PRAK: AN AUTOMATIC PHONETIC ALIGNMENT TOOL FOR CZECH

Václav Hanžl[1], Adléta Hanžlová[2]

[1]Freelance researcher, [2]Institute of Phonetics, Charles University in Prague
vhanzl@gmail.com, adleta.hanzlova@gmail.com

ABSTRACT

Labeling speech down to the identity and time boundaries of phones is a labor-intensive part of phonetic research. To simplify this work, we created a free open-source tool generating phone sequences from Czech text and time-aligning them with audio.

Low architecture complexity makes the design approachable for students of phonetics. Acoustic model ReLU NN with 56k weights was trained using PyTorch on small CommonVoice data. Alignment and variant selection decoder is implemented in Python with matrix library.

A Czech pronunciation generator is composed of simple rule-based blocks capturing the logic of the language where possible, allowing modification of transcription approach details.

Compared to tools used until now, data preparation efficiency improved, the tool is usable on Mac, Linux and Windows in Praat GUI or command line, achieves mostly correct pronunciation variant choice including glottal stop detection, algorithmically captures most of Czech assimilation logic and is both didactic and practical.

Keywords: phonetic alignment, segmentation, PyTorch, Czech, Praat

1. INTRODUCTION

Labeling speech recordings and identifying phone boundaries is a significant part of phonetic research. Even though there are software tools to automate this process, many of them target only languages with great quantities of speakers and most are also not freely available, their use requires complex installation and can be restricted by license agreements. Tool choices are even more limited for less common languages like Czech.

Praat's [1] own integrated alignment tool is usable when aligning individual words or short sentences, but its performance is significantly worsened when the audio contains pauses. It also only works from the Praat *View & Edit* window menu and doesn't enable automatic alignment of larger datasets.

The Czech aligner used in academic research so far and preferred as the most useful available is Prague Labeller [2, 3] which is based on HTK GMM models and shows admirable longevity. Subsequent research used Kaldi [4]. Other ad-hoc alignment solutions were never finalized as universally usable.

As identifying words and phones and their time boundaries in a recording is very useful in phonetics not only for research purposes, but also for educational needs and general better orientation in the audio, we find it important that a cross-platform tool enabling the automation of such processes and striving to be as precise as possible is widely available and easy to access by phoneticians and students and requires no programming knowledge from the user. In this paper, we present Prak – a tool we developed with this idea in mind.

2. DESIGN GOALS AND SCOPE

Observing current practices when preparing Czech phone-aligned research data, we had several goals for our new alignment tool:

- an open-source tool free for any use – MIT license [5] for code, trained on free audio data
- functioning on Mac, Linux and Windows
- easy to install – low dependencies, only reliable dependencies which are (hopefully) here to stay
- simple architecture, preferably building on techniques from phonetics students' curricula
- usable from both GUI and command line
- using explainable and modifiable logic (rather than a trained blackbox) where possible
- automatic pronunciation variant selection

The tool expects an audio recording and text transcript as an input. While we can imagine doing the transcript via ASR, we deliberately left this task to an external tool if so desired (manual transcription is a noticeably smaller task than phone boundary adjustments). This way we can always use a state of the art ASR and combine it with our phone alignment tool, without modifying the ASR tool for phone alignment needs – merging these tasks into one tool used to be a vital approach in the GMM HTK era, but modern ASR tool construction often abandons the notion of phones altogether.

The initial release supports Czech language only, as practical help to Czech research was our primary

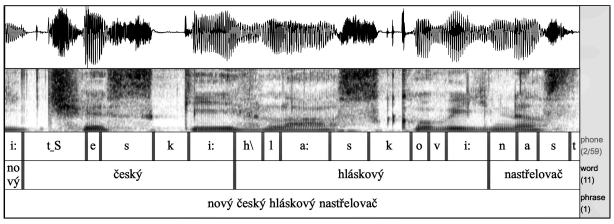


Figure 1: Output textgrid of Prak used in Praat.

goal. Nevertheless, we tried to make extensions to more languages easy by using a multilingual training data source and by modular design of the code.

3. DESIGN

3.1. Training data

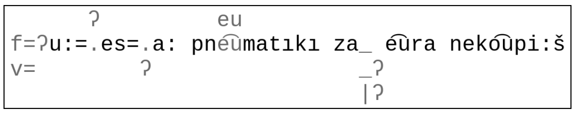
Czech language has high quality speech corpora for acoustic model (AM) training, however, many data sets are unsuitable for a free tool. Some are commercial and costly (SpeechDAT [6], Speecon [7]), some free of cost, but for research use only. There is no LibriSpeech [8] or TEDx for Czech. Corpus of audio recordings from the Chamber of Deputies of the Parliament of the Czech Republic available freely online [9, 10] is big and free, but contains reverberations stronger than standard studio recordings. Some Czech TV recordings are available, but have a low number of speakers.

We decided to first try the CommonVoice (CV) [11] dataset, the size of which is marginal for Czech (14 hours of training data, 4 hours being silence), but it has suitable recording conditions, contains a high number of speakers, has an extremely permissive license and is available for many other languages. To our surprise, CV itself allowed us to train an AM of sufficient quality, so we initially released Prak with a model trained on CV alone.

We are also very grateful to the Institute of Phonetics, Charles University in Prague, for providing their manually labeled recordings from which we selected 5 hours of easily usable data. We did not use this data in any training to keep our tool free of additional dependencies, but we used 10% of this dataset to evaluate the tool by comparison to human labeling made by potential users themselves.

3.2. Software tools

Currently available free and precise deep learning ASR tools [12] naturally come to mind as building blocks, however, the shift towards direct mapping between text and audio omits the phone level, requiring significant modifications of these complex tools to repurpose them for phone alignment.



```
        ʔ           eu
f=ʔu:=.es=.a: pnéu͡matɪkɪ za_ é͡ura neko͡upi:š
v=        ʔ                     _ʔ
                                |ʔ
```

Figure 2: Command line output showing phonetic transcription with pronunciation variants.

From older ASR tools, Kaldi [13] is still close to the traditional HMM-based approach of HTK and explicitly uses phones. It is however quite a complex dependency in a software system. We therefore used a modern NN toolkit PyTorch [14] for construction of the phone AM, but designed our ASR-like infrastructure from scratch in Python [15] using a classical HMM approach. As phone alignment is quite a limited subset of ASR, we expected this to be feasible.

The matrix library in PyTorch also has a very close alternative in the even more broadly used NumPy [16], allowing easy reimplementation of at least the inference part, should such need ever arise.

3.3. Integration in Praat GUI

The Praat [1] integration of Prak is designed to be simple to use without any programming knowledge. A Praat script which embeds the main Python tool is easy to add to Praat dynamic menu, so users can align text with audio in just one click – see fig. 1 for example result. The script also performs several input file checks and adds additional features such as aligning multiple sounds using one text source.

4. PRONUNCIATION GENERATOR

4.1. Phonetic alphabets

Alignment output in Praat TextGrid files uses SAMPA [17] to stay compatible with established practice. Terminal (command line) output uses much more readable IPA for Czech as used in [18], see fig. 2 for an example of generated pronunciation and its variants. Internally (in the source code) the tool uses programming-friendly encoding where each phone is represented by one Czech character.

4.2. Text cleanup

Great care was taken to be able to process all usual texts on all the supported platforms. Apart from notoriously known end-of-line encoding differences [19], there are more subtle pitfalls like invisible Unicode BOM characters, NFC and NFD forms of accented characters etc. We hopefully made it safe to process all the variants happening in practice.

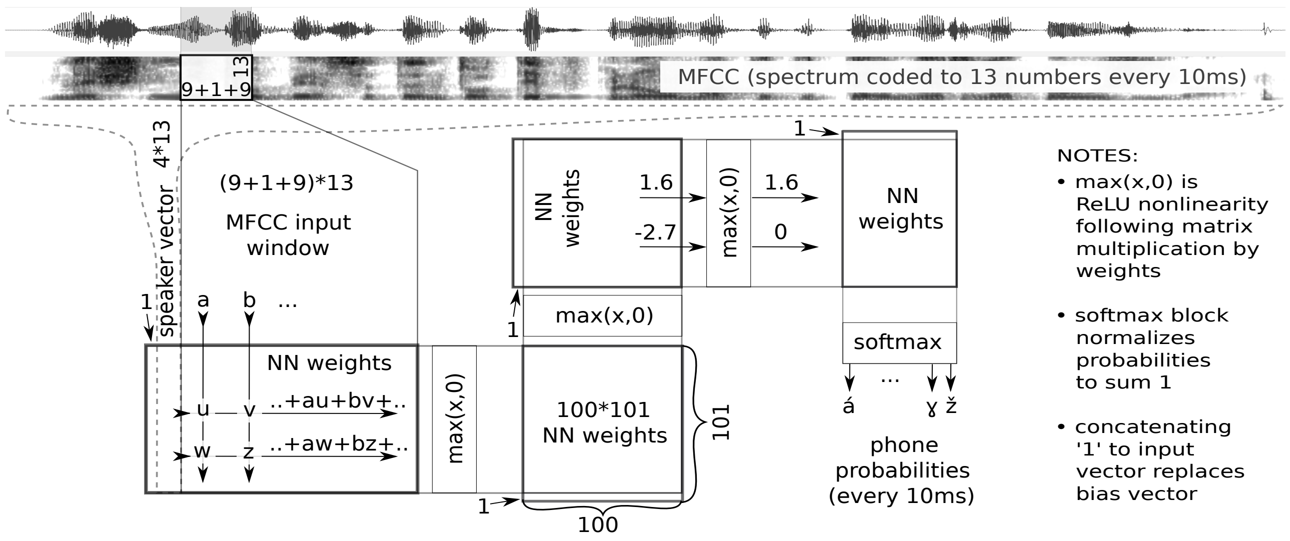

MFCC (spectrum coded to 13 numbers every 10ms)
9+1+9
13
4*13
speaker vector
(9+1+9)*13
MFCC input window
1
a
b
...
NN weights
u — v ..+au+bv+..
w — z ..+aw+bz+..
max(x,0)
1
100*101
NN weights
101
100
max(x,0)
1
NN weights
1.6
-2.7
max(x,0)
1.6
0
1
NN weights
softmax
...
á ɣ ž
phone probabilities (every 10ms)
NOTES:
• max(x,0) is ReLU nonlinearity following matrix multiplication by weights
• softmax block normalizes probabilities to sum 1
• concatenating '1' to input vector replaces bias vector

Figure 3: Acoustic model with ReLU NN stack converting MFCC window to phone probabilities.

4.3. Foreign words

Users can add exceptions in the form of replacement rules. For each word, the longest match in rules is replaced by all pronunciation variants from the rule (e.g. rule ”washington vošingtn” can do replacement in ”Washingtonu”). Replaced text is not matched anymore, but parts of words before and after replacement are subject to potential additional (shorter) matches. The resulting text eliminates differences between foreign and native Czech words.

We find the longest match priority more practical than the approach used in [3] where rules are tried in sequence, making it hard to identify the right place for adding new rules among hundreds of old ones.

4.4. Assimilation logic in backward FSTs

Many Czech assimilations can be expressed as Finite State Transducer (FST) with a small number of states (often just 2 or 3) processing the phone sequence backwards, including rather far distance effects. For example, in the sequence ”vošingtnu” (made by the replacement rule above), processing [t] changes FST state to *devoicing* and [g] turns into voiceless [k]. In another FST, [k] or [g] forces state *velarization* and affected [n] becomes [ŋ]. We use a convolution of FSTs taking care of:

- glottal stop [ʔ], intervocalic [j]
- assimilation of voicing
- consonant groups containing dtn/ďťň such as “ntní” [ntɲiː/ncɲiː/ɲcɲiː]
- bě/pě/vě/mě/fě [bjɛ, pjɛ, vjɛ, mɲɛ, fjɛ]
- velarization in nk/ng [ŋk/ŋg]

FSTs provide multiple outputs (needed esp. on word boundaries), leaving the final choice to the AM.

5. PHONE ACOUSTIC MODEL

While transformers [20, 21] of various kinds or at least convolutional structures come to mind as appropriate state of the art in speech processing, our limited goal of only phone alignment seemed achievable with much simpler architectures, significantly lowering the barrier for students trying to grasp the inner workings of the tool. We therefore started with a simple stack of layers composed of matrix multiplication and ReLU – Rectifying Linear Units, $\max(x, 0)$ – applied to each element of a vector which is passed between layers (see fig. 3).

As input, the NN gets 19 consecutive MFCC frames plus a ”speaker vector” which is an average MFCC in 4 energy bands (frames are split to above and below average energy and then sub-split again). We compute Kaldi-like MFCCs via PyTorch.

The stack of ReLU layers only maps one position in the audio to one phone hypothesis, not sequence to sequence. For this purpose, we stuck to a classical HMM structure, replacing a GMM model with ReLU NN. Our training is a cross between Baum-Welch reestimation and NN training by gradient descent, alternating phases of time re-alignment of the target phone sequence and gradient descent of the AM guessing these phones.

Phone sequences are guessed by Viterbi alignment using phone probabilities estimated by NN AM, while the NN AM itself is trained by gradient descent, using the phone sequences as training

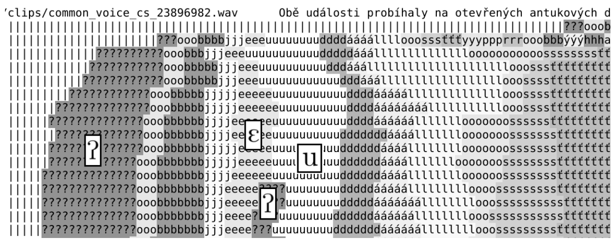

'clips/common_voice_cs_23896982.wav Obě události probíhaly na otevřených antukových d
ʔ
ɛ
u
ʔ

Figure 4: Phone alignments during training, rows show train epochs. An optional second glottal stop is only found later, leading to [ʔobjɛʔu...]

targets. Tools like HTK solve similar circular dependencies in Baum-Welch reestimation by a wild initial alignment guess. What we do is similar: For the initial alignment, every phone is presupposed to be 30ms long with equal silences preceding and following speech in a recording. The system is able to converge from this (mostly false) initial alignment to a very good phone AM and very good phone alignments. Fig. 4 shows example training dynamic.

The only adjustment needed for convergence is an artificial boost of rare phones. In Viterbi decoding, we adjusted phone probabilities according to the current overall number of frames classified as this phone in the whole training dataset.

6. EVALUATION OF PHONE ALIGNMENTS

We compared the performance of Prak with Prague Labeller [3], using manually aligned data as a reference. Direct comparison is not easy as phone sets differ between aligners (e.g. Prak also detects glottal stops, uses both voiced and voiceless "ř", considers assimilation at word boundaries etc.). The manual alignment, while made with great effort and care, uses several slightly different approaches and contains some phone identity errors. An important parameter for practical users is the frequency of misalignments which must be corrected by moving multiple phone boundaries, therefore we tried to detect major phone boundary misplacements.

We compared pronunciations, counting phone insertions, deletions and substitutions. At places of match, we measured phone center time shifts against the manual reference (while the exact phone identity may be questioned,[1] the time positions are very exact in our manual reference data). The count of time mismatches exceeding a threshold was used as a quality measure, results are shown in tab. 1.

test	PL [3]	Prak
phone mismatch	6.61	1.88
match but misplace 0.1s+	**4.28**	**0.36**
match but misplace 0.2s+	3.22	0.09
mismatch or misplace 0.1s+	10.89	2.24

Table 1: Percentage of phone mismatch and boundary misplacement, comparing most used tool to ours.

7. FUTURE WORK

We highly emphasized simplicity, the only bigger tool being a deep learning framework, only applied to a basic ReLU stack. We avoided FST toolkit (a basic tool which Kaldi explicitly uses, unlike HTK where this functionality is present but dissolved in higher layers) and used a simple "sausage" structure to capture pronunciation variants. There are many wonderful ready-to-use building blocks in current ASR toolkits which would be great to test in this task.

While modern ASR tools abandoned explicit design of middle layer representation like phonemes or phones, it would be possible to map text to a phone sequence and train e.g. wav2vec2 [22] architecture on phones instead of characters and extract time boundaries of phones using a method such as [23].

Another option would be to use more complex NN structures like transformers for the AM. The question is how much would time boundaries be smeared by the structure paying attention to too distant parts of audio. Such AM would perhaps only be useful as a first layer alignment anchoring audio to text on the level of words, followed by fine-grained alignment by a more local AM like ours.

Phone boundaries could be further fine-tuned. Movements of boundaries towards spectrum change might help for some phone pairs, different processing for others – in fact, the challenge here is translating [18] into a programming language.

Our speaker-vectors can certainly be replaced by i-vectors [13] or x-vectors [24], trading better results for considerably increased complexity.

We hope that our alignment tool can serve as a useful framework and baseline for such experiments.

8. CONCLUSION

We made a practical Czech phone alignment tool which significantly outperforms the best tool available so far in the boundary misalignment measure. Prak also provides new features like automatic variant selection and is more resilient to problematic input texts. We consider our biggest achievement that we gave researchers and students an open-source tool with no restrictions and simple to understand architecture, hopefully allowing others to build on it. Free download of the source code is possible here: `github.com/vaclavhanzl/prak`

9. REFERENCES

[1] P. Boersma and D. Weenink, *Praat: doing phonetics by computer [Computer program]. Version 6.1.56*, http://www.praat.org/, 2021.

[2] P. Pollák, J. Volín, and R. Skarnitzl, "Influence of HMM´s parameters on the accuracy of phone segmentation - evaluation baseline," in *Proceedings of the 16th Conference Joined with the 15th Czech-German Workshop "Speech Processing"*, vol. 1, 2005, pp. 302–309.

[3] ——, "HMM-based phonetic segmentation in Praat environment," in *The XII International Conference Speech and Computer - SPECOM*, 2007, pp. 537–541.

[4] Z. Patc, P. Mizera, and P. Pollak, "Phonetic segmentation using KALDI and reduced pronunciation detection in causal Czech speech," in *Text, Speech, and Dialogue*. Springer International Publishing, 2015, pp. 433–441.

[5] "The MIT license." [Online]. Available: https://opensource.org/licenses/MIT

[6] P. Pollák, J. Černocký, 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, and A. Virag, "SpeechDat(E) - Eastern European telephone speech databases," in *Proceedings LREC'2000 Satelite workshop XLDB - Very Large Telephone Speech Databases*, 2000, pp. 20–25.

[7] "Czech Speecon database." [Online]. Available: http://catalog.elra.info/en-us/repository/browse/ELRA-S0298/

[8] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5206–5210.

[9] M. Kopp, V. Stankov, O. Bojar, B. Hladká, and P. Straňák, *ParCzech 3.0*. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University, 2021. [Online]. Available: http://hdl.handle.net/11234/1-3631

[10] "Společná česko-slovenská digitální parlamentní knihovna." [Online]. Available: 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, and G. Weber, "Common Voice: A massively-multilingual speech corpus," *arXiv preprint arXiv:1912.06670*, 2019.

[12] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," *arXiv preprint arXiv:2212.04356*, 2022.

[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, "The Kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*, Dec. 2011.

[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: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, 2019, pp. 8024–8035.

[15] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. CreateSpace, 2009.

[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, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020.

[17] "Czech SAMPA." [Online]. Available: https://www.phon.ucl.ac.uk/home/sampa/czech-uni.htm

[18] P. Machač and R. Skarnitzl, *Fonetická segmentace hlásek*. Epocha, 2010.

[19] "CR/LF issues and text line-endings." [Online]. Available: 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," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[21] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," *AI Open*, vol. 3, pp. 111–132, 2022.

[22] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in Neural Information Processing Systems*, vol. 33, pp. 12 449–12 460, 2020.

[23] M. Hira, "Forced alignment with Wav2Vec2." [Online]. Available: 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-Vectors: Robust DNN embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5329–5333.

[1] We manually checked the test set and corrected obvious errors, however given its size (there are 20k phones in the 10% test subset of the full 5 hour set), we certainly did not correct all errors even by this additional pass of human work.