PRAK：捷克语自动语音对齐工具

Václav Hanžl[1],Adléta Hanžlová[2]

[1]自由研究人员， [2]布拉格查理大学语音学研究所 vhanzl@gmail.com, adleta.hanzlova@gmail.com

arXiv:2304.08431v1 [cs.CL] 17 Apr 2023

摘要

将语音标注到音素身份及其时间边界，是语音学研究中一项耗费大量人力的工作。为了简化这项工作，我们创建了一款免费的开源工具，可从捷克语文本生成音素序列，并将其与音频进行时间对齐。

架构复杂度低，使该设计易于语音学专业学生理解。这个含有 56k 个权重的 ReLU 神经网络声学模型，使用 PyTorch 在小规模 CommonVoice 数据集上训练。对齐与变体选择解码器使用 Python 和矩阵库实现。

捷克语发音生成器由简单的基于规则的模块组成，在可能的情况下捕捉语言逻辑，并允许修改转写方法的细节。

与迄今使用的工具相比，该工具提高了数据准备效率，可在 Mac、Linux 和 Windows 上通过 Praat 图形用户界面或命令行使用，能够较准确地选择发音变体（包括声门塞音检测），以算法方式捕捉了大部分捷克语同化逻辑，而且兼具教学性和实用性。

关键词： 语音对齐、分段、PyTorch、捷克语、Praat

1. 引言

为语音录音添加标注并确定音素边界，是语音学研究的重要组成部分。尽管已有软件工具可以自动完成这一流程，但其中许多工具只面向使用人数众多的语言，而且大多数并非免费提供；使用它们需要进行复杂的安装，还可能受到许可协议的限制。对于捷克语等较少见的语言，可选工具更为有限。

Praat 自带的 [1] 对齐工具适用于对齐单个词或较短的句子，但当音频中包含停顿时，其性能会显著下降。此外，该工具只能通过 Praat 查看与编辑窗口菜单使用，也无法自动对较大的数据集进行对齐。

迄今用于学术研究且被认为是现有最实用工具的 Czechaligner 是 Prague Labeller [2, 3] 基于 HTK 高斯混合模型，表现出令人赞叹的持久性。后续研究使用了 Kaldi [4]。其他临时开发的对齐方案始终未能最终定型为普遍适用的工具。

在语音学中，识别录音中的词语和音素及其时间边界非常有用，不仅有助于研究，也能满足教学需求，并帮助人们更好地理解音频。因此，我们认为，应让语音学家和学生都能广泛、便捷地使用一种可跨平台运行的工具，以自动完成此类流程并尽可能提高精度，而且无需用户具备编程知识。本文介绍了我们秉持这一理念开发的工具 Prak。

2. 设计目标与范围

在考察当前制作捷克语音素对齐研究数据的实践后，我们为新的对齐工具设定了几个目标：

- 可免费用于任何用途的开源工具——代码采用 MIT 许可证 [5]，基于免费的音频数据训练
- 可在 Mac、Linux 和 Windows 上运行
- 易于安装——依赖项少，且只依赖可靠、（希望）长期可用的组件
- 架构简单，最好基于语音学学生课程中所学的技术
- 可通过图形用户界面和命令行使用
- 尽可能采用可解释、可修改的逻辑（而非经过训练的黑盒）
- 自动选择发音变体

该工具需要将音频录音和文本转录稿作为输入。虽然我们可以设想通过自动语音识别来生成转录稿，但我们有意将这项任务留给外部工具（如果用户希望如此）；手动转写的工作量明显小于音素边界调整。这样，我们就可以始终使用最先进的自动语音识别，并将其与我们的音素对齐工具结合，而无需针对音素对齐的需求修改自动语音识别工具——在高斯混合模型和 HTK 时代，将这些任务合并到一个工具中曾是至关重要的做法，但现代自动语音识别工具的构建通常完全不再采用音素的概念。

初始版本仅支持捷克语，因为为捷克语研究提供切实帮助是我们的首要



图 1： 在 Praat 中使用的 Prak 输出 TextGrid。

目标。尽管如此，我们仍尝试通过使用多语言训练数据来源以及模块化代码设计，让扩展至更多语言变得简单。

3. 设计

3.1. 训练数据

捷克语拥有用于声学模型 (AM) 训练的高质量语音语料库，但许多数据集不适用于免费工具。有些是商业数据集，价格昂贵（SpeechDAT [6], Speecon[7]）；有些免费，但仅供研究使用。捷克语没有 LibriSpeech [8] 或 TEDx。可在线免费获取的捷克共和国议会众议院音频录音语料库 [9, 10] 规模庞大且免费，但其混响比标准录音棚录音更强。有些捷克语电视录音也可获取，但说话人数量较少。

我们决定先尝试使用 CommonVoice (CV)[11] 数据集。对于捷克语而言，该数据集的规模较小（含 14 小时训练数据，其中 4 小时为静音），但录音条件合适，涵盖的说话人数量众多，许可证限制极少，而且也提供许多其他语言的数据集。令我们惊讶的是，仅用 CV 就足以训练出质量合格的声学模型，因此 Prak 最初发布时附带的模型仅使用 CV 进行训练。

我们还非常感谢布拉格查理大学语音学研究所提供人工标注的录音，我们从中选取了 5 小时易于使用的数据。为使工具不依赖额外组件，我们没有将这些数据用于任何训练，而是使用该数据集的 10% 来评估工具，并将结果与潜在用户自行进行的人工标注进行比较。

3.2. 软件工具

目前可用的免费且精准的深度学习自动语音识别工具 [12] 自然会让人想到可以将其作为构建模块；然而，转向文本与音频之间的直接映射会省略音素级信息，因此需要对这些复杂工具进行大幅修改，才能将其用于音素对齐。

```
      ʔ          eu
f=ʔu:=.es=.a: pnéʊ̀matɪkɪ za_ éʊ̀ra nekóʊ̀pi:š
v=        ʔ                 _ʔ
                            |ʔ
```

图 2： 命令行输出示例，展示了带有发音变体的语音转写。

在较早的自动语音识别工具中，Kaldi [13] 仍接近 HTK 的传统隐马尔可夫模型方法，并且明确使用音素。不过，它作为软件系统中的依赖项相当复杂。因此，我们使用现代神经网络工具包 PyTorch [14] 来构建音素声学模型，但采用经典的隐马尔可夫模型方法，用 Python[15]从头设计了类似自动语音识别的基础架构。由于音素对齐只是自动语音识别中一个相当有限的子集，我们预计这是可行的。

PyTorch 中的矩阵库也有一个非常相近的替代方案：应用更为广泛的 NumPy [16], ；如果将来有此需要，至少可以轻松地重新实现推理部分。

3.3. 集成到 Praat 图形用户界面

Prak 与 Praat [1] 的集成旨在让没有编程知识的用户也能轻松使用。将包含主要 Python 工具的 Praat 脚本添加到 Praat 动态菜单中十分简单，用户只需单击一次即可对齐文本与音频——示例结果见图 1。该脚本还会检查多个输入文件，并提供其他功能，例如使用同一文本源对齐多个声音。

4. 发音生成器

4.1. 音标体系

Praat TextGrid 文件中的对齐输出使用 SAMPA [17]，以保持与既有惯例兼容。终端（命令行）输出使用更易读的捷克语国际音标，如 [18], 所示。图 2 展示了生成的发音及其变体示例。工具内部（源代码中）采用便于编程的编码方式，每个音素都由一个捷克语字符表示。

4.2. 文本清理

我们竭尽全力，确保能够在所有受支持的平台上处理各种常见文本。除了众所周知的行尾编码差异[19], ，还有一些更隐蔽的问题，例如不可见的 Unicode BOM 字符、带重音符号的字符的 NFC 和 NFD 形式等。我们希望已确保能够安全处理实践中出现的所有变体。

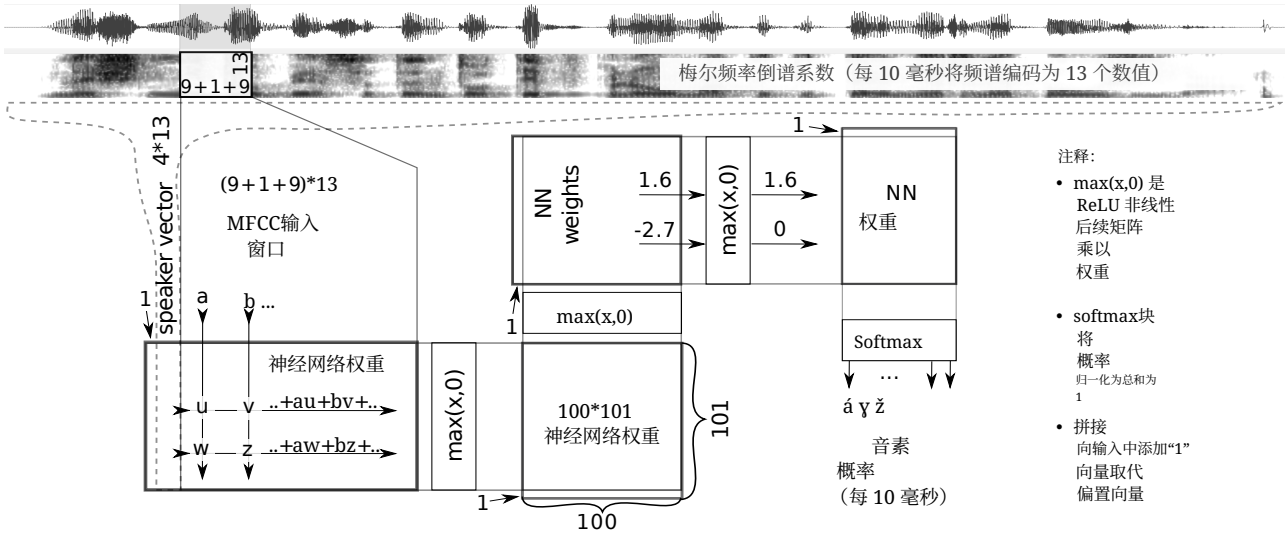


图 3： 使用 ReLU 神经网络堆栈将 MFCC 窗口转换为音素概率 s.

4.3. 外来词

用户可以通过添加替换规则来设置例外。对于每个单词，规则中匹配最长的部分会被替换为该规则中的所有发音变体（例如，规则 "washington vošingtn" 可以替换 "Washingtonu" 中的部分内容）。替换后的文本不会再参与匹配，但替换前后剩余的单词部分仍可能进行其他（较短的）匹配。生成的文本消除了外来捷克语单词与本族捷克语单词之间的差异。

我们发现，最长匹配优先的方法比 [3] 中使用的方法更实用；后者按顺序依次尝试规则，因此很难在数百条旧规则中确定添加新规则的合适位置。

4.4. 逆向有限状态转换器中的同化逻辑

许多捷克语同化现象都可以用有限状态转换器（FST）表示，它只需少量状态（通常只有 2 或 3 个），即可反向处理音素序列，甚至处理距离相当远的影响。例如，在序列“vošingtnu”（由上述替换规则生成）中，处理[t] 会使 FST 状态变为清化，而 [g] 则变为清音 [k]。在另一个 FST 中，[k] 或 [g] 会使状态软腭化，受影响的 [n] 则变为 [ŋ]。我们将多个 FST 组合起来处理以下现象：

- 声门塞音 [ʔ]，元音间的 [j]
- 浊音同化
- 包含 dtn/dťťn 的辅音群，例如“ntní” [ntɲiː/ncɲiː/ɲcɲiː]
- bˇe/pˇe/vˇe/mˇe/fˇe [bjɛ, pjɛ, vjɛ, mɲɛ, fjɛ]
- nk/ng 中的软腭化 [ŋk/ŋg]

有限状态转换器（FST）会提供多个输出（尤其是在词边界处需要），最终由声学模型决定采用哪一个。

5. 音素声学模型

虽然在语音处理中，Transformer [20, 21]或至少卷积结构会让人联想到适当的先进技术，但我们的目标有限，只需实现音素对齐，因此看起来用简单得多的架构就能达成，这也大大降低了学生理解该工具内部工作原理的门槛。因此，我们从一个简单的层堆栈开始，由矩阵乘法和 ReLU（修正线性单元）组成，即对向量的每个元素应用 max(x, 0)，并在各层之间传递该向量（见图 3）。

神经网络的输入是连续的 19 个 MFCC 帧，外加一个“说话人向量”，它是 4 个能量频带上的平均 MFCC（帧先按能量高于或低于平均值进行划分，然后再细分）。我们通过 PyTorch 计算类似 Kaldi 的 MFCC。

ReLU 层堆栈只将音频中的一个位置映射为一个音素假设，而不是进行序列到序列的映射。为此，我们采用了经典的隐马尔可夫模型结构，用 ReLU 神经网络取代高斯混合模型。我们的训练方法结合了 Baum-Welch 重新估计与通过梯度下降训练神经网络：交替进行目标音素序列的时间重新对齐，以及让声学模型猜测这些音素的梯度下降。

音素序列通过维特比对齐进行推测，所用的音素概率由神经网络声学模型估计；而神经网络声学模型本身则以音素序列为训练目标，通过梯度下降进行训练

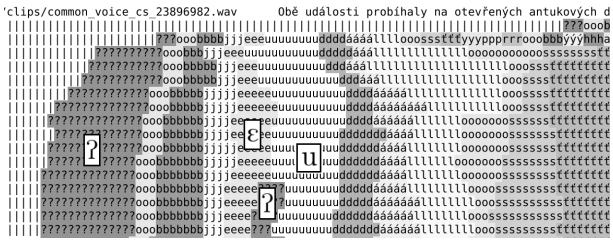


图 4： 训练期间的音素对齐，各行显示训练轮次。只有在后期才会出现可选的第二个声门塞音，从而导致 [ʔobjɛʔu...]

。HTK 等工具会通过大胆猜测初始对齐，来解决 Baum-Welch 重估中的类似循环依赖问题。我们采用的方法与此类似：对于初始对齐，假设每个音素时长均为 30 毫秒，并且录音中语音前后的静音时长相等。系统能够从这一（大部分不准确的）初始对齐出发，逐渐收敛到非常好的音素声学模型和音素对齐。图 4 展示了训练过程的示例。

要实现收敛，唯一需要的调整是人为提高罕见音素的权重。在维特比解码中，我们根据整个训练数据集中当前被分类为该音素的帧总数，调整了音素概率。

6. 音素对齐的评估

我们以人工对齐数据为参照，比较了 Prak 与 Pragues Labeller [3]，的性能。由于对齐器使用的音素集有所不同（例如，Prak 还会检测声门塞音，同时使用浊音和清音“ˇr”，并会考虑词边界处的同化等），因此直接比较并不容易。人工对齐虽然经过了大量细致的工作，但采用了几种略有不同的方法，并且存在一些音素身份错误。对实际用户来说，一个重要参数是需要通过移动多个音素边界来纠正的对齐错误频率，因此我们尝试检测较大的音素边界错位。

我们比较了发音，并统计了音素插入、删除和替换。在匹配的位置，我们对照人工参考测量了音素中心时间偏移（尽管音素的具体身份可能存在争议，[1] 但我们的人工参考数据中的时间位置非常精确）。我们将超出阈值的时间不匹配数量作为质量指标，结果如表 1 所示。

7. 未来工作

我们特别强调简洁性，唯一较大的工具是一个深度学习框架，且仅

test	PL [3]	Prak
音素不匹配	6.61	1.88
匹配但错位 0.1s+	**4.28**	**0.36**
匹配但错位 0.2s+	3.22	0.09
错配或错位 0.1秒+	10.89	2.24

表 1： 对比使用最广泛的工具与我们的工具，音素错配和边界错位的百分比。

用于基本的 ReLU 堆叠。我们没有使用有限状态转换器工具包（Kaldi 明确使用的一种基础工具，而 HTK 中也具备此功能，但它融入了更高层），而是采用简单的“香肠”结构来捕捉发音变体。目前的自动语音识别工具包中有许多出色的现成构建模块，非常适合在此任务中进行测试。

尽管现代自动语音识别工具已不再显式设计音位或音素等中间层表示，但仍可以将文本映射为音素序列，并在音素而非字符上训练例如 wav2vec2 [22] 架构，然后使用诸如 [23] 之类的方法提取音素的时间边界。

另一种方案是为声学模型采用更复杂的神经网络结构，例如 Transformer。问题在于，这种结构关注距离过远的音频部分，会在多大程度上模糊时间边界。这样的声学模型或许只适合作为第一层对齐，将音频与文本按词对齐，随后再由像我们这样更局部的声学模型进行细粒度对齐。

还可以进一步微调音素边界。对于某些音素对，将边界向频谱变化处移动或许会有所帮助；其他音素对则需要采用不同的处理方式——事实上，这里的挑战在于将 [18] 转换为一种编程语言。

我们的说话人向量当然可以替换为 i-vector [13]或 X向量 [24], ，以显著增加复杂度为代价换取更好的结果。

我们希望我们的对齐工具能为此类实验提供一个实用的框架和基准。

8. 结论

我们开发了一款实用的捷克语音素对齐工具，在边界错位度量方面显著优于迄今为止最好的工具。Prak 还提供了自动变体选择等新功能，并且对有问题的输入文本更具适应性。我们认为，最大的成就是为研究人员和学生提供了一款不受限制、架构简单易懂的开源工具，希望能让其他人以此为基础继续开发。可在此免费下载源代码：github.com/vaclavhanzl/prak

9. 参考文献

[1]P. Boersma 和 D. Weenink，*Praat*：用计算机进行语音学研究 *[*计算机程序*]*。版本 *6.1.56*，http://www.praat.org/，2021年。[2] P. Pollák、J. Volín 和 R. Skarnitzl，“HMM´参数对音素切分准确度的影响——评估基线”，载于第*16*届会议暨第*15*届捷克—德国“语音处理”研讨会论文集，第1卷，2005年，第302–309页。[3] ——，“Praat 环境中的基于隐马尔可夫模型的语音切分”，载于第十二届国际语音与计算机会议——*SPECOM*，2007年，第537–541页。[4]Z. Patc、P. Mizera 和 P. Pollak，“利用 KALDI 进行语音切分及因果语境捷克语语音中的简化发音检测”，载于文本、语音与对话。施普林格国际出版，2015年，第433–441页。[5]“MIT 许可证。”[在线]。网址：https: //opensource.org/licenses/MITˇ[6]P. Pollák、J. Cernocký、J. Boudy、K. Choukri、J. Kochanina、W. Majewski、V. Ostroukhov、M. Rusko、J. Sadowski、R. Siemund、P. Staroniewicz、M. Trnka、H. Tropf、K. Vicsi、H. Heuvel 和 A. Virag，“SpeechDat(E)——东欧电话语音数据库”，载于*LREC'2000*卫星研讨会 *XLDB*——超大型电话语音数据库论文集，2000年，第20–25页。[7]“捷克 Speecon 数据库。”[在线]。网址：http://catalog.elra.info/en-us/repository/browse/ELRA-S0298/[8]V. Panayotov、G. Chen、D. Povey 和 S. Khudanpur，“LibriSpeech：基于公有领域有声读物的自动语音识别语料库”，载于*2015*年 *IEEE* 国际声学、语音与信号处理会议（*ICASSP*），2015年，第5206–5210页。[9] M. Kopp、V. Stankov、O. Bojar、B. Hladká 和 P. Straˇnák，*ParCzech 3.0*。查理大学数学与物理学院形式语言学与应用语言学研究所（ÚFAL）的 LINDAT/CLARIAH-CZ 数字图书馆，2021年。 [在线]。网址：http://hdl.handle.net/ 11234/1-3631[10]“捷克—斯洛伐克联合数字议会图书馆。”[在线]。网址：https://www.psp. cz/eknih/[11] R. Ardila、M. Branson、K. Davis、M. Henretty、M. Kohler、J. Meyer、R. Morais、L. Saunders、F. M. Tyers 和 G. Weber，“Common Voice：大规模多语言语音语料库”，*arXiv* 预印本 *arXiv:1912.06670*，2019年。[12] A. Radford、J. W. Kim、T. Xu、G. Brockman、C. McLeavey 和 I. Sutskever，“通过大规模弱监督实现稳健的语音识别”，*arXiv* 预印本 *arXiv:2212.04356*，2022年。[13] D. Povey、A. Ghoshal、G. Boulianne、L. Burget、O. Glembek、N. Goel、M. Hannemann、P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely，“Kaldi语音识别工具包”，发表于*2011*年*IEEE 2011*年自动语音识别与理解研讨会，2011年12月。[14]A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala，“PyTorch：一种命令式风格、高性能深度学习库”，发表于第*32*届神经信息处理系统进展，2019年，第8024–8035页。[15]G. Van Rossum and F. L. Drake, *Python 3*参考手册。CreateSpace，2009年。[16]C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant，“使用 NumPy 进行数组编程”，《自然》，第585卷，第7825期，第357–362页，2020年9月。[17]“捷克语 SAMPA。”[在线]。网址：https: // www.phon.ucl.ac.uk/home/sampa/czech-uni.htm[18] P. Machaˇc and R. Skarnitzl，语音的语音学分段。Epocha，2010年。[19] “CR/LF问题与文本行结束符。”[在线]。网址：https://portal.perforce.com/s/article/ 3096 [20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin，“Attention Is All You Need”，发表于神经信息处理系统进展，第30卷，2017年。[21] T. Lin, Y. Wang, X. Liu, and X. Qiu，“Transformer综述”，*AI Open*，第3卷，第111–132页，2022年。[22] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli，“wav2vec 2.0：一种用于自监督学习语音表示的框架”，神经信息处理系统进展，第33卷，第12 449–12 460页，2020年。[23]M. Hira，“使用Wav2Vec2进行强制对齐。”[在线]。网址：https://pytorch.org/audio/main/tutorials/forced_alignment_tutorial.html#forced-alignment-with-wav2vec2[24] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur，“X向量：用于说话人识别的鲁棒深度神经网络嵌入”，发表于*2018*年*IEEE*国际声学、语音与信号处理会议 *(ICASSP)*，2018年，第5329–5333页。

[1] 我们人工检查了测试集并纠正了明显错误；不过，考虑到测试集的规模（完整 5 小时数据集中 10% 的测试子集包含 2 万个音素），即使经过这轮额外的人工工作，我们也肯定没有纠正所有错误。