# Nkululeko: A Tool For Rapid Speaker Characteristics Detection

**Felix Burkhardt**[1,2], **Johannes Wagner**[1], **Hagen Wierstorf**[1], **Florian Eyben**[1], **Björn Schuller**[1,3,4]

[1]audEERING GmbH, Germany, [2]Technical University of Berlin, Germany,
[3]Chair EIHW, Universität Augsburg, Germany,
[4]GLAM, Imperial College London, UK
{fburkhardt, jwagner, hwierstorf, fe, bs}@audeering.com

## Abstract

We present advancements with a software tool called Nkululeko, that lets users perform (semi-) supervised machine learning experiments in the speaker characteristics domain. It is based on audformat, a format for speech database metadata description. Due to an interface based on configurable templates, it supports best practise and very fast setup of experiments without the need to be proficient in the underlying language: Python. The paper explains the handling of Nkululeko and presents two typical experiments: comparing the expert acoustic features with artificial neural net embeddings for emotion classification and speaker age regression.

**Keywords:** tools, machine learning, speaker characteristics

## 1. Introduction and Related Work

In the past decades, the research community has been confronted with the tremendous success of approaches to estimate knowledge with artificial neural nets (ANN), predominantly under the label *deep learning* (DL). Especially empirical sciences benefit from the opportunity to test hypotheses with machine learning experiments that are able to analyse statistically very large data quantities.

Many empirical researchers, phoneticians, and linguists, did not study computer science and struggle with the necessary programming skills to set machine learning experiments up.

Nkululeko[1] is being developed preliminary as a tool for a series of machine learning seminars at the institute for speech communication at the Technical University of Berlin to enable students to conduct machine learning experiments with a very flat learning curve by simply filling configuration files.

This makes it much easier to be used compared to other high level frameworks for deep learning like Keras, Torch, Google AutoML, or end2you (Chollet, 2017; Chaudhary et al., 2020; Tzirakis et al., 2018; Bisong, 2019) while still keeping the flexibility as it is based on Torch and Keras.

There are other frameworks that let users do data processing without the need to program too much code. Knime[2] is probably the most prominent one: data processing chains can be specified by connecting predefined modules, many of them machine learning models. In contrast to Nkululeko, Knime is a rather general data exploration software and not focused on audio processing and thus misses the acoustic feature extraction components as well as making it harder to set up specific experiments.

In (Evain et al., 2021), a framework named Lebenchmark is described, that enables users to perform self-supervised learning (SSL) experiments with pretrained models for French emotional speech. Although also meant to be used by the public, this framework is not meant to be a general tool for speech related machine learning experiments.

This article is structured as follows: in Section 2 we start with an overview of the software and its features. Section 3 introduces the audformat which is the main basis to interface databases with the tool. The following Section 4 explains the general possibilities to configure experiments with Nkululeko and the format of the configuration file. In Section 5, we introduce prototypical experiments that have been conducted utilising Nkululeko, specifically for speaker emotion (as classification problem) and speaker age (as regression problem), by comparing the performance of an expert acoustic feature set with pre-trained ANN embeddings. We conclude with Section 6 and give a brief outlook.

Contributions of this paper are as follows:

- We present a software that can be used to run acoustic machine learning experiments out of the box.

- As it is targeted solely on audio databases, it is easier to be set up than comparable approaches to enlarge knowledge in this domain.

## 2. Overview of Nkululeko

Nkululeko is open source software written in Python and hosted on github[3].

The main features are: training and evaluation of labeled speech databases with state of the art machine

---

[1]On the lookout for a distinctive name for this project we stumbled across an 1980ies punk album title. They tried new things out quickly, so this seemed fitting.

[2]https://www.knime.com/
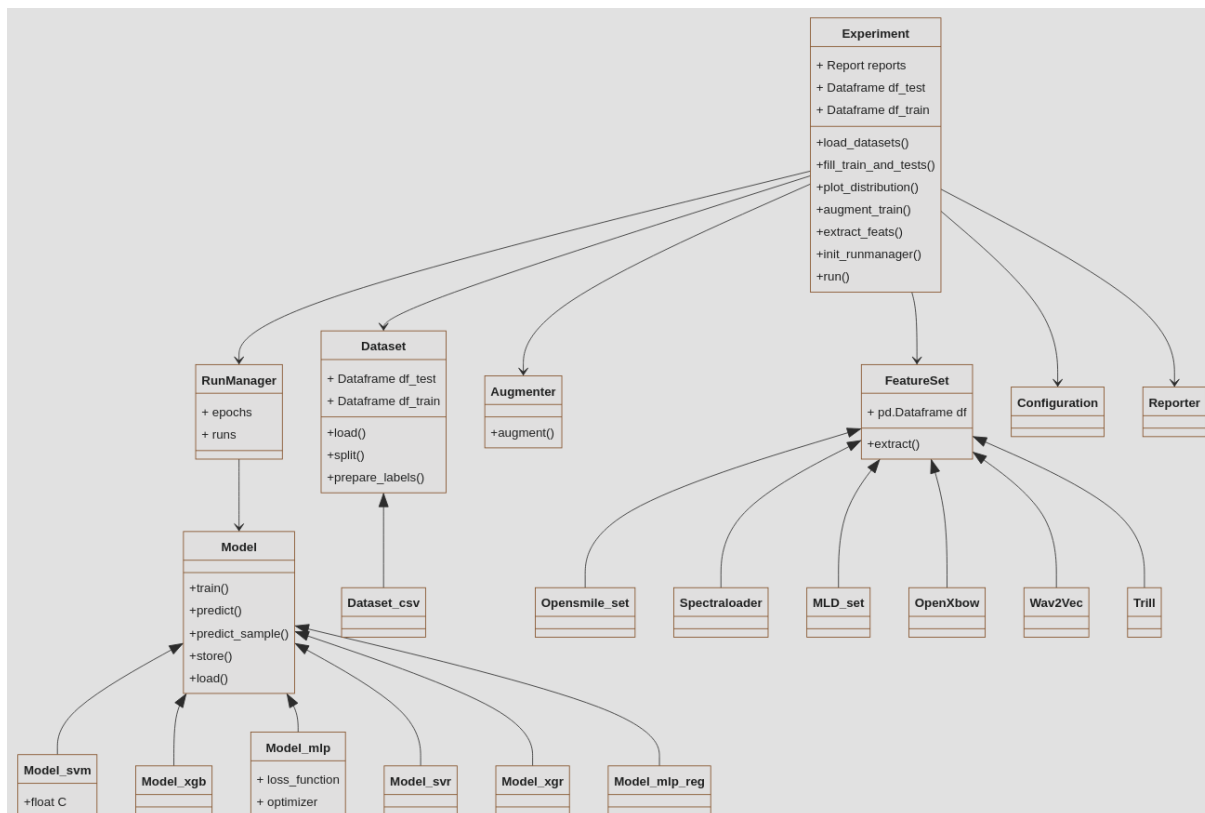
[3]https://github.com/felixbur/nkululeko/

Figure 1: Class architecture of Nkululeko.

learning approaches and acoustic features extractors, a live demonstration interface, and the possibility to store databases with predicted (aka "weak") labels for semi-supervised learning.

The data management is based on audformat, as explained in Section 3, but a simpler CSV formalism is also supported. Basically, to be used by Nkululeko, the data format should include the audiofile path, a speaker id, a sex label, and a task specific label. Here is an example for a database labelled with emotion:

```
x/sample.wav, s1, female, happy
...
```

or with age:

```
x/sample.wav, roger, male, 45
...
```

In Figure 1, a broad overview of the Nkululeko architecture is given. It consists of a central "experiment" class which can combine a set of acoustic features with machine learning classifiers and regressors. The specific feature sets are described in Section 4.3 and the classifiers/regressors (aka "models") are detailed in Section 4.4.

A central "Runmanager" class is responsible for actually running the experiments for several runs (repetitions) and epochs (updates of the trained model). When done, a "Reporter" is delivered that provides for textual and visual output.

All the parameters of an experiment are stored in the global "Config" class as well as utility functions used throughout the code and the label encodings.

A special "Scaler" class provides for several scaling methods, like z-transformation or speaker normalisation.

These classes can be used in own code, but it is generally not necessary for users to program in Python, as they can simply call ready-made python scripts that are provided with the distribution. The users then describe the nature of their experiments entirely in the configuration file, as described in Section 4.

## 3. The audformat Package

*audformat*[4] defines an open format for storing media data, such as audio or video, together with corresponding annotations. The format was designed to be universal enough to be applicable to as many use cases as possible, yet simple enough to be understood easily by a human and parsed efficiently by a machine.

A database in *audformat* consists of a header, which stores information about the database (source, author, language, etc.), the type of media (sampling rate, bit depth, etc.), the raters (age, mother tongue, etc.), the schemes (numerical, categories, text, etc.), and the splits (train, test, etc.). It also keeps reference of all

---

[4] https://audeering.github.io/audformat/

tables that belong to the database, which hold the actual annotations and are stored in separate files in text and/or binary format.

A corresponding Python implementation[5] provides tools to access the data, create statistics, merge annotations, and search / filter information.

# 4. The Configuration File

An experiment in Nkululeko is specified by a configuration file. The distribution contains several examples and a document that explains all the options that are currently available with Nkululeko[6], but we will discuss the central aspects here.

There is also a blog series on the usage of Nkululeko[7]. The configuration file consists of a set of key-value pairs that are organised into four sections, which are discussed in the following subsections. Almost all keys have default values so they do not have to be specified.

## 4.1. EXP (Experiment) Section

This is a global section to set up the root experiment folder, specify the number of runs (to average over diverging results per random initialisation) and epochs, and generally distinguish classification from regression experiments.

Additionally, the folder names to store results in textual and visual form, trained models, and database split sets for training and evaluation, are specified here.

## 4.2. DATA (Databases) Section

This section deals with the database management. The most important parameters are the paths to the databases that should be used in the experiment. Additionally, one can specify whether predefined train and test sets should be used, or if the test set should be randomised with a disjunct speaker set, or if whole databases should be used as training versus test set. Of course, partitions of databases can also be selected. Here is a sample listing of a database section:

```
databases = ['emodb', 'emovo']
strategy = cross_data
emodb = /data/audb/emodb/
emodb.split_strategy = speaker_split
emodb.testsplit = 40
emodb.mapping =
   {'anger':'angry', 'joy':'happy',...}
emovo = /data/audb/emovo/
tests = ['emodb']
trains = ['emovo']
target = emotion
labels = ['angry', 'happy', 'neutral']
```

As can be seen, some of the values simply contain Python data structures like arrays or dictionaries.

Within this example, an experiment is specified to use 40 % of the speakers of the German database "emodb" as a test set and the whole Italian database "emovo" as a training set. To map the category labels, a mapping strategy is provided. For this specific experiment, not all categories (most acted databases contain five to seven emotion categories) are used, but only a subset: *angry*, *happy*, and *neutral*.

Furthermore, pre-defined test and training splits (see audformat, Section 3) can be freely assigned as test and evaluation sets. Nkululeko outputs overview plots like the ones displayed in Figure 2.

For continuous data, the DATA section might include the following keys:

```
...
type = continuous
labels = ['20ies', '30ies', '40ies',
   '50ies', '60ies', '70ies', '80ies']
bins  = [-1000, 30, 40, 50, 60,
   70, 80, 10000]
```

If *type* is set to "continuous" (while the experiment type is "classification"), even numerical data can be used with classifier algorithms (Section 4.4) as it is binned prior to being processed further. The *labels* and *bins* keys are always required for continuous data, to plot the confusion matrices (see Section 4.5).

## 4.3. FEATS (Acoustic Features) Section

Within the feature section, the kind of acoustic parameters that should be extracted from the audio files are specified. A typical section might look like this.

```
type = xbow
with_os = True
size = 1000
assignments = 10
needs_feature_extraction = True
scale = standard
```

The most important key is "type", because here, the basic type of feature extractor is specified. The example above states several xbow specific parameters (see below: *with_os*, *size* and *assignments*). *needs_feature_extraction* is a flag to tell Nkululeko that the features should be extracted even if already present, as the default is to re-use features. *scale* turns on feature scaling, and *standard* means standard normalisation, an alternative would be *speaker* to denote speaker scaling of the features.

We distinguish several kinds of feature types:

- **expert features**: A usually smaller set of hand-picked acoustic features like the 88 eGeMAPS (Eyben et al., 2015) features or mid level descriptors (Reichel et al., 2020).

---

- **brute force features**: A very large set of acoustic features combining frame-based values with functionals, like the 6,373 features of the Compare 16 challenge (Schuller et al., 2016). Of course, a features selection step should follow. Mel spectrograms could either be categorised here or as an input for the next category: learnt features.

- **learnt features** / embeddings: A way to do transfer learning with deep artificial neural networks. A large network gets trained with many hours of speech data and one of the deeper layers is then used as an abstract representation of the raw audio samples. Other deep learning techniques, like variational autoencoders (VAEs) or generative adversarial networks (GANs) that reduce dimensionality, also would fall into this category, but are currently not yet interfaced with Nkululeko.

Available values for the *type* key are:

- **os**: openSMILE (Eyben et al., 2010) feature sets. They are based on frame-based low level descriptors such as $F_0$ combined by statistical functionals. The open source Python version of openSMILE[8] is being used and all available features sets can be specified. The default is the eGeMAPS set (Eyben et al., 2015), an expert set of 88 acoustic features. These features are being used in numerous articles in the literature as baseline features (Ringeval et al., 2018; Schuller et al., 2016; Burkhardt et al., 2021), as they work reasonably well with many tasks and are easy to handle for most classifiers based on their small number.

- **spectra**: A feature extractor that results in Mel spectrograms per speech file. This input transforms speech into images, and is often used by Convolutional Neural Networks.

- **mld**: Mid level descriptors as described in (Reichel et al., 2020). Based on openSMILE low level descriptors, a syllabification is performed, and features describe suprasegmentals.

- **xbow**: openXBOW[9] the Passau Open-Source Crossmodal Bag-of-Words Toolkit (Schmitt and Schuller, 2017), an approach to use Bag-of-words techniques, known from natural language processing (NLP) that counts basic elements, for audio processing. The audio words are based in this case again on openSMILE low level descriptors.

- **trill** TRILL embeddings (Shor et al., 2020), meaning the penultimate layer of a deep neural network that has been trained with a triplet loss that distinguishes frames from the same speech file by

those from different ones. TRILL has been trained on the speech tagged samples of the Audio Set database (Gemmeke et al., 2017), which is about 2,793 hours of audio data and has been published by Google. TRILL embeddings perform quite well for emotion classification, as shown in Section 5.2.

- **wav2vec**: Wav2Vec 2.0 (Baevski et al., 2020) embeddings also use the penultimate layer of a pretrained deep neural network. This one has been trained by an approach to adopt the idea of the (again) NLP domain of word embeddings, that model semantics by aligning words that occur in similar environments, to the audio domain. Although Wav2Vec primarily targets speech recognition (ASR), it is well suited to model speaker characteristics as well, as shown in Section 5.1. There are several pre-trained versions that have been published by Facebook.

All models needed for feature/embeddings extraction have to be downloaded and installed from their distributors and the paths made known to Nkululeko. There is a series of tutorials linked from the Nkululeko github README how to do this.

## 4.4. MODEL (Classifier/Regressor Meta Parameters) Section

The MODEL section describes the classifier or regressor that should be used in the experiment. A typical section would look like this:

```
type = mlp
layers = {'l1':1024, 'l2':32}
loss_function = 1-ccc
learning_rate = 0.001
optimizer = adam
device = cpu
```

This snippet specifies to use a a multi layer perceptron (MLP (Haykin, 1994)).

The *layers* key specifies the number and size of hidden layers – it is the only key that has no default. The *loss function* is the most important parameter for an ANN, as it specifies the way the net learns by computing a distance between prediction and ground truth. It defaults to "mean square error" (MSE), but in the example, the concordance correlation coefficient (CCC) (Lin, 1989) is used, which works well for regression problems and has been implemented within Nkululeko.

The *learning rate* is a weight factor for the updates during training and the *optimizer* specifies the algorithm to enhance the gradient descent. Currently, the defaults of the Torch framework are being used. *device* denotes the processor on the machine, either "cpu" or some "gpu" instance.

Here is an example of a model section using a XGBoost classifier (Chen and Guestrin, 2016). This classifier is basically a very sophisticated algorithm based

---

[8] https://github.com/audeering/opensmile-python

[9] https://github.com/openXBOW/openXBOW

on classification trees and has been working quite well in many of our experiments (Burkhardt et al., 2021); hence, we used it as a baseline in the experiments described in Section 5.

```
type = xgb
tuning_params =
  ['subsample', 'n_estimators',
  'max_depth']
scoring = recall_macro
subsample = [.5, .7]
n_estimators = [50, 80, 200]
max_depth = [1, 6]
class_weight = 1
```

If the *tuning_params* key is filled, Nkululeko will perform a five fold cross optimisation of the specified classifier or regressor parameters (using the underlying Scikit-learn functionality (Pedregosa et al., 2011)). The individual tuning parameter options must then be specified as own keys, like shown in the example. *class_weight* turns on the Scikit-learn class weight optimisation for unbalanced data by weighting sparse categories higher than frequent ones.

Following, one finds a brief description of the possible types of models that are currently implemented:

- **svm**: Support Vector Machine (Cortes and Vapnik, 1995). A very well known algorithm motivated by subdividing meta planes in the feature space. We use (again) the Scikit-learn implementation.

- **svr**: Support Vector Regression. Like SVM, but for continuous labels.

- **xgb**: XG-Boost. As described above.

- **xgr**: XG-Regression. The regressor version of XG-Boost.

- **mlp**: Multi-Layer-Perceptron for classification. As described above.

- **mlp reg**: Multi-Layer-Perceptron for regression.

- **cnn**: Convolutional neural network (Lecun and Bengio, ). A different approach to deep neural nets that utilises kernel functions from the computer vision domain to drastically reduce the size of the hidden layers and model local phenomena – not yet included in the open source version.

### 4.5. PLOT (Plotting Options) Section

One of the most important issues in data processing is visualisation. Within Nkululeko, therefore, plots have an own section in the configuration. Here are some possibly keys for the section:

- **plot epochs**: A flag to activate confusion matrix plotting (like the ones shown in Figures 4 and 5) for each epoch. The ultimate result will always be
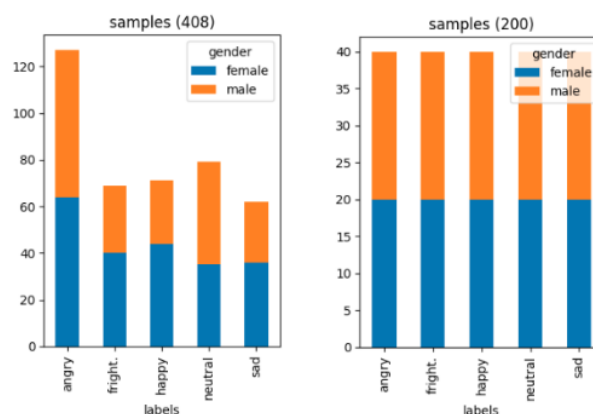


Figure 2: Distribution of categories and speaker sex for training (EmoDB, left) and test (Polish, right) samples for the acted emotions classification experiment
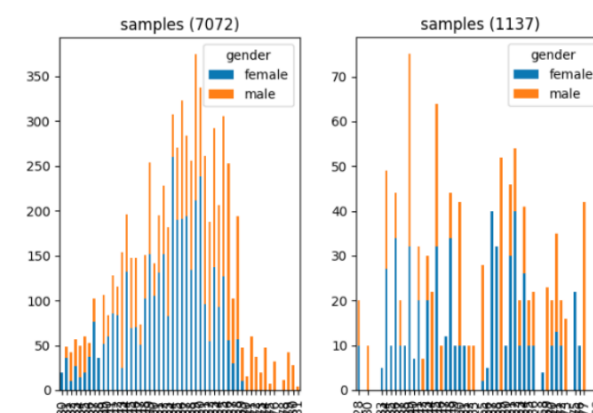


Figure 3: Distribution of speaker age for training (left) and test (right) samples for the age regression experiment

plotted. Confusion matrices are also plotted for continuous data / regression problems by binning the data according to the bins stated in the DATA section (see Section 4.2).

- **plot anim progression**: Generate an animated gif from the epoch plots to visualise the progress while the training updates the weights. The frame rate can be specified as well.

- **plot epoch progression**: Plot the progression of test, train, and loss results over epochs, like shown in Figures 4 and 5, right hand side.

- **plot best model**: Search for the best performing model and plot its confusion matrix. All results are also stored in textual form, and the best results per run can be compared.

- **value counts**: Plot class and sex distribution statistics for each database and the training / evaluation splits. Examples are Figure 2 and 3.

- **tsne**: Generate a t-distributed stochastic neighbor embedding (t-SNE) plot (van der Maaten and Hinton, 2008). This is sometimes a very good way to asses whether the features can explain the labels at all: reduce the dimensionality of the features to two and colour the data dots according to category.

## 5. Experiment Conducted with Nkululeko

To illustrate typical use of Nkululeko, we describe here two experiments. The first one deals with classification of prototypical emotional expression, the other one with regression of age in years.

### 5.1. Cross Database Classification of Prototypical Emotional Expression

For this experiment we used two comparable databases from the literature: EmoDB (Burkhardt et al., 2005) and the Polish Emotional Database (Powroźnik, 2017). Both include samples from actors portraying basic emotions with uniform text material of non-emotional semantics. An extended version of this experiment has been reported in (Burkhardt et al., 2022).

To demonstrate the abilities of Nkululeko, we contrast two runs, one using a linear classifier with expert features, and one using an ANN with learnt features. As a cross database experiment, we chose Polish to be the test set (Figure 2, right hand side), as it is the smaller database (8 speakers), and EmoDB (German, Figure 2, left hand side) as the training data set (10 speakers).

The two approaches considered are:

- A) Using XG-Boost classifier (see Section 4.4) and the eGeMAPS acoustic feature set (see Section 4.3).

- B) Using an MLP classifier (see Section 4.4) and the Wav2vec embeddings as features (see Section 4.3). In Nkululeko, the user has to specify the path to a local Wav2Vec 2.0 model. We used the one accessible at hugginface[10] that has been pre trained with 300 hours of speech data from Libri Speech (Panayotov et al., 2015), Common Voice (Ardila et al., 2020), and Switchboard (Godfrey et al., 1992).

The results are shown in Figure 4 as confusion matrices. Both approaches result in an unweighted average recall (UAR) clearly above chance level (equals .2, as the classes are evenly distributed). As expected, the approach B) that uses transfer learning (see Section 4.3 for the discussion on Wav2vec) performs clearly better than the approach using eGeMAPS features. A serious comparison could have used the same classifier, but this experiment mainly shall demonstrate Nkululeko features. As can be seen in the epoch progression (right hand in Figure 4), the model does not need 1000 epochs

to converge, but sometimes a chance weight configuration in the ANN leads to UARs above average. It is to be suspected that this overfits on the specific test set. Nkululeko prints the UAR per confusion plot on top of the figure as seen in Figure 4. The result for the MLP trained (this can be seen as a kind of fine-tuning actually) with Wav2Vec 2.0 embeddings is with .52 10 % over the baseline (.42).

### 5.2. Comparing Expert Features and Embeddings for Age Regression

To demonstrate a regression experiment, we describe here a second investigation. The data in this case is a small set of speeches from the German parliament[11]. The speeches of 593 politicians were segmented with a voice activity detection (VAD) approach. This results into short speech samples of about 3-5 seconds length. About 20 samples per speaker were then collected for a database. The age span is between 29 and 81, but very old and very young persons are sparse. We selected an evaluation split for the age groups so that the decades are nearly balanced. The rest of the data has been used for training, the corresponding distributions are shown in Figure 3.

We performed again two experiments using different approaches:

- A) Using XGR (XG-Boost regression, see Section 4.4) and the eGeMAPS acoustic feature set (see Section 4.3) is being used as baseline.

- B) Using a MLP classifier (see Section 4.4) and the TRILL embeddings as features (see Section 4.3).

The results are shown in Figure 5. As stated above, machine learning results in Nkululeko are always visualised as confusion matrices, even when the data is continuous. In this case, we binned the data to the age decades that are present in our data.

Table 1 displays the CCC and Pearson's correlation coefficient (PCC) results. As confirmed by the confusion matrices, the performance more than doubled with the MLP using pre trained TRILL embeddings. In the right hand side of Figure 5), the progression of loss, training, and evaluation set performance per epoch is shown. As the ANN overfitted fastly in a first version, we set the learning rate to 0.00001 for this experiment. Of course, a drop-out layer would also have been an option and is envisaged to be integrated with Nkululeko. As can be seen, the ANN still starts to overfit after the 20th epoch.

## 6. Conclusions and Outlook

We presented Nkululeko – a free for research new open-source tool to set up machine learning experiments in the speech research domain that can be used without programming skills. Future works will include

---

[10]https://huggingface.co/facebook/wav2vec2-large-robust-ft-swbd-300h

[11]https://www.bundestag.de/parlament/praesidium/reden

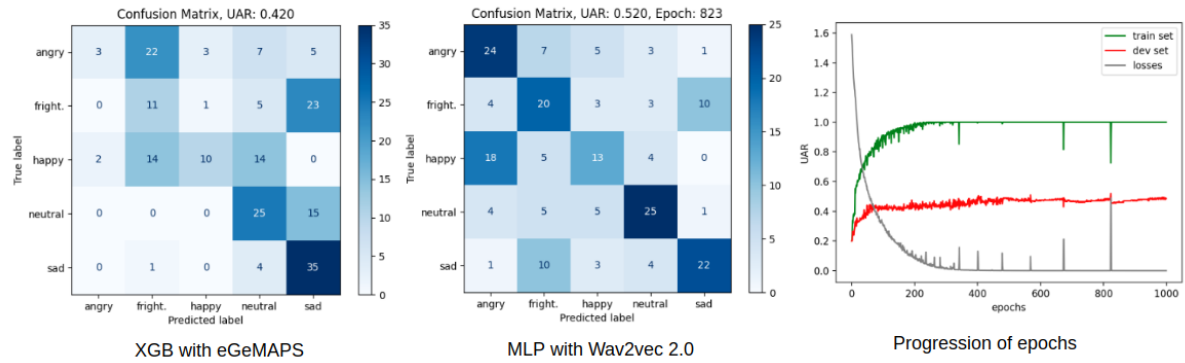| XGB with eGeMAPS | MLP with Wav2vec 2.0 | Progression of epochs |

Figure 4: Confusion matrices for an XGB classifier with eGeMAPS openSMILE features and an MLP classifier using Wav2Vec 2.0 embeddings. The training set is a German acted database, the evaluation set a Polish one. Right figure: evolution of performance during epochs for the MLP classifier.



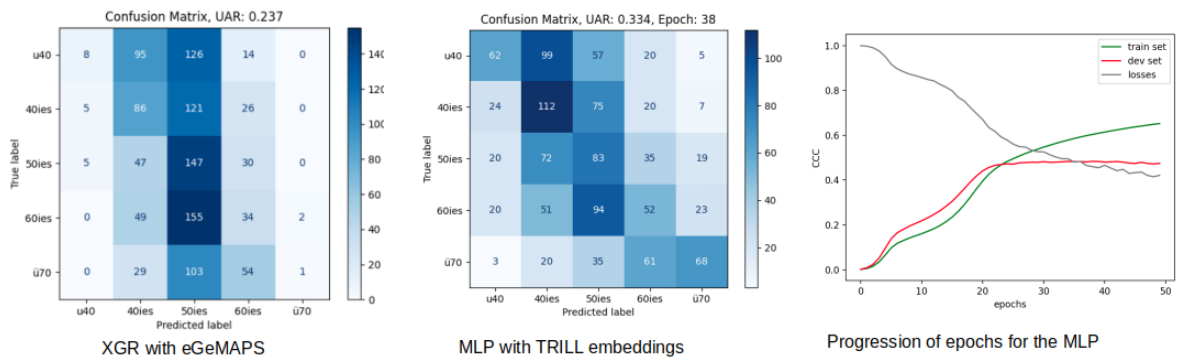| XGR with eGeMAPS | MLP with TRILL embeddings | Progression of epochs for the MLP |

Figure 5: Confusion matrices for an XGR regressor with eGeMAPS openSMILE features and a MLP regressor using TRILL embeddings (binned to categories: u40 means under forty, o70 over seventy). Test and training sets are speaker disjunct random (short) samples of German parliament speeches. Right figure: evolution of performance during epochs for the MLP classifier.

| | XGR w. eGeMAPS | MLP w. TRILL |
|---|---|---|
| CCC | .223 | **.483** |
| PCC | .272 | **.463** |

Table 1: Results of the age regression experiment

extension of its functionality. Especially, we will add open source modules for Mel spectrogram extraction and convolutional neural net models.

## 7. Acknowledgments

## 8. Bibliographical References

Ardila, R., Branson, M., Davis, K., Henretty, M., Kohler, M., Meyer, J., Morais, R., Saunders, L., Tyers, F. M., and Weber, G. (2020). Common voice: A massively-multilingual speech corpus.

Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations. In H. Larochelle, et al., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460. Curran Associates, Inc.

Bisong, E., (2019). *Google AutoML: Cloud Vision*, pages 581–598. 09.

Burkhardt, F., Paeschke, A., Rolfes, M., Sendlmeier, W., and Weiss, B. (2005). A database of german emotional speech. volume 5, pages 1517–1520, 09.

Burkhardt, F., Brückl, M., and Schuller, B. (2021). Age classification: Comparison of human vs machine performance in prompted and spontaneous speech. In Stefan Hillmann, et al., editors, *Studientexte zur Sprachkommunikation Elektronische Sprachsignalverarbeitung 2021*. TUDpress, Dresden, 03.

Burkhardt, F., Hacker, A., Reichel, U., Wierstorf, H., Eyben, F., and Schuller, B. (2022). A comparative cross language view on acted databases portraying basic emotions utilising machine learning. In *Proceedings of LREC 2022*.

Chaudhary, A., Chouhan, K. S., Gajrani, J., and

Sharma, B. (2020). Deep learning with pytorch.

Chen, T. and Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA. ACM.

Chollet, F. (2017). Deep learning with python & keras.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.

Evain, S., Nguyen, H., Le, H., Boito, M. Z., Mdhaffar, S., Alisamir, S., Tong, Z., Tomashenko, N., Dinarelli, M., Parcollet, T., Allauzen, A., Estève, Y., Lecouteux, B., Portet, F., Rossato, S., Ringeval, F., Schwab, D., and laurent besacier. (2021). Task agnostic and task specific self-supervised learning from speech with lebenchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Eyben, F., Wöllmer, M., and Schuller, B. (2010). opensmile – the munich versatile and fast open-source audio feature extractor. *MM'10 - Proceedings of the ACM Multimedia 2010 International Conference*, pages 1459–1462, 01.

Eyben, F., Scherer, K., Schuller, B., Sundberg, J., Andre, E., Busso, C., Devillers, L., Epps, J., Laukka, P., Narayanan, S., and Truong, K. (2015). The geneva minimalistic acoustic parameter set (gemaps) for voice research and affective computing. *IEEE Transactions on Affective Computing*, 7:1–1, 01.

Gemmeke, J. F., Ellis, D. P. W., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., Plakal, M., and Ritter, M. (2017). Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 776–780.

Godfrey, J., Holliman, E., and McDaniel, J. (1992). Switchboard: telephone speech corpus for research and development. In *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 517–520 vol.1.

Haykin, S. (1994). *Neural networks: a comprehensive foundation*. Prentice Hall PTR.

Lecun, Y. and Bengio, Y., ).

Lin, L. I.-K. (1989). A concordance correlation coefficient to evaluate reproducibility. *Biometrics*, 45.

Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. (2015). Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Powroźnik, P. (2017). Kohonen network as a classifier of polish emotional speech. *ITM Web of Conferences*, 15.

Reichel, U., Triantafyllopoulos, A., Oates, C., Huber, S., and Schuller, B. (2020). Spoken language identification by means of acoustic mid-level descriptors. 03.

Ringeval, F., Cowie, R., Amiriparian, S., Michaud, A., Schuller, B., Kaya, H., Cummins, N., Çiftçi, E., Valstar, M., Schmitt, M., Lalanne, D., Güleç, H., Salah, A. A., and Pantic, M. (2018). AVEC 2018 Workshop and Challenge: Bipolar disorder and cross-cultural affect recognition. In *AVEC 2018 - Proceedings of the 2018 Audio/Visual Emotion Challenge and Workshop, co-located with MM 2018*.

Schmitt, M. and Schuller, B. (2017). openxbow - introducing the passau open-source crossmodal bag-of-words toolkit. *Journal of Machine Learning Research*, 18:1–5, 10.

Schuller, B., Steidl, S., Batliner, A., Hirschberg, J., Burgoon, J., Baird, A., Elkins, A., Zhang, Y., Coutinho, E., and Evanini, K. (2016). The interspeech 2016 computational paralinguistics challenge: Deception, sincerity and native language. pages 2001–2005, 09.

Shor, J., Jansen, A., Maor, R., Lang, O., Quitry, F., Tagliasacchi, M., Tuval, O., Shavitt, I., Emanuel, D., and Haviv, Y. (2020). Towards learning a universal non-semantic representation of speech, 02.

Tzirakis, P., Zafeiriou, S., and Schuller, B. W. (2018). End2you–the imperial toolkit for multimodal profiling by end-to-end learning. *arXiv preprint arXiv:1802.01115*.

van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.