{SS} 则表示同时考虑重音和音节边界的 WER。

在表 2 中，我们首先比较了 LiteG2P模型三种配置的准确率和模型大小。由于考虑了更复杂的输出，WER_S 和 WER_{SS} 的表现往往不如 WER。可以看出，随着模型规模增大，模型性能也有所提升。LiteG2P-large模型的准确率最高，而 LiteG2P-小型模型在这些模型中体积最小。值得注意的是，这三种模型都非常紧凑，LiteG2P 的小型版本仅有 0.6M 个参数，因此很有希望应用于大多数资源极其有限的移动设备。

在分析了 LiteG2P 的模型大小和性能后，我们进一步对运行速度进行了实验，结果如表 3 和表 4 所示。在运行时间评估中，我们从词典中随机选取了 100 个单词，并计算这 100 个单词的平均预测时间。我们首先在云设备上开展实验：选用芯片频率为 2.4GHz 的 Xeon 8260 作为 CPU设备，选用 Nvidia V100 卡作为 GPU设备。如表 3 所示，在中央处理器和图形处理器设备上，随着模型规模增大，运行时间都会增加。然而，即使是 LiteG2P-large，在中央处理器上预测一个单词也只需 3.1ms，在图形处理器上仅需 1.6ms，速度确实非常快。我们还发现，与图形处理器相比，中央处理器的速度变化更加明显。我们推测可能有两个原因。其一，本实验没有使用批量预测，因此无法充分利用图形处理器的并行计算能力。其二，由于模型计算速度过快，存储交互时间的影响被放大了。鉴于云设备在计算方面具有令人振奋的优势，我们进一步在移动设备上开展了实验，结果如表 4 所示。我们选择 LiteG2P-small 作为移动模型，并在涵盖低端到高端芯片的 5 种芯片上进行了实验。可以看到，即使是在最低端的芯片上，该小型模型每预测一个单词也只需 6.38ms，确实非常快速，并且有望应用于许多离线场景。

如表 5 所示，我们进一步与其他最先进的端到端方法进行了全面比较。在分析结果之前，需要说明几点：首先，[12],中没有提供速度数据；这里我们使用内部实现的模型测试速度。其次，对于 [13], 中的 DBLSTM-CTC模型，未注明速度数据是在何种设备上测试的，因此这里暂且将其视为在 CPU设备上的结果，并作为速度上限。最后，综合考虑性能与速度之间的权衡，我们选择 LiteG2P-medium作为云端模型。可以看出，与性能最佳的 DBLSTM-CTC[13]模型相比，LiteG2P的词错误率（WER）低 1.5%，模型体积缩小至十分之一，速度提升 30 倍。其准确率甚至可与 Transformer模型 [12],相媲美；在未进行任何优化的情况下，LiteG2P在中央处理器上的速度快 33 倍，在图形处理器上也快 1.8 倍以上，同时模型体积更小。LiteG2P在所有端到端方法中实现了最佳的性能与速度权衡，这归功于数据驱动的模型学习与专家知识的有效融合。我们还简单研究了是否引入专家词典。对于单词 *cat*，不使用词典时预测的音素为 *"KAE1 T F"*，使用词典后结果修正为 *"K AE1 T"*，这表明专家知识能够过滤掉不可能的音素。

5. 结论

我们研究了一种名为 LiteG2P 的新型端到端字素到音素架构。得益于引入额外的专家知识以及连接主义时序分类损失函数的帮助，该模型具有准确率高、可并行处理、轻量且计算速度快等优点。在 CMU数据集上进行的全面实验表明，LiteG2P 的表现优于最先进的基于CTC的方法，且参数量少 10 倍；与最先进的基于 Transformer 的序列到序列模型相比，它的参数量和计算量更少，性能甚至不相上下。该模型在移动应用领域也展现出巨大潜力。未来，我们将进一步探索 LiteG2P 在多语言方向上的应用，以及专家词典的自动生成。

6. 参考文献

[1]Yuxuan Wang、R.J. Skerry-Ryan、Daisy Stanton、Yonghui Wu、Ron J. Weiss、Navdeep Jaitly、Zongheng Yang、Ying Xiao、Zhifeng Chen、Samy Bengio、Quoc Le、Yannis Agiomyrgiannakis、Rob Clark 和 Rif A. Saurous，《Tacotron：迈向端到端语音合成》，载于 *Interspeech 2017* 会议论文集，2017 年，第 4006–4010 页。

[2] Yi Ren、Yangjun Ruan、Xu Tan、Tao Qin、Sheng Zhao、Zhou Zhao 和 Tie-Yan Liu，《FastSpeech：快速、稳健且可控的文本转语音》，神经信息处理系统进展，第 32 卷，2019年。[3] Yuxiang Zou、Shichao Liu、Xiang Yin、Haopeng Lin、Chunfeng Wang、Haoyu Zhang 和 Zejun Ma，《使用 ToBI 表示对神经语音合成进行细粒度韵律建模》，载于 *Interspeech*，2021年，第 3146–3150 页。[4] Dario Amodei、Sundaram Ananthanarayanan、Rishita Anubhai、Jingliang Bai、Eric Battenberg、Carl Case、Jared Casper、Bryan Catanzaro、Qiang Cheng、Guoliang Chen 等，《Deep Speech 2：英语和普通话端到端语音识别》，载于国际机器学习会议。PMLR，2016年，第 173–182 页。[5] Linhao Dong 和 Bo Xu，《CIF：用于端到端语音识别的连续积分发放》，载于*ICASSP 2020-2020 IEEE* 国际声学、语音与信号处理会议 *(ICASSP)*。IEEE，2020年，第 6079–6083 页。[6] Alan W Black、Kevin Lenzo 和 Vincent Pagel，《构建通用字母到语音规则的问题》，载于第三届 *ESCA/COCOSDA* 语音合成研讨会 *(ETRW)*，1998年。[7] Maximilian Bisani 和 Hermann Ney，《用于字素到音素转换的联合序列模型》，*Speech Communication*，第 50 卷，第 5 期，第 434–451 页，2008年。[8] Lucian Galescu 和 James F Allen，《使用双向字素到音素转换的联合 n 元语法模型读出专有名称》，载于第七届国际口语处理会议，2002年。[9] Josef R Novak、Paul R Dixon、Nobuaki Minematsu、Keikichi Hirose、Chiori Hori 和 Hideki Kashioka，《通过对齐约束和 RNNLM n-best 重评分改进基于 WFST 的字素到音素转换》，载于国际语音通信协会第十三届年会，2012年。

[10] Josef R Novak、Nobuaki Minematsu 和 Keikichi Hirose，《联合 n-gram 模型和字素到音素转换的失败转移》，载于*Interspeech*，2013年，第 1821–1825 页。[11] Sevinj Yolchuyeva、G´eza N´emeth 和 B´alint Gyires-T´oth，《使用卷积神经网络进行字素到音素转换》，《应用科学》，第 9 卷，第 6 期，第 1143 页，2019 年。[12] Sevinj Yolchuyeva 、 G´eza N´emeth 和 B´alint Gyires-T´oth，《基于 Transformer 的字素到音素转换》，载于*Interspeech 2019* 会议论文集，2019年，第 2095–2099 页。[13] Kanishka Rao、Fuchun Peng、Has¸im Sak 和 Franc¸oise Beaufays，《使用长短期记忆循环神经网络进行字素到音素转换》，载于*2015*年 *IEEE* 国际声学、语音与信号处理会议 *(ICASSP)*。IEEE，2015年，第 4225–4229 页。[14] Alex Graves、Santiago Fern´andez、Faustino Gomez 和 J¨urgen Schmidhuber，《连接时序分类：使用循环神经网络标注未分段序列数据》，载于第 *23* 届国际机器学习会议论文集，2006年，第 369–376 页。[15] Baoguang Shi、Xiang Bai 和 Cong Yao，《一种可端到端训练的基于图像的序列识别神经网络及其在场景文本识别中的应用》，*IEEE* 模式分析与机器智能汇刊，第 39 卷，第 11 期，第 2298–2304 页，2016年。[16] Kyunghyun Cho、Bart van Merri¨enboer、Dzmitry Bahdanau 和 Yoshua Bengio，《神经机器翻译的特性：编码器–解码器方法》，载于*SSST-8* 论文集：第八届统计机器翻译语法、语义与结构研讨会，多哈，卡塔尔，2014年 10 月，第 103–111 页，计算语言学协会。[17] Sergey Ioffe 和 Christian Szegedy，《批归一化：通过减少内部协变量偏移加速深度网络训练》，载于国际机器学习会议。PMLR，2015年，第 448–456 页。[18] Dan Hendrycks 和 Kevin Gimpel，《高斯误差线性单元 (GELU)》，*arXiv* 预印本 *arXiv:1606.08415*，2016年。[19] Stanley F Chen 等，《用于字素到音素转换的条件模型和联合模型》，载于*INTERSPEECH*。Citeseer，2003年。