The Journal of Open Source Software

Speakerbox: Few-Shot Learning for Speaker Identification with Transformers

Eva Maxfield Brown [1], **To Huynh** [2], **and Nicholas Weber** [1]

1 University of Washington Information School, University of Washington, Seattle **2** University of Washington, Seattle

DOI: 10.21105/joss.05132

Software

- Review
- Repository
- Archive

Editor: Fabian-Robert Stöter

Reviewers:

- @desilinguist
- @FabianHofmann

Submitted: 25 October 2022
Published: 17 March 2023

Summary

Automated speaker identification is a modeling challenge for research when large-scale corpora, such as audio recordings or transcripts, are relied upon for evidence (e.g. Journalism, Qualitative Research, Law, etc.). To address current difficulties in training speaker identification models, we propose Speakerbox: a method for few-shot fine-tuning of an audio transformer. Specifically, Speakerbox makes multi-recording, multi-speaker identification model fine-tuning as simple as possible while still fitting an accurate, useful model for application. Speakerbox works by ensuring data are safely stratified by speaker id and held-out by recording id prior to fine-tuning of a pretrained speaker identification Transformer on a small number of audio examples. We show that with less than an hour of audio-recorded input, Speakerbox can fine-tune a multi-speaker identification model for use in assisting researchers in audio and transcript annotation.

Statement of Need

Speaker-annotated transcripts from audio recordings are an increasingly important applied research problem in natural language processing. For example, speaker-annotated audio and transcript data has previously been used to create comprehensive analyses of conversation dynamics (Jacobi & Schweers, 2017; Maltzman & Sigelman, 1996; Miller & Sutherland, 2022; Morris, 2001; Osborn & Mendez, 2010; Slapin & Kirkland, 2020). However, multi-speaker audio classification models (for the purpose of speaker identification) can be cumbersome and expensive to train and unwieldy to apply. Speaker diarization is, "the unsupervised identification of each speaker within an audio stream and the intervals during which each speaker is active" (Anguera et al., 2012). Diarization is a useful method in certain applications of large-scale automated analysis and there are free tools available to perform these tasks (e.g. Bredin et al., 2020; Bredin & Laurent, 2021). However, due to it's unsupervised nature, diarization may inconsistently label speakers across different audio recordings. As such, diarization may not adequately meet the needs for research purposes depending upon the identification of speakers (e.g. Journalism, Qualitative Research, Law, etc.).

Speakerbox is built with the goal of making multi-recording, multi-speaker identification model training as simple as possible while still attempting to help train an accurate, useful model for application (e.g. a set of recordings where each recording has some subset of a set of speakers).

To this end Speakerbox provides functionality to:

1. Create or import annotation sets;
2. Prepare an audio dataset into stratified and held-out, train, test, and validation subsets;
3. Train and evaluate a fine-tuned speaker identification model; and,
4. Apply a speaker identification model to an audio file.

Related Work

While there is continuous research in new methods and model architectures for few-shot speaker identification models (Kumar et al., 2020; Li et al., 2022; Wolters et al., 2020), there exists little work in creating an open-source, easy-to-use library for their training and evaluation.

For more general speech processing and filtering, SpeechPy is an open-source solution for "speech processing and feature extraction ... [by providing the] most frequent used speech features including MFCCs and filterbank energies" (Torfi, 2018).

Open-source libraries more closely related to diarization and speaker identification include Pyannote.Audio and Transformers. Pyannote.Audio "provides a set of trainable end-to-end neural building blocks that can be combined and jointly optimized to build speaker diarization pipelines" (Bredin et al., 2020; Bredin & Laurent, 2021). While Transformers is a library which "provides thousands of pretrained models to perform tasks on different modalities such as text, vision, and audio" (Wolf et al., 2020). Speakerbox makes use of both `pyannote.audio` and `transformers`.

Speaker diarization (provided by `pyannote.audio`) is the *unsupervised* process of splitting audio into segments grouped by speaker identity. Speakerbox in comparison, provides functionality to train a model for speaker indentification, the *supervised* process of splitting audio into segments and identifying them as a known speaker.

The Transformer architecture was originally introduced by Vaswani, et al. for use in sequence-to-sequence machine learning tasks such as machine translation (Vaswani et al., 2017). However, there are now many researchers creating or fine-tuning Transformer-based models to suit there own needs. The `transformers` library provided by HuggingFace is one such library that provides an API for downloading, using, and/or fine-tuning Transformer-based models. In our case, Speakerbox utilizes the `transformers` library to fine-tune a speaker identification model made available by the HuggingFace platform.

There are also paid solutions for quickly annotating and training a custom speaker diarization and audio classification model such as ExplosionAI's Prodigy Platform.

Finally, SetFit ("an efficient and prompt-free framework for few-shot fine-tuning of Sentence Transformers") provides similar functionality as Speakerbox but for text-based data (Tunstall et al., 2022).

Speakerbox brings many of these tools together in order to be a solution for speaker identification as an easy-to-use and open-source library for the annotation of audio data and the fine-tuning of a speaker identification model.

Functionality

Speakerbox attempts to simplify transformer fine-tuning by making the following processes easier:

- Creation of a dataset via diarization;
- Import of a dataset from existing annotation platforms;
- Preparation of train, test, and validation subsets protected from speaker and recording data leakage;
- Fine-tuning and evaluation of a transformer; and,
- Application of a trained model across an audio file

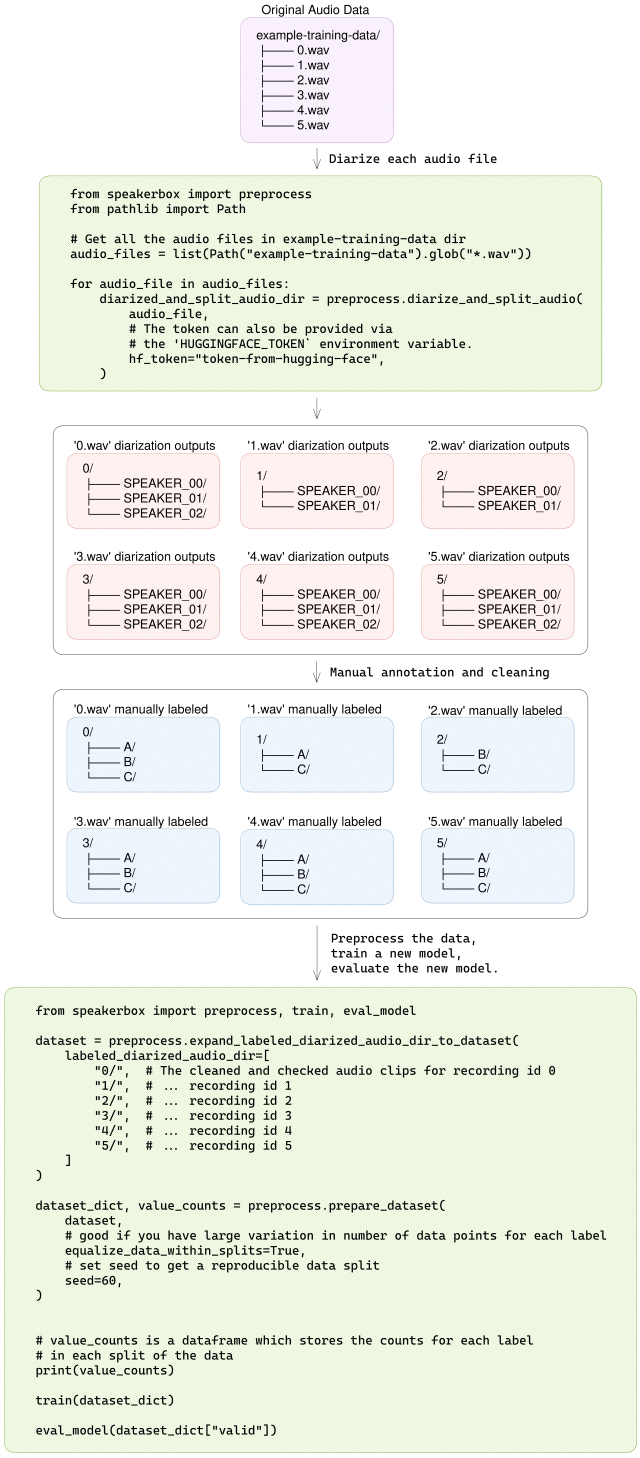

Figure 1: Typical workflow to prepare a speaker identification dataset and fine-tune a new model using tools provided from the Speakerbox library. The user starts with a collection of audio files that include portions speech from the speakers they want to train a model to identify. The `diarize_and_split_audio` function will create a new directory with the same name as the audio file, diarize the audio file, and finally, sort the audio portions produced from diarization into sub-directories within this new directory. The user should then manually rename each of the produced sub-directories to the correct speaker identifier (i.e. the speaker's name or a unique id) and additionally remove any incorrectly diarized or mislabeled portions of audio. Finally, the user can prepare training, evaluation, and testing datasets (via the `expand_labeled_diarized_audio_dir_to_dataset` and `preprocess_dataset` functions) and fine-tune a new speaker identification model (via the `train` function).

The goal of this library is to simplify these proceses while maintaining fidelity in the training and utility of a speaker identification model for application.

Dataset Generation and Import

Diarization

Diarization is the unsupervised process of splitting audio into segments grouped by speaker identity. The output of a diarization model is usually a random ID (e.g. "speaker_0", "speaker_1", etc.) where there is no guarantee that "speaker_0" from a first audio file, is the same "speaker_0" from a second audio file. Because of this unsupervised nature, diarization cannot be used as a single solution for multi-speaker, multi-recording speaker identification. It can however be used for quickly generating large amounts of training examples which can be validated and labeled for use in a later speaker identification training set.

We make use of diarization as one method for preparing a speaker identification training set by using a model provided by `pyannote.audio` to diarize an audio file and place the unlabeled portions of audio into directories on the user's file system. A user can then listen to a few or all of the samples of audio in each directory, remove any samples that were mis-classified, and finally rename each of the directories with a true and consistent speaker identifier (e.g. a name, database ID, etc.).

Using Gecko Annotations

If a fully supervised method for dataset generation is preferred, or to improve model accuracy and improve coverage of edge cases, users of Speakerbox may use Gecko: a free web application for manual segmentation of audio files by speaker as well as annotation of the linguistic content of a conversation (Golan Levy, 2019). Speakerbox can make use of Gecko annotations as a method for training set creation by providing functions to split and prepare audio files using the annotations stored in a Gecko created JSON file.

Preparation for Model Training and Evaluation

To ensure that a Speakerbox model is learning the features of each speaker's voice (and not the features of the microphone or the specific words and phrases of each recording) we create dataset training, test, and evaluation subsets based off of a recording holdout and speaker stratification pattern. Each train, test, and evaluation subset must contain unique recording IDs to reduce the chance of learning the features of specific microphones or recording contexts, and each produced subset must contain recordings of every speaker available from the whole dataset. For example, if there are nine unique speakers in the complete dataset, then each train, test, and evaluation subset is required to have examples of all nine speakers.

If these recording holdout and speaker stratification conditions are not met, Speakerbox iteratively retries this random sampling process again. If the sampling function cannot find a valid dataset configuration given the sampling iterations, we inform the user of this failure and prompt them to add more examples to the dataset.

Model Fine-Tuning

The Speakerbox training process consists of fine-tuning a pre-trained speaker identification model (Yang et al., 2021) provided by Huggingface's Transformers library (Wolf et al., 2020). The default model for fine-tuning (superb/wav2vec2-base-superb-sid) was pre-trained on the VoxCeleb1 dataset (Nagrani et al., 2017).

As Speakerbox is a framework for fine-tuning a pre-trained speaker identification transformer, we will not cover the original evaluation of the model but instead, provide details about what we believe is a more typical use case for a fine-tuned speaker identification model.

Our example dataset contains 9 unique speakers across 10 unique recordings and each recording has some or all of the 9 unique speakers for a total of 1 hour of audio. For our use case in computational social science, this dataset represents a typical audio (and transcript) dataset created from government meetings, 1-on-1 interviews, group interviews, etc.

We further created random samples of this dataset with 15 minutes and 30 minutes of audio (each then split between the train, test, and evaluation subsets). The results reported in Table 1 are the mean accuracy, precision, and recall of five iterations of model training and evaluation using the differently sized datasets as inputs to our `train` and `eval_model` functions.

dataset_size	mean_accuracy	mean_precision	mean_recall	mean_training_duration_seconds
15-minutes	0.874 ± 0.029	0.881 ± 0.037	0.874 ± 0.029	101 ± 1
30-minutes	0.929 ± 0.006	0.94 ± 0.007	0.929 ± 0.006	186 ± 3
60-minutes	0.937 ± 0.02	0.94 ± 0.017	0.937 ± 0.02	453 ± 7

All results reported are the average of five model training and evaluation trials for each of the different dataset sizes. All models were fine-tuned using an NVIDIA GTX 1070 TI.

We provide a method to reproduce these models by downloading the example dataset unzipping its contents and then running:

```
from speakerbox.examples import train_and_eval_all_example_models

# Returns a pandas DataFrame
results = train_and_eval_all_example_models(
  "/your/path/to/the/unzipped/example-data/",
)
```

Usage in Existing Research

We are utilizing Speakerbox-trained models to annotate municipal council meeting transcripts provided by the Council Data Project (Brown et al., 2021). In our initial research, we first annotated ~10 hours of audio using the Gecko platform in ~12 hours of time, we then used our diarization and labeling method to annotate an additional ~21 hours of audio in ~6 hours of time. In total, the dataset was annotated and compiled in less than ~18 hours and contained ~31 hours of audio from meetings of the Seattle City Council (Eva & Weber, 2022). The model trained from the annotated dataset with the best precision and recall achieved 0.977 and 0.976 respectively. We additionally have used this model to annotate ~200 audio-aligned transcripts of Seattle City Council meetings and are now conducting analysis of speaker behaviors and group dynamics in such meetings.

Future Work

The few-shot approach we have described in Speakerbox can be extended, and made more accessible in future work. This could include:

1. **The creation of a GUI for dataset preparation, training, and application:** GUIs developed with Python have become much more accessible in recent years with tools such as MagicGUI being developed. A GUI would likely help non-computational scientists more easily train their own fine-tuned Speakerbox models.

2. **The creation of a template repository to assist in the "productization" of Speakerbox models:** The construction of a "template" or "cookiecutter" (Greenfeld et al., 2015) repository may also be useful. A template repository may provide a standard configuration for the storage and versioning of the training data and the management of model training and evaluation via continuous integration. We plan to create and embed a similar system within Council Data Project (CDP) infrastructures to enable the transparent training of speaker identification models specific to each municipal council. In our case, we plan to have a continuous integration system attempt to train a new model when new training data is added to the infrastructure repository. If the trained model reaches an accuracy threshold, we plan to store this model on our remote storage system. This model would then be used to annotate transcripts with their speakers during the processing of each municipal meeting.

Acknowledgements

We wish to thank the University of Washington Information School for support, and a grant from New America Ventures for the Puget Sound Public Interest Technology Clinic. We wish to thank all the past and present contributors of the Council Data Project.

References

Anguera, X., Bozonnet, S., Evans, N., Fredouille, C., Friedland, G., & Vinyals, O. (2012). Speaker diarization: A review of recent research. *IEEE Transactions on Audio, Speech, and Language Processing*, *20*(2), 356–370. https://doi.org/10.1109/TASL.2011.2125954

Bredin, H., & Laurent, A. (2021). End-to-end speaker segmentation for overlap-aware resegmentation. *Proc. Interspeech 2021*. https://doi.org/10.21437/interspeech.2021-560

Bredin, H., Yin, R., Coria, J. M., Gelly, G., Korshunov, P., Lavechin, M., Fustes, D., Titeux, H., Bouaziz, W., & Gill, M.-P. (2020). pyannote.audio: neural building blocks for speaker diarization. *ICASSP 2020, IEEE International Conference on Acoustics, Speech, and Signal Processing*.

Brown, E. M., Huynh, T., Na, I., Ledbetter, B., Ticehurst, H., Liu, S., Gilles, E., Greene, K. M. f., Cho, S., Ragoler, S., & Weber, N. (2021). Council Data Project: Software for Municipal Data Collection, Analysis, and Publication. *Journal of Open Source Software*, *6*(68), 3904. https://doi.org/10.21105/joss.03904

Eva, M. B., & Weber, N. (2022). Councils in action: Automating the curation of municipal governance data for research. *Proceedings of the Association for Information Science and Technology*, *59*(1), 23–31. https://doi.org/10.1002/pra2.601

Golan Levy, I. A., Raquel Sitman. (2019). GECKO - a tool for effective annotation of human conversations. *20th Annual Conference of the International Speech Communication Association, Interspeech 2019*. https://github.com/gong-io/gecko/blob/master/docs/gecko_interspeech_2019_paper.pdf

Greenfeld, A. R., Greenfeld, D. R., & Pierzina, R. (2015). cookiecutter. In *GitHub repository*. GitHub. https://github.com/cookiecutter/cookiecutter

Jacobi, T., & Schweers, D. (2017). *Justice, Interrupted: The Effect of Gender, Ideology and Seniority at Supreme Court Oral Arguments* [{SSRN} {Scholarly} {Paper}]. https://papers.ssrn.com/abstract=2933016

Kumar, N., Goel, S., Narang, A., & Lall, B. (2020). *Few shot adaptive normalization driven multi-speaker speech synthesis*. arXiv. https://doi.org/10.48550/ARXIV.2012.07252

Li, Y., Wang, W., Chen, H., Cao, W., Li, W., & He, Q. (2022). *Few-shot speaker identification using depthwise separable convolutional network with channel attention*. arXiv. https://doi.org/10.48550/ARXIV.2204.11180

Maltzman, F., & Sigelman, L. (1996). The Politics of Talk: Unconstrained Floor Time in the U.S. House of Representatives. *The Journal of Politics*, *58*(3), 819–830. https://doi.org/10.2307/2960448

Miller, M. G., & Sutherland, J. L. (2022). The Effect of Gender on Interruptions at Congressional Hearings. *American Political Science Review*, 1–19. https://doi.org/10.1017/S0003055422000260

Morris, J. S. (2001). Reexamining the Politics of Talk: Partisan Rhetoric in the 104th House. *Legislative Studies Quarterly*, *26*(1), 101–121. https://doi.org/10.2307/440405

Nagrani, A., Chung, J. S., & Zisserman, A. (2017). VoxCeleb: A large-scale speaker identification dataset. *INTERSPEECH*. https://doi.org/10.21437/interspeech.2017-950

Osborn, T., & Mendez, J. M. (2010). Speaking as Women: Women and Floor Speeches in the Senate. *Journal of Women, Politics & Policy*, *31*(1), 1–21. https://doi.org/10.1080/15544770903501384

Slapin, J. B., & Kirkland, J. H. (2020). The Sound of Rebellion: Voting Dissent and Legislative Speech in the UK House of Commons. *Legislative Studies Quarterly*, *45*(2), 153–176. https://doi.org/10.1111/lsq.12251

Torfi, A. (2018). SpeechPy - a library for speech processing and recognition. *Journal of Open Source Software*, *3*(27), 749. https://doi.org/10.21105/joss.00749

Tunstall, L., Reimers, N., Jo, U. E. S., Bates, L., Korat, D., Wasserblat, M., & Pereg, O. (2022). *Efficient few-shot learning without prompts*. arXiv. https://doi.org/10.48550/ARXIV.2209.11055

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *30*.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., Platen, P. von, Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., … Rush, A. M. (2020). Transformers: State-of-the-art natural language processing. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 38–45. https://www.aclweb.org/anthology/2020.emnlp-demos.6

Wolters, P., Careaga, C., Hutchinson, B., & Phillips, L. A. (2020). A study of few-shot audio classification. *ArXiv*, *abs/2012.01573*.

Yang, S., Chi, P.-H., Chuang, Y.-S., Lai, C.-I. J., Lakhotia, K., Lin, Y. Y., Liu, A. T., Shi, J., Chang, X., Lin, G.-T., & others. (2021). SUPERB: Speech processing universal PERformance benchmark. *arXiv Preprint arXiv:2105.01051*. https://doi.org/10.21437/interspeech.2021-1775