Human Computer Interaction Improvement by Interjection Recognition: A new speech processing task and dataset

Yaniv Goren, Alexander Apartsin, Yehudit Aperstein

Intelligent Systems department, Afeka Tel Aviv Academic College of Engineering

**Abstract.** One of the main challenges ofsuccessful human-computer interaction is to enable a natural and spontaneous dialogue between human and machine such as human-to-human dialogue.Although the use of interjections (e.g. “mmm”, “hmm”) convey important information in colloquial speech, they are usually considered to be “non-words” by Automatic Speech Recognition (ASR) engines. Recognizing and supporting interjection in speech-based user interfaces (e.g. voice control) could result in much more natural human-machine interactions. Moreover, interjection recognition can be utilities for Speech Emotion Recognition (SER) in emergency call center or call centers services. In this work, we present a dataset of interjection audio records collected for the task of interjection recognition. The collected dataset is used to train and evaluate two baseline neural networks models on inter-speaker and intra speaker interjection classification. To improve the performance the collected dataset was extended using a mix of several augmentation techniques (e.g. tempo and pitch transformation). We show that the training based on the augmented dataset results in a significant improvement in the classification accuracy and reduces the need for large amount of records for training the models.

**Keywords:** Interjection, Human Computer Interaction, Data Augmentation.

1. Introduction

People tend to choose an efficient form of verbal communications by leveraging common understanding of context between two parties. Interjections is one of the major parts of speech that is frequently used to convey meaning in a specific context. Many smart systems include a speech recognition system that enables a natural dialogue between man and machine [1]. Among well-known example are virtual personal assistant including Apple's artificial intelligence System Siri [2], or Amazon’s Alexa [3]. A user of Siri or Alexa can use natural speech queries for obtaining the information and performing various actions in several domains (e.g. check weather or stock prices, order pizza, etc.).

In recent years, we are in the midst of a revolution in communications that expands from a basic mode of people-to-people communications to include people to-machines communications. In order to succeed, this revolution demands a high-quality interface to support successful Human Computer Interactions (HCI) [4]. However, to create successful speech-enabled applications those systems must overcome the limitations of both current speech technologies and human cognitive processing. The challenge is not only advancing the performance of the technologies, but also understand how to integrate these technologies into viable, easy-to-use spoken language systems.

One of the limitations in HCI is that although spontaneous conversation is optimized for human-human interactions, it different from the types of speech for which human language technology is often developed. Adding the interjection recognition capabilities to the voice assistants will allow to improve the human computer interactions and increasing the usage with spontaneous conversation in human machine interfaces. The improvement can be expressed by interpreting the meaning of a hard-to-understand speech, such as heavy accent, or a sentence in which not all words are clear [31]. By understanding prominent keywords, we can understand the whole sentence. For example, when a human says “Oy, …” in a conversation he wants to express an unexpected situation that can be a fright from something.

Furthermore, expressing a spontaneous feeling is one of the key features of most interjections and it can be utilities for Speech Emotion Recognition (SER) [40]. While humans can efficiently recognize the emotional aspects of speech, this ability in machines is still an ongoing subject of research. Adding the ability to understand emotions to machine can provide efficient methods of detecting the emotions in different call centers services, emergency call center, and many other human-machine communication users. The capability of machines to detect emotions and act accordingly is a critical factor of making machines appear and act in a human-like manner.

In another sense, voice interjections can be considered as a sort of “voice touch” signals. Similarly, to immensely successful haptic touch interfaces, the interjection-based voice touch interfaces might allow effortless human machine interactions with voice assistants and other voice-enabled devices. For some applications, tasks can be accomplished successfully by interjection identified and then mapped it to an appropriate action or response. The interjection can be considered as a shortcut to a repeated action that requires more effort from the user and releases him from saying the request from the machine in a detailed way. For example, consider a user that used to request from his personal assistance “What time is the next train to King's Cross from my closest station? “, or “How long would it take me to drive home? “. Or another user that every time he got in the car after a day's work, used to set his GPS driving home and right after that to call home to his wife. By letting the user, the ability to pick an interjection phrase from a set of interjections and map it to a desire actions, interactions may be facilitated by customizing and adapting the system to the needs and preferences of individual users.

It is important to emphasize that our goal is not to detects certain interjection phrase within the speech, although a considerable part of interjections have a semantic meaning by themselves without the context of a conversation. For example, “A-ha” is a consent regardless of the context, and “wow” express impression or astonished. More so, the number of phrases considered as interjections are very big and we are not trying to build a complete system that can recognize all interjections. The purpose of this document is to expose a system that will form as a benchmark interjection identifier system which will be a solid foundation for adding new interjections quickly and simply. Our motivation is to enrich user interface technologies that enable system designers to create habitable human machine interfaces and dialogues which maintain natural interactions with the machine.

The collection and the preparation of the training data is a major challenge for this project. The available speech datasets are focusing on either word level [5], phoneme level [6] or event level [7] tasks, while the interjections lie somewhere in-between. Therefore, we collected our own unique baseline dataset by recording relatively large set of interjections and negative examples from some speakers.

In this work we propose a neural network models for interjection recognition and classification. We collected datasets and use them to train, evaluate and test the models. An interjection recognizer accepts a waveform features and return K+1 labels where K is the number of supported interjection classes. The additional class label is reserved for non-interjection audio input. It is known that deep learning requires a large amount of data to train an accurate model. To increase the amount of training data and reduce overﬁtting, we enriched the dataset by augmenting the original data through applying various artificial distortions [8]. The data augmentation includes the addition of background noise [9], pitch and tempo modifications [10].

In this paper, we present results of several different interjection recognition baseline experiments, where relative improvement obtained by using the proposed data augmentation methods over a state of the art Feedforward Neural Network (FNN) and Recurrent Neural Networks (RNN) classification models. Our data augmentation methods have been implemented on the training data. By training the network on the additional deformed data, the network becomes invariant to these deformations and generalizes better to unseen data.

The rest of the paper is organized as follows. In section 2 a previous work is reviewed with an emphasis on speech recognition, augmentation methods in speech recognition and interjections-related research. Section 3 dedicated to an explanation of our contribution. In section 4 we describe the dataset creation process, and description of our data augmentation tool followed by the interjection recognition models. This is followed by the experimental setup in section 5. In section 6 we discuss the results, and a summary of the proposed method with some possible research directions.

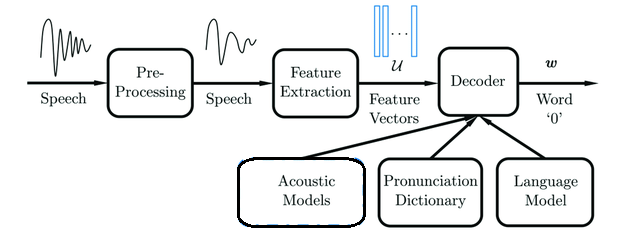
1. Previous Work

Processing and understanding speech and voice signals are studied in several contexts. One of the most common tasks associated with the processing speech and voice signals is Automatic Speech Recognition (ASR) [11].

ASR is a task of translating audio signal into text. The main challenge is to overcome the non-stationarity of speech signal and the large variations in its spatiotemporal representation. As illustrates in Figure 1, typical ASR system usually including several steps. The pre-processing step, where an analog speech signal is transformed into a digital signal, includes speech/non-speech segmentation and filtering. The feature extraction step deals with the transformation of the incoming digital waveform into a vector representation of desirable speech features that emphasize linguistic information. Usually, a speech signal is broken into short (usually around 20-30 ms) segments (frames), overlapped every 10ms, during which the signal is assumed to be stationary [13]. During the last few decades, several techniques have been developed for feature extraction from speech signals. These approaches include Mel-Frequency Cepstral Coefficients (MFCC), Perceptual Linear prediction (PLP), Relative Spectral (RASTA), Linear Predictive Coding (LPC) [12]. However, probably, the mostly used is MFCC [13].

MFCC extract spectral features by defining an analysis window (around 25ms) and divide the speech signal into different time frames by shifting the window using a shifting stride of 10ms. Then Fast Fourier Transformation (FFT) is calculated for each frame to obtain the frequency features, and the logarithmic Mel-Scaled filter bank is applied to its power spectrum estimate. MFCCs calculate the Discrete Cosine Transformation (DCT) of the log energies in the corresponding frequency bands to obtain a m-dimensional coefficients vector. The measured power spectrum envelope in each frame, correlates to the shape of the vocal tract, providing an appropriate representation of the sound or phone being produced. This procedure results are a feature vectors that can be arranged in a [n×m] matrix where m is the number of coefficients and n is the number of frames.

At the heart of an ASR system is the decoder. During this phase, feature vectors are decoded into linguistic units that make up speech. The decoding relies on acoustic and language models [14] to recover the most probable utterance by modeling the conditional probability of the nth word, using the (n-1) earlier words. Linguistic and pronunciation dictionary are often used to improve the decoding performance. An acoustic model [14] is a fundamental part of ASR system, where the connection between the acoustic information (e.g. feature vectors) and phonetics is established. The acoustic models are frequently implemented using various methods that includes Hidden Markov Model (HMM) [15], support vector machines (SVM) [16]. HMM is the most commonly used acoustic model for speech recognition in many practical applications.



**Fig. 1.** Automatic speech recognition system diagram.

ASR is a wide topic that includes sub-topics related to interjection recognition. One of those sub-topics is Keyword Spotting (KWS) [17]. The need for enabling users to have a fully hands-free experience that can be very useful in situations like driving, resulted in the development of system that listens continuously for specific keywords to initiate voice input. Keyword Spotting aims at detecting predefined keywords in an audio stream. A commonly used technique for keyword spotting is the Keyword/Filler Hidden Markov Model (HMM). One disadvantage with this technique is that it can be computationally expensive, and the model is trained separately for each keyword [18].

Keyword spotting is sometimes performed using a pattern matching approach where the input is compared with a few pre-recorded commands. Since, the same word might be articulated with a different speed, the Dynamic Time Warping (DTW) algorithm is used for aligning two sequences in an optimal way [19]. Recent neural network models show a significant improvement over the HMM approach. The recurrent neural networks (RNN) is used at Deep KWS model [18] shows good performance while keeping reduced runtime computation, and smaller memory footprint. A Convolutional Neural Network (CNN) architecture that is described in [20] shows improvements over FNN in a variety of small and large vocabulary tasks. The described CNN architecture generalizes more easily to different speaking styles compared to a fully connected FNN architecture.

Another sub-topic related to interjection recognition is Audio Event Detection (AED) [7] which considered as a common task of processing speech and voice signals. An audio event is a specific type of sound, such as footsteps, running water, exhaust fan noise, scream, ocean wave breaking or music. Many sound clips contain multiple acoustic events that overlap on the time axis. AED is a task in which a relatively long (several seconds to tens of seconds) sound clip including multiple acoustic events is input and acoustic event labels and their time stamps (start and end times) are output. The process of AED is combined from extracting acoustic features using MFCC in general, and then constructing classiﬁcation models using Gaussian mixture models (GMM) [21], HMM [22], SVM [23] or more recently models such as CNN [24] or RNN [25]. Because acoustic events can often have a temporal overlap, the most diﬃcult problem in AED is how to detect active durations of acoustic events. For instance, in [26] Lee et al. proposed an acoustic event detection based on a convolutional neural network, which calculates a posterior for the existence of acoustic events time frame by time frame.

Generally, deep learning requires a large amount of labeled training data to enable accurate speech recognition. For interjection recognition such large dataset is not existing to our best knowledge. Therefore, data augmentation is proposed where the speech data is artificially augmented by applying different types of distortions in a way that does not change the label. In [27] data augmentation is proposed for low resource speech recognition tasks. The performance of a FNN speech recognition model depends on how well the training data matches the testing data. This can be done by increasing the generalization of a model further than the data provided to it, and by attempting to create in the training data a similarity between the representative characteristics that are seen in real data, such as voice variability of different speakers, or different background noises.

There are many options how data can be augmented. Some of them are applied in the feature level of a neural network, and some are applied directly in the raw audio level. Vocal tract length perturbation (VTLP) [28] is a popular method for doing feature level data augmentation in speech, has shown gains on the TIMIT (TIMIT is a corpus of phonemically transcribed speech of [American English](https://en.wikipedia.org/wiki/American_English) speakers) phoneme recognition task. SpecAugment [29] is another feature level data augmentation method for speech recognition that operates on the log Mel spectrogram of the input audio. In the raw audio level, an intuitive and practical transformations such as Dynamic Range Compression (DRC), pitch-shifting, time stretching, and background noise combination, is applying audio effects to the original training audio files [30].

Despite the prevalence of interjections in human speech patterns, interjections in literature mentioned mainly in the context of emotion researchers and understanding the nature of interjections with approaches to studying interjectional meaning [32]. There is a difference between a formal speech and conversation speech. Conversation speech is more spontaneous and efficient and not requires special training. Elizabeth Shriberg describes in [33] four fundamental properties of spontaneous speech that present problems for spoken language applications such as lack of punctuation or the inability to “hear” a speaker’s emotion or state of being through speech. As stated before, SER is still an ongoing subject of research and main traditional techniques for SER are based on feature extraction and selection relevant features to identify various types of emotions [40].

No prior work, to our knowledge, has explored inserting interjections in speech recognition systems. In [34] the authors describe an experiment that systematically manipulated the Amazon Alexa TTS by adding some emotional-cognitive expressions like interjections. This experiment also examines the influence of inserting interjections with different duration and pitch level. The conclusion was that those manipulations improved user’s ratings of their conversation across thousands of real user interactions. An emotional speech recognition system that classified “fear emotion” for systems like emergency call centers is described in [35]. This system used Support Vector Machines (SVM), with an interjections feature, which can be classified as a type of incredulity observed in spontaneous speech when a speaker gets hyperemotional. It is important to clarify that this system can’t recognized different interjections but rather knows how to classify calls into two classes: calls that includes fear and calls that not includes fear.

1. Our Contribution

We represent a new problem of interjection classification task. We implement the classification by a fully connected feedforward neural network (FNN) and Long Short-Term Memory Long (LSTM) . It can be challenging to identify a sound that is a short fragment, has no context, and same interjection could be pronounced differently by the same speaker, and more likely by various speakers.

To our best knowledge, there is no dataset of interjections that can be used for the research. In this work we create a new dataset of some interjection phrases, for some different speakers. In addition, we propose the use of audio data augmentation for overcoming the disadvantage of data scarcity and explore the influence of different augmentations sets on the performance of the proposed architecture. By training the network on the additional augmentation data, we make the network becomes invariant to these deformations and generalizes better to unseen data.

1. Method
   1. Dataset

The lack of dataset of interjection required to create one. The dataset includes a “clean” unsynthesized audio samples of the 5 labels by 5 different speakers (2 females and 3 males) without background noises. Followed by expanding the dataset with augmentation process that artificially modified pitch, tempo and background noise addition to existing recordings.

### New Dataset Creation. One of the prerequisites of deep learning is a high-quality collection of data that can be used for training. For this project, annotated audio datasets are needed, consisting of short sound clips, and accompanying labels that tell us the subject of the recording. This would identify each sound clip as belonging to one of a finite set of categories and enable the problem to be tackled as a supervised learning task. We choose 4 examples of interjections (Table 1) for this study. They were chosen since they are independent of language and have a semantic meaning by themselves and not requires the context of a conversation. The 5th class of non-interjections words consists of different words that were selected by reading them from several books in completely random order. In addition, the words in this class are different from speaker to speaker.

**Table 1.** Four selected interjections that recorded into separate sound clips.

|  |  |  |
| --- | --- | --- |
| interjection | alternate/similar | translation/meaning |
| nah |  | **"No"** - Informal no |
| mmm | mhm, uh-hu | **"Yes"** - Agreement, acknowledgement |
| ahah | Aha, ahh | **"I understand"** - Understanding,  Confirmation |
| oy | oy vay | **"Oh no..."** - Mainly Jewish: Used to express self-pity, or expression of  unexpected situation |

The sound clips recorded using Auditok [36], which is an VAD (Voice Activity Detection) [37] tool that enable recording and saving each sound clip as a separate wav file with 16000 Hz sampled rate. At the end of that process, 5 folders (4 interjections and one negative example) were created with 5 speakers each. Table 2 lists the profile and the number of audio samples recorded per speaker. The column “number of audio samples” refers to a specific word, i.e. speaker A has 850 audio samples per word and a total number of 4,250 audio samples.

**Table 2.** Speakers profile and number of audio samples recorded for each speaker and each word.

|  |  |  |
| --- | --- | --- |
| speaker | number of audio samples | profile |
| A | 850~ | Gender: **Male**, Age:**44**, Native language: **Hebrew** |
| B | 550~ | Gender: **Female**, Age:**42**, Native language: **Hebrew** |
| C | 300~ | Gender: **Male**, Age:**16**, Native language: **Hebrew** |
| D | 300~ | Gender: **Female**, Age:**81**, Native language: **Spanish** |
| E | 300~ | Gender: **Male**, Age:**50**, Native language: **Hebrew** |

### Data Preprocessing. To extract the useful features from the audio file, Librosa library [38] has been used. This library is a python package for music and audio analysis that provides several methods to retrieve information from sound clips.

### Feature extraction for FNN model. A minimum and maximum recording length was determined. The minimum length is 0.45 second, so as not to allow short and unrelated recordings, such as background noises which were mistakenly recorded separately, to be part of the dataset. The maximum length is 1.55 seconds, for the case quiet was not detected by Auditok between recordings. Records with length that is not in the range between minimum and maximum length were removed.

The methods from Librosa that been used to extract various features and the number of features extracted for each method are:

* ***MFCC (Librosa.feature.mfcc)*** - 40 features.
* ***Melspectrogram (librosa.feature.melspectrogram)*** - 128 features.
* ***Chorma-stft (librosa.feature.chroma\_stft)*** - 12 features.
* ***Spectral contrast (librosa.feature.spectral\_contrast)*** - 7 features.
* ***Tonnetz (librosa.feature.tonnetz)*** - 6 features.

The result of that process is a matrix with a row for each sample audio file, and a column for each mean feature value (193 column).

#### Feature extraction for LSTM. All recordings remained at their original length filtered like recording in FNN by a minimum length of 0.45 second and maximum recording length of 1.55 seconds. MFCC from Librosa has been used with default sample rate of 22 KHz and analysis window of 10 KHz. Each window divided into 21 frames and 40 coefficients were extracted from each frame. The result of that process is a 3-dimension matrix [w, 21 (number of frames), 40 (number of coefficients)] where w is the number of windows.S

Data Augmentation System (DAS). To extend the collected dataset a data augmentation system (DAS) that generates a synthetically samples was developed. We used Pysox [39], a Python library that provides a simple interface between Python and Sox, which is a popular command line tool for sound processing that can apply various effects to audio files. We tried 4 different sound effects on our original unsynthesized (“clean data”) audio samples. Each effect is applied directly to the sound file prior to converting it into the input vector representation used to feed into the neural network. The effects for create data augmentation are described below:

* ***Tempo***: Change the audio playback speed but not its pitch. This function gets the parameter ‘factor’ (factor > 1 speed up the audio signal, factor < 1 slow down the audio signal). Duration of the sound file is changing.
* ***Pitch***: Change the audio pitch (but not tempo). The sensation of a frequency is commonly referred to as the pitch of a sound. A high pitch sound corresponds to a high frequency sound wave and a low pitch sound corresponds to a low frequency sound wave. One octave (the [interval](https://en.wikipedia.org/wiki/Interval_(music)) between one musical [pitch](https://en.wikipedia.org/wiki/Pitch_(music)) and another) is divided into 12 semitones (tones) of 100 cents each. Typically, cents are used to express small intervals. 1200 cents equal to 1 octave. This function gets the parameter ‘n-semitones’ (The positive or negative number of semitones to shift).
* ***Background noise***: Mix the original “clean” audio with another recording containing background sounds from several acoustic scenes. Each sample is mixed with 9 acoustic scenes: baby gibberish, ambulance, crowd laughing, football crowd, mall, passing bus, rain, street traffic and tv. Each mix Smix was generated using equation 2

Smix = (Worig × Sorig) + (Wbackground × Sbackground) (1)

where Sorig is the original audio sample, Sbackground is the signal of the background

scene. Worig and Wbackground are weighting volume parameters determined under the

care that Worig + Wbackground = 1. For each Sorig audio, 2 or 3 Smix audio files with

different weights were generated, depends on the dataset.

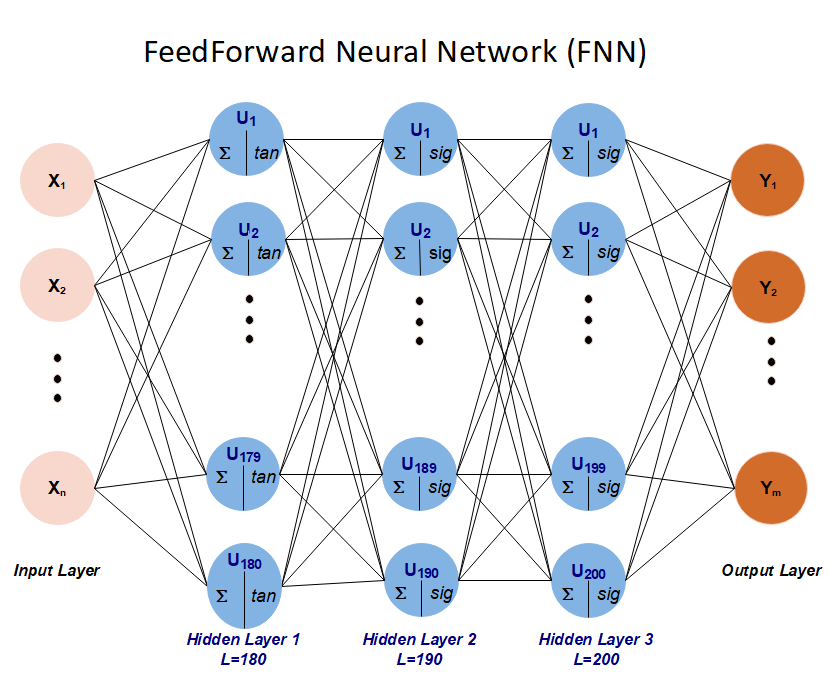
* ***Norm***: Normalize an audio file to a dB level. This function gets as a parameter the requested ‘db\_level’.

DAS can create a large amount of synthetically samples from original “clean” samples. The number of synthetically samples determined by the configuration of the methods to use (tempo, pitch and background noise are the 4-supported methods now, but any other method can be easily added), and by configuration of each method values. For example, if the configuration for the pitch method contains 4 different values for the semitone parameter, then the system will be generated 4 different augmented audio samples with the desire pitch level. A significant advantage in DAS is the fact that different effects can be combined for each recording. This allows us to control the size and the diversity of the dataset we want to create i.e. if the desired methods are tempo (with desired factor 0.9 and 1.1) and pitch (with desired semitone 2 and -2), additional 8 audio samples will be created from 1 original audio (2 audios with only tempo effect, 2 with only pitch and 4 that are result of combining parameters of tempo and pitch together).

In addition, DAS can generate white noise [36] in the background of each original audio files. This functionality was added because white noise is a type of noise that is produced by combining sounds of all different frequencies together and can be used to mask other sounds like different background noises.

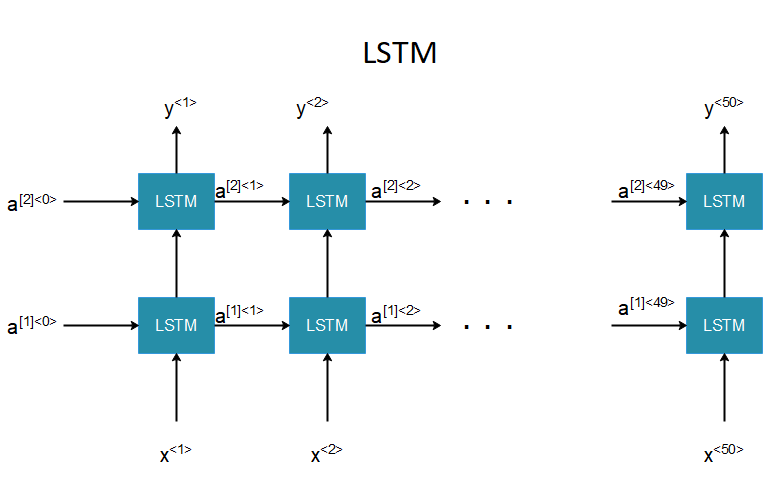
* 1. Network architecture

FNN. The implementation made in TensorFlow. A set of three layers was trained with 280 units and tangent activation function, 290 and 300 units for the second and third layers with sigmoid function. Since we are dealing with multiclass classification problem, the output layer use SoftMax as its activation function, which outputs a vector that represents the probability distributions of a list of potential outcomes. The loss function used was Multi-class cross-entropy. Training was done using the Adam optimizer with initial learning rate of 0.009 and ~300 epochs. Every 5 epochs F-score performance on the validation set was checked and the model with the best validation F-score was saved.



**Fig. 2.** Feedforward neural network architecture.

LSTM. The implementation made in TensorFlow. A set of two LSTM layers with 50 units each was used. The output use SoftMax as its activation function. The loss function used was Multi-class cross-entropy. Training was done using the Adam optimizer with initial learning rate of 0.008 and ~300 epochs, while in each epoch a mini-batch of approximately size of 10% from the training length is trained. Every 5 epochs F-score performance on the validation set was checked and the model with the best validation F-score was saved.



**Fig. 3.** LSTM architecture. x<t> is the input at time t. y<t> is the input at time t. a[l]<t> is the input of cell at time t+1 from layer l.

1. Experimental Setup
   1. Creating Different Datasets By DAS

It’s important to choose the augmentation parameters such that the semantic validity of the label is maintained. The parameters we choose to create our augmentation sets are: factor in range of 0.86-1.14 for tempo, semitone in range of (-2.4)-2.4 for pitch, Worig in range of 0.83-0.93 and Wbackground in range of 0.07-0.17 for background noises. We claim that each of the augmentation is helpful, but their combination gives better results. The resulting augmentation sets are described in the next table:

**Table 3.** Description of the different augmentation sets created.

|  |  |  |  |
| --- | --- | --- | --- |
|  | original samples per class | used effects | generated samples per class and per speaker |
| 1 | 120 | Pitch | 2520 |
| 2 | 120 | Tempo | 2520 |
| 3 | 100 | Background | 2900 |
| 4 | 60 | Tempo + Pitch | 4860 |
| 5 | 40 | Pitch + Background | 6880 |
| 6 | 40 | Tempo + Background | 6880 |
| 7 | 10 | Tempo + Pitch + Background | 9320 |
| 8 | 50 | Pitch + Norm | 1500 |
| 9 | 1 | Tempo + Pitch + Norm + Background | 932 |

For each speaker 120 different audio samples were used from our original dataset (600 files altogether from 4 interjections folders and 1 folder of negative examples), whose overall length is about 8 minutes, to generate 7 augmentation sets with total amount of 180,000 audio samples, whose overall length is about **78 hours**. To all speakers, the system generates **more than 300 hours** of augmentation data from 32 minutes of original recordings. Depending on the results obtained in the experiments, at a more advanced stage of those experiments an additional function was needed. The ‘Norm’ method added to DAS and 2 additional data sets (datasets 8 and 9) have been created which are described later in this paper.

* 1. Scenario

In this section, we describe scenario using different configurations, while training and test sets have been kept separate. F-score, which is a commonly used measure of classification accuracy that gives equal weight to precision (how many instances were correctly predicted, given all the predicted labels for a given class X) and recall (how many instances were correctly captured, from all instances that should have a label X), used for computing the score of the recognition process. The F-Score equation is shown below:

(2)

Where TP (True positive) is the number of correct classifications by the classifier, FN (False negative) is the number of misclassified predictions where the model predicts it’s not from label X, but it is. And FP (False positive) is the number of misclassified predictions where the model predicts it’s from label X, but it’s not. The choice was based on the intuition that a good classifier should maximize both precision and recall simultaneously. So, a model with good precision and recall will score better than a model that has extremely good performance on just one of them.

We evaluate the performance of our models on scenarios that testing previously unseen speakers. In each scenario the model trained with each of the data augmentation sets described in table 3, while F-score for each data set is computed by the mean F-score of run it 7 times.

### Scenario description. The goal is to check how augmentation improves the results for previously unseen speaker. The first step of this scenario was to train the model with clean unsynthesized data of two speakers (1 female and 1 male) and then validate and test it separately on clean unsynthesized data of 2 unseen speakers (1 female and 1 male). Next, we trained it separately on each of our 7-data augmentation sets and then validate and test it the same way as in the first step i.e. same unsynthesized data of each unseen speakers.

We extended this scenario for the LSTM model and trained the model not only with 2 speakers, but also with 1 and 3 different speakers. The testing done separately on unsynthesized data of the same 2 unseen speakers. Depending on the results obtained for the experiments described above, another extension and important test done with only 1 recording of 10 speakers. 10 different speakers of both genders (age 11 to 75) recorded only once for each word. One new dataset augmented by tempo, pitch and background methods was created from those 10-speaker’s audio recording to train the model, and again as in the previous tests, the testing done separately on unsynthesized data of the same 2 unseen speakers.

1. Results and Discussion

The experiment was conducted in our study for each model (FNN and LSTM). The accuracy assessment was measured by compared the baseline (unsynthesized clean dataset) F-score with the F-score of each of the proposed augmentation datasets. First, the model was trained by two speakers and better results were obtained by the LSTM model. Because of this reason, it was decided to extend the experiment in this model to train it with one and three different speakers as well. By doing so, we want to examine the impact of the number of trained speakers on F-score.

* 1. FNN Results

Results of the FNN model are presented in the tables below. The model trained twice with samples of 2 speakers (speaker A and speaker B). Then, first validate on unseen speaker C and tested on speakers D and E separately (table 4) and in addition validate on unseen speaker D and tested on speakers C and E separately (table 5). We can realize that almost each of the augmented method significantly improves the F-score relatively to the baseline F-score that trained with original unprocessed samples. Except for a few cases, each of the augmentation sets were helpful in this scenario, and in 4 of 6 columns, the highest classification F-score improvement for each unseen speaker was achieved by the dataset combined with at least 2 augmentation methods.

**Table 4.** Results of the first scenario for FNN model where validation made on unseen speaker C and test made on speaker D and E. The percentages within the parentheses present the F-score improvement/decrease compared to the baseline original unsynthesized dataset F-score.

|  |  |  |  |
| --- | --- | --- | --- |
| **FNN:** train with 2 speakers and validate with unseen speaker c | | | |
| training dataset  (speakers a+b) | validate f-score (**speaker c**) | test f-score (**speaker d**) | test f-score (**speaker e**) |
| Unsynthesized Data | 0.42 | 0.443 | 0.208 |
| Pitch | 0.516 (↑ 22.9%) | 0.425 (↓ 4.2%) | 0. 24 (↑ 15.4%) |
| Tempo | 0.485 (↑ 15.5%) | 0.45 (↑ 1.6%) | 0. 252 (↑ 21.2%) |
| Background | 0.504 (↑ 20%) | 0.401 (↓ 10.5%) | 0.3 (↑ 44.2%) |
| Tempo + Pitch | 0.538 (↑ 28.1%) | 0.425 (↓ 4.2%) | 0.25 (↑ 20.2%) |
| Pitch + Background | 0.544 (↑ 29.5%) | 0.445 (↑ 0.4%) | 0.247 (↑ 18.8%) |
| Tempo + Background | 0.521 (↑ 24%) | 0.447 (↑ 0.9%) | **0.304** (↑ 46.2%) |
| Tempo + Pitch + Background | **0.631** (↑ 50.2%) | **0.503** (↑ 13.5%) | 0.23 (↑ 10.6%) |

In table 5, best F-score for speakers C and E accepted by 1 method, but not far from the best F-score, the second highest F-score was accepted by a combination of at least 2 methods. F-score of unseen speaker E was very low related to accepted F-score of tested speakers D and C. Still, all augmented datasets significantly improved the F-score.

**Table 5.** Results of the first scenario for FNN model where validation made on unseen speaker D and test made on speaker C and E. The percentages within the parentheses present the F-score improvement/decrease compared to the baseline original unsynthesized dataset F-score.

|  |  |  |  |
| --- | --- | --- | --- |
| **FNN:** train with 2 speakers and validate with unseen speaker d | | | |
| training dataset  (speakers a+b) | validate f-score (**speaker d**) | test f-score (**speaker c**) | test f-score (**speaker e**) |
| Unsynthesized Data | 0.521 | 0.333 | 0.213 |
| Pitch | 0.525 (↑ 0.8%) | 0.455 (↑ 36.6%) | 0. 266 (↑ 24.9%) |
| Tempo | 0.521 (0%) | **0.528** (↑ 58.6%) | 0. 247 (↑ 16%) |
| Background | 0.506 (↓ 3%) | 0.466 (↑ 40%) | **0.298** (↑ 40%) |
| Tempo + Pitch | 0.534 (↑ 2.5%) | 0.505 (↑ 51.7%) | 0.282 (↑ 32.4%) |
| Pitch + Background | 0.497 (↓ 4.8%) | 0.404 (↑ 21.3%) | 0.254 (↑ 19.2%) |
| Tempo + Background | 0.539 (↑ 3.5%) | 0.38 (↑ 14.1%) | 0.282 (↑ 32.4%) |
| Tempo + Pitch + Background | **0.591** (↑ 13.4%) | 0.525(↑ 57.7%) | 0.274 (↑ 28.6%) |

* 1. LSTM Results

Results of the LSTM model are presented in the next tables. Tables 6 and 7 presents the results where model trained with audio samples of 1 speaker (speaker A). Tables 7 and 8 presents the results where model trained with audio samples of 2 speakers (speaker A and B) and tables 9 and 10 present the results where model trained with audio samples of 3 speakers. After training, models first validated on previously unseen speaker C and tested on speakers D and E, and in addition, validated on previously unseen speaker D and tested on speakers C and E (Except for the case where model trained on 3 speakers and tested only on speaker E).

Training By 1 Speakers. In table 6, each of the augmented method significantly improves the validation F-score relatively to the baseline F-score that trained with original unprocessed samples. Test F-score of speaker D also achieved significantly improvement except for the pitch method, and test F-score of speaker E achieved lowers results than speaker D and improvement only in some cases, but for all speakers the best result accepted by a combination of methods (22.3% and 23.2% with Tempo+Pitch for speakers D and E and 21% improvement with 3 methods for speaker C).

**Table 6.** Results of the first scenario for LSTM model where validation made on unseen speaker C and test made on speaker D and E. The percentages within the parentheses present the F-score improvement/decrease compared to the baseline original unsynthesized dataset F-score.

|  |  |  |  |
| --- | --- | --- | --- |
| **lstm:** train with 1 speaker and validate with unseen speaker c | | | |
| training dataset  (speakers a) | validate  f-score (**speaker c**) | test  f-score (**speaker d**) | test  f-score  (**speaker e**) |
| Unsynthesized Data | 0.501 | 0.403 | 0.297 |
| Pitch | 0.603 (↑ 20.4%) | 0.454 (↑ 12.7%) | 0. 365 (↑ 22.9%) |
| Tempo | 0.503 (↑ 0.4%) | 0.392 (↓ 2.8%) | 0. 302 (↑ 1.7%) |
| Background | 0.54 (↑ 7.8%) | 0.431 (↑ 6.9%) | 0.249 (↓ 19.3%) |
| Tempo + Pitch | 0.595 (↑ 18.8%) | 0.424 (↑ 5.2%) | 0.265 (↓ 12.1%) |
| Pitch + Background | 0.58 (↑15.8%) | **0.493** (↑ 22.3%) | **0.366** (↑ 23.2%) |
| Tempo + Background | 0.562 (↑ 12.2%) | 0.419 (↑ 4%) | 0.275 (↓ 8%) |
| Tempo + Pitch + Background | **0.606** (↑ 21%) | 0.478(↑ 18.6%) | 0.296 (↓ 0.3%) |

In table 7, most of the methods improves the F-score for speakers C and D (validate) and improvements for speaker E only in 3 augmented datasets. Unlike table 6, best improvement for all speakers accepted by only 1 method (pitch), but also combined datasets for speaker C and D achieved very significant improvement. Possible explanation that pitch is such an important factor in table 7 is that the model trained only with 1 speaker and validate by speaker that is very different in its characteristics (44 years old male and 81 years old female).

**Table 7.** Results of the first scenario for LSTM model where validation made on unseen speaker D and test made on speaker C and E. The percentages within the parentheses present the F-score improvement/decrease compared to the baseline original unsynthesized dataset F-score.

|  |  |  |  |
| --- | --- | --- | --- |
| **lstm:** train with 1 speaker and validate with unseen speaker d | | | |
| training dataset  (speakers a) | validate  f-score (**speaker d**) | test  f-score (**speaker c**) | test  f-score  (**speaker e**) |
| Unsynthesized Data | 0.494 | 0.378 | 0.351 |
| Pitch | **0.576** (↑ 16.6%) | **0.58** (↑ 53.4%) | **0. 443** (↑ 26.2%) |
| Tempo | 0.493 (↓ 0.2%) | 0.411 (↑ 8.7%) | 0. 323 (↓ 8.7%) |
| Background | 0.502 (↑ 1.6%) | 0.448 (↑ 18.5%) | 0.235 (↓ 49.4%) |
| Tempo + Pitch | 0.537 (↑ 8.7%) | 0.552 (↑ 46%) | 0.389 (↑ 10.8%) |
| Pitch + Background | 0.57 (↑ 15.4%) | 0.536 (↑ 41.8%) | 0.341 (↓ 2.9%) |
| Tempo + Background | 0.493 (↓ 0.2%) | 0.498 (↑ 31.7%) | 0.228 (↓ 53.9%) |
| Tempo + Pitch + Background | 0.559 (↑ 13.2%) | 0.533(↑ 41%) | 0.35 (↓ 0.3%) |

#### Training By 2 Speakers. Results in tables 8 and 9 are analogous in terms of trained speakers to the results of the FNN model in tables 4 and 5. The column “FNN” present improvement/decrease compared to the results of the same dataset in tables 4 and 5. As shown in table 8, the highest improvement was achieved significantly by combining all three augmentation methods for all unseen speakers C, D and E. Each of the augmented datasets (except a small decrease for Tempo + BckGrd dataset for tested speaker D) significantly improves F-score relatively to the baseline F-score and also over the results of corresponding dataset that shown in table 4. Best validation F-score achieved in LSTM for speaker C was 0.631, which is equal to the best validation F-score achieved for speaker C in FNN, but F-score of tested speakers D and E was higher compares to the analogous FNN F-score. The efficiency of DAS is noticeable when looking at the results of speaker D. F-score of Speaker D with unsynthesized dataset in LSTM was lower in 5.2% than corresponding F-score in FNN, but 6 of 7 augmented datasets in LSTM produce a much higher F-score in LSTM corresponding to the analogous F-score in FNN model.

**Table 8.** Results of first scenario for LSTM where validation made on unseen speaker C and test made on speaker D and E. The percentages within the parentheses present improvement/decrease compared to the baseline clean dataset F-score. The columns “**FNN**” present improvement/decrease compared to the results of the same dataset in Table 4. (**BckGrd** = Background)

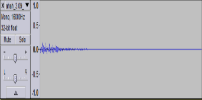
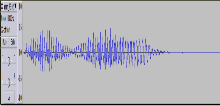
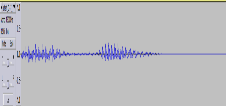
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **lstm:** train with 2 speakers and validate with unseen speaker c | | | | | | |
| training dataset  (speakers a+b) | validate  f-score  (**speaker c**) | fnn | test  f-score  (**speaker d**) | fnn | test  f-score  (**speaker e**) | fnn |
| Unsynthesized Data | 0.516 | ↑22.9% | 0.421 | ↓5.2% | 0.279 | ↑34.1% |
| Pitch | 0.573 (↑11%) | ↑11% | 0.492 (↑9.2%) | ↑15.8 | 0.381 (↑36.6%) | ↑58.8% |
| Tempo | 0.568 (↑10.1%) | ↑17.1% | 0.481 (↑4.9%) | ↑6.9% | 0.382 (↑36.9%) | ↑51.6% |
| BckGrd | 0.593 (↑14.9%) | ↑17.7% | 0.452 (↑4.7%) | ↑12.7% | 0.323 (↑15.8%) | ↑7.7% |
| Tempo + Pitch | 0.575 (↑11.4%) | ↑6.9% | 0.521 (↑6.5%) | ↑22.6% | 0.398 (↑42.7%) | ↑59.2% |
| Pitch + BckGrd | 0.588 (↑14%) | ↑8.1% | 0.528 (↑10.3%) | ↑18.7% | 0.372 (↑33.3%) | ↑50.6% |
| Tempo + BckGrd | 0.526 (↑1.9%) | ↑1% | 0.432 (↓0.2%) | ↓3.5% | 0.326 (↑16.8%) | ↑7.2% |
| Tempo + Pitch + BckGrd | **0.631** (↑22.3%) | 0% | **0.566** (↑22.6%) | ↑12.5% | **0.401** (↑43.7%) | ↑74.3% |
| Pitch + Norm | 0.59(↑14.3%) |  | 0.557 (↑32.3%) |  | **0.476** (↑70.6%) |  |

In table 9, all augmented datasets improve the F-score for speakers C and D (except a small decrease for Tempo + BckGrd dataset for validate speaker D) and improvements for speaker E achieved only in 2 augmented datasets. The highest classification F-score improvement for unseen speakers C and D achieved by the dataset combined with all three augmentation methods (36.3% for speaker C and 22.6% for speaker D), while the best F-score improvement for speaker E achieved by the pitch dataset. F-score in this table shows a consistent growth of LSTM F-score for speakers D and E comparing to F-score of corresponding speakers in FNN. For speaker C the results are slightly different, while 4 of 7 augmented datasets, including the dataset with the best F-score, achieved lower result comparing to the baseline dataset.

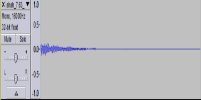
**Table 9.** Results of first scenario for LSTM where validation made on unseen speaker D and test made on speaker C and E. The percentages within the parentheses present improvement/decrease compared to the baseline clean dataset F-score. The columns “**FNN**” present improvement/decrease compared to the results of the same dataset in Table 5. (**BckGrd** = Background)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **lstm:** train with 2 speakers and validate with unseen speaker c | | | | | | | |
| training dataset  (speakers a+b) | validate  f-score  (**speaker d**) | fnn | test  f-score  (**speaker c**) | fnn | test  f-score  (**speaker e**) | fnn |
| Unsynthesized Data | 0.535 | ↑2.7% | 0.372 | ↑1.2 | 0.426 | ↑100% |
| Pitch | 0.584 (↑9.2%) | ↑11.2% | 0.481 (↑29.3%) | ↑5.7 | **0.457** (↑7.3%) | ↑71.8% |
| Tempo | 0.561 (↑4.9%) | ↑7.7% | 0.382 (↑2.7%) | ↓38.2% | 0.37 (↓15.1%) | ↑49.8% |
| BckGrd | 0.56 (↑4.7%) | ↑10.7% | 0.429 (↑15.3%) | ↓8.6% | 0.343 (↓24.2%) | ↑15.1% |
| Tempo + Pitch | 0.57 (↑6.5%) | ↑6.7% | 0.449 (↑20.7%) | ↓12.5% | 0.452 (↑6.1%) | ↑60.3% |
| Pitch + BckGrd | 0.59 (↑10.3%) | ↑18.7% | 0.488 (↑31.2%) | ↑20.8% | 0.397 (↓7.3%) | ↑56.3% |
| Tempo + BckGrd | 0.534 (↓0.2%) | ↓0.9% | 0.43 (↑15.6%) | ↑13.2% | 0.385 (↓10.6%) | ↑36.5% |
| Tempo + Pitch + BckGrd | **0.656** (↑22.6%) | ↑11% | **0.507** (↑36.3%) | ↓3.6% | 0.396 (↓7.6%) | ↑44.5% |
| Pitch + Norm | 0.589(↑10.1%) |  | **0.533** (↑43.3%) |  | **0.506** (↑18.8%) |  |

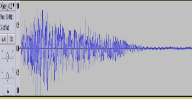
Following the much lower results of speaker E relative to the results of baseline datasets in tables 6,7 and 9 and relative to the results of tested speakers C and D, an examination was performed to understand the cause for it. From examination of several audio samples waveforms of speakers A to E it was discerned that the amplitude level varies greatly between speakers like as seen in figure 5, even though all speakers recorded in same conditions and environment. All the models were trained on speakers A-D, validated on speakers C or D, and tested on speakers C, D and E. Speaker E was the only



**Speaker C**



**Speaker D**



**Speaker E**

**Speaker A**

**Speaker B**

**Fig. 4.** Unsynthesized waveforms of speakers A-E saying “ahah”.

speaker that was functioned only as a tested speaker, and he was tested on models that trained and validated on speakers with much lower amplitude level. In order to improve the results for speaker E, the goal was to find a way to adjust the training dataset to better classification of speakers with different amplitude level. The “norm” method that added to DAS was intended to make this adjustment by normalize an audio sample to a particular dB (decibel) level.

Dataset number 8 from table 3 was created to examine the effect of the ‘norm” method on the results described in tables 8 and 9. 5 dB level values (-1,0,1,2,3) were tried in this dataset. As can be seen in tables 8 and 9 the F-score of speaker E with the new dataset including the “norm” method outperform the best F-score achieved without the norm method (0.476 instead of 0.401 in table 8 and 0.506 instead of 0.457 in table 9). For speakers C and D in both tables, the “norm” dataset achieved F-score that mostly was close or even better than the best F-score without “norm”. Adding this new method illustrates the effectiveness of DAS that can be adjust and become more efficient very easily by adding new methods and apply different characteristics to the dataset.

Training By 3 Speakers. The third phase of this scenario, where the model trained with 3 speakers and tested with unseen speaker support the claim that augmentation is helpful for improving the model F-score. For validation sets the combination of the three methods achieved the highest F-score. Speaker C achieved 23.1% improvement in F-score (0.751) over the F-score of the basic unsynthesized dataset, and speaker D achieved improvement of 13%. Testing the models was done only on speaker E, and although the F-score was much lower than validation F-score, still, DAS managed to greatly improve F-score relative to baseline dataset F-score. As in the case of training the model with 2 speakers, the assumption is that F-score of speaker E can be improved by using a different configuration that include the “norm” method. Another thing that can be noticed is the improvement of test F-score comparing to corresponding F-score achieved in tables 8 and 9 where the model trained with 2 speakers.

**Table 10.** Results of the first scenario for LSTM model. The percentages within the parentheses present improvement/decrease compared to the baseline clean dataset F-score.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **lstm:** training with 3 speakers and validating with 2 unseen speakers | | | | |
|  | training (speakers **a + b + d**) | | training (speakers **a + b + c**) | |
| dataset | validate  f-score  (**speaker c**) | test f-score  (**speaker e**) | validate  f-score  (**speaker d**) | test f-score  (**speaker e**) |
| Unsynthesized Data | 0.61 | 0.425 | 0.592 | 0.446 |
| Pitch | 0.712 (↑16.7%) | 0.506 (↑19.1%) | 0.636 (↑7.4%) | 0.481 (↑7.8%) |
| Tempo | 0.606 (↓0.6%) | 0.48 (↑12.9%) | 0.625 (↑5.6%) | **0.549** (↑23.1%) |
| Background | 0.609 (↓0.2%) | 0.448 (↑5.4%) | 0.568 (↓4.2%) | 0.408 (↓9.3%) |
| Tempo+Pitch | 0.666 (↑9.2%) | **0.543** (↑27.8%) | 0.619 (↑4.6%) | 0.515 (↑15.5%) |
| Pitch+Background | 0.67 (↑9.8%) | 0.458 (↑7.8%) | 0.636 (↑7.4%) | 0.422 (↓5.7%) |
| Tempo+Background | 0.6 (↓1.7%) | 0.406 (↓4.7%) | 0.548 (↓8%) | 0.434 (↓2.8%) |
| Tempo+Pitch+Background | **0.751** (↑23.1%) | 0.377 (↓12.7%) | **0.669** (↑13%) | 0.418 (↓6.7%) |

F-Score gives some perspective to look at the quality of the model, but its main problem is that it hides the detail we need to better understand the performance of our classification model. For example, we may get a classification F-score of 85%, but we don’t know if that is because all classes are being predicted equally well or whether one or two classes are being neglected by the model. By knowing the true labels, we can use another important metric – a confusion matrix, where each column represents the number of instances in a predicted class and each row represents the instances in an actual class. This metric help to determine where the system is confusing between two classes by comparing the predicted classes with the actual classes.

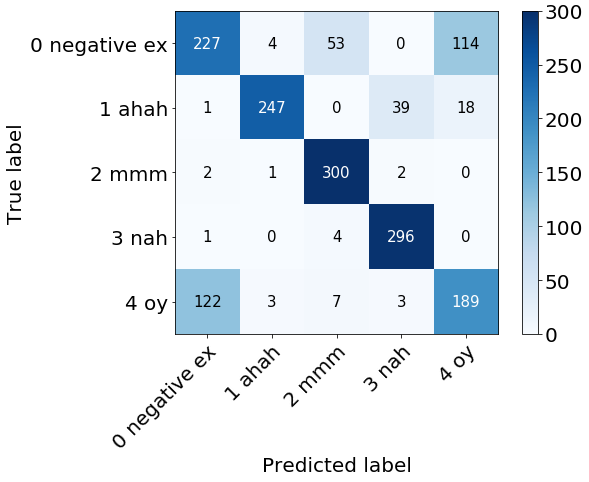
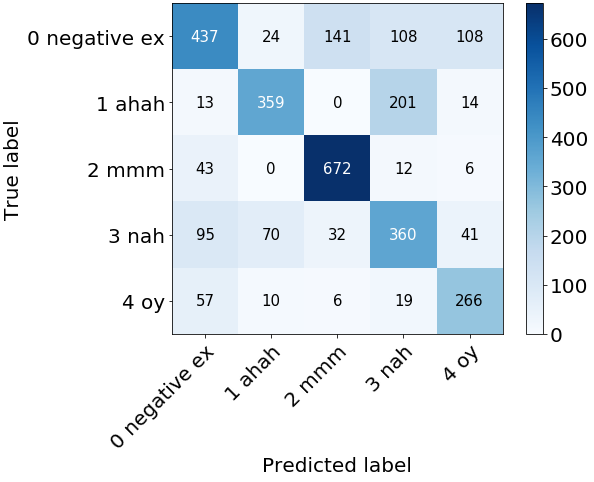
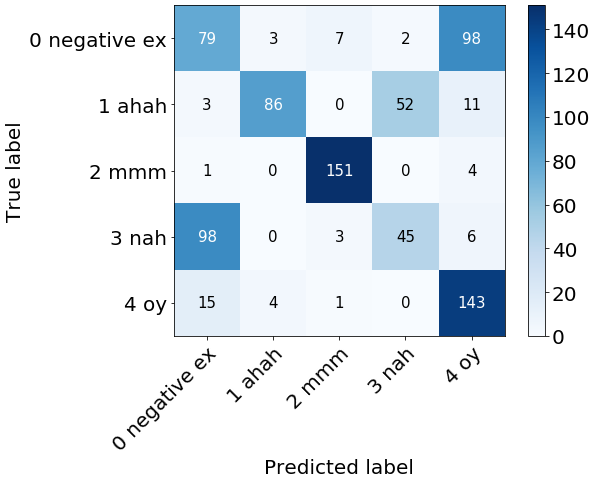
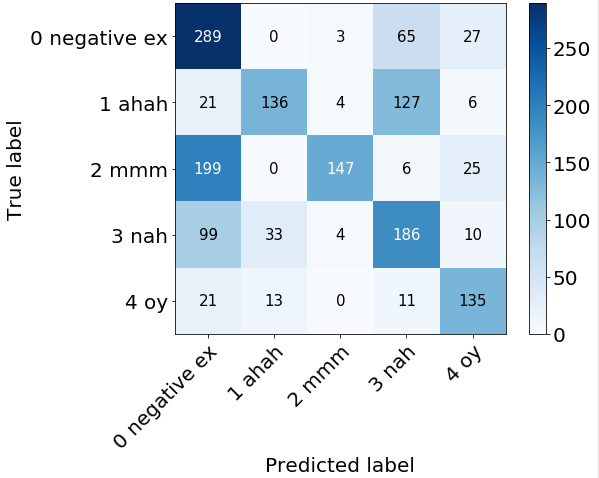
Figure 6 shows two pairs, one pair for each speaker C and D, of confusion matrices taken from training with 3 speakers in the first experiment. Each pair includes one matrix trained by original unsynthesized data, and another matrix trained by all three methods. Table 11 present the recall and precision values for each matrix.

For speaker C labels 0 (“negative example”) and 3 (“nah”) were poor predicted with unsynthesized data. Recall of label 3 is 0.296 i.e. from all labels of class 3, only 45 instances (29.6%) correctly captured (64.5% which are 98 instances predicted as negative examples, 6 predicted as “oy” and 3 predicted as “mmm”). Precision of label 3 is 0.455 i.e. from 99 instances that predicted as “nah”, only 45 instances are really from that class (52 belongs to “ahah” class and 2 are negative examples). With synthesized data recall and precision significantly improved to 0.983 and 0.871 respectively. Recall of label 0 is 0.418 i.e. from all labels of class 0, only 79 instances (41.8%) correctly captured (51.9% which are 98 predicted as “oy”). Precision of label 0 is 0.403 i.e. from 196 instances that predicted as negative example, only 79 instances are really from that class (98 belongs to “nah” class and 15 to “oy”). With synthesized data recall and precision improved to 0.57 and 0.643 respectively. In the same way the recall of label 1 (“ahah”) was improved from 0.566 to 0.81. Except of recall of label 4 (“oy”) that significantly decreased from 0.877 to 0.583, all other values have not changed significantly.

For speaker D a significantly improves between the datasets were detected in precision of label 0 (negative examples) while 45.9% with unsynthesized data increases to 0.678 in synthesized data and recall of 0.463 for class 1 (“ahah”), and 0.39 for class 2 (“mmm”) increases to 0.612 and 0.917 respectively. However, decrease was detected in recall of negative examples from 0.753 to 0.534, and in precision of class 2 (“mmm”) from 0.93 to 0.79.

**Table 11.** Recall and precision values for matrices shown in figure 2.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | speaker c | | | | speaker d | | | |
|  | unsynthesized | | synthesized | | unsynthesized | | synthesized | |
| class | recall | precision | recall | precision | recall | precision | recall | precision |
| negative | 0.418 | 0.403 | 0.57 | 0.643 | 0.753 | 0.459 | 0.534 | 0.678 |
| ahah | 0.566 | 0.925 | 0.81 | 0.967 | 0.463 | 0.747 | 0.612 | 0.775 |
| mmm | 0.968 | 0.932 | 0.984 | 0.824 | 0.39 | 0.93 | 0.917 | 0.79 |
| nah | 0.296 | 0.455 | 0.983 | 0.871 | 0.56 | 0.471 | 0.602 | 0.514 |
| oy | 0.877 | 0.546 | 0.583 | 0.589 | 0.75 | 0.665 | 0.743 | 0.614 |



**Fig. 5.** Top left matrix: Trained by unsynthesized data - Validation F-score for speaker C - 0.621. Top right matrix: Trained by synthesized data - Validation F-score for speaker C: 0.771. Bottom left matrix: Trained by unsynthesized data - Validation F-score for speaker D - 0.57. Bottom right matrix: Trained by synthesized data - Validation F-score for speaker D: 0.677.

DAS greatly improves recall and precision of trained model compared to the trained model with unsynthesized data. Still, most of the higher values outside the diagonal line are belongs to the row or column of negative examples, which means that a lot of samples that belongs to negative class incorrectly captured to this class and a lot of samples that predicted as belongs to negative class actually belongs to other class. To improve that, we suggest increasing the number of samples in the negative class, especially with labels samples of words that is similar to interjections words from other classes.

The second issue examined is the impact of the number of trained speakers on the F-score of previously unseen speaker. In each case of training with 1 speaker, 2 speakers and 3 speakers, the average validation F-score of speakers C and D was calculated among all 8 datasets (1 Original unsynthesized dataset and 7 augmented datasets). Figure 7 shows the average of each datasets compared between the different number of speakers. Improvement can be noticed as the number of trained speakers increases. The F-score of all 8 datasets trained on 3 speakers is larger than the corresponding dataset trained on 2 speakers, and the F-score of 7 from 8 datasets trained on 2 speakers is larger than the corresponding dataset trained on 1 speaker. The claim that the combination of three methods achieved the higher improvement of F-score in relation to use only unsynthesized data is getting support also in this figure.

**Fig. 6.** The average F-score of both speakers C and D according to the number of trained speakers. The dark blue bar indicates the average F-score of dataset trained on 3 speakers. Beneath it, the blue bar indicates the average F-score of dataset trained on 2 speakers, and the orange bar indicates the average F-score of dataset trained on 1 speaker.

Training By 10 Speakers. The strong impact of the number of trained speakers on the F-score of previously unseen speaker convinced us to do one test with a dataset created from only 1 unsynthesized audio sample for each word. The unsynthesized audio file recorded among 10 different speakers as mention in the scenario description. In most cases of validation and test set in the previous experiments with 1,2 and 3 speakers, the best results obtained by combining the three effects (tempo, pitch and background). So first, this test was made with new dataset created from 10 speaker’s audio samples with the same parameters of the corresponding dataset created for 1, 2 and 3 speakers. The first line in tables 12 and 13 shows the results of this experiment. A great improvement can be seen in validation F-score and test F-score for speaker C and D, but on the other side, test F-score on speaker E was much lower from that obtain in table 10 where the model trained with audio sample of 3 speakers.

To improve it a new dataset that includes all 4 effects that DAS can provide, was build (dataset 9 from table 3). Table 12 shows a great improvement both for speaker D and E, and in table 13, F-score of speaker C was decreased a bit to 0.665, but a great improvement was achieved for speaker E.

**Table 12.** Results of first scenario for LSTM trained on 10 speakers where validation made on unseen speaker C and test made on speaker D and E. The percentages within the parentheses present improvement/decrease compared to the test done with 3 speakers.

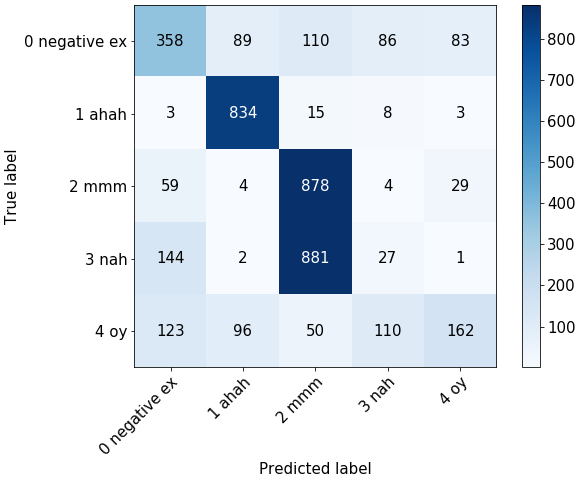
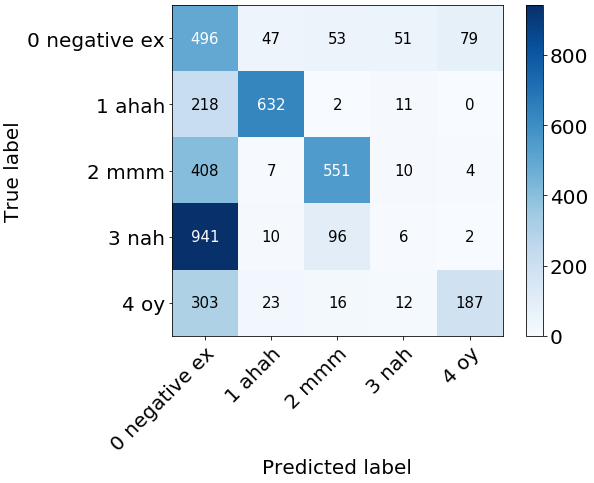
|  |  |  |  |
| --- | --- | --- | --- |
| **lstm:** train with 10 speakers and validate with unseen speaker c | | | |
| training dataset  (10 speakers) | validate f-score (**speaker c**) | test f-score  (**speaker d**) | test f-score  (**speaker e**) |
| Tempo + Pitch + Background | 0.805 (↑ 7.2%) | 0.7 | 0.463 (↓ 17.3%) |
| Tempo + Pitch + Norm +Background | 0.695 | **0.742** | **0.543** |

**Table 13.** Results of first scenario for LSTM trained on 10 speakers where validation made on unseen speaker D and test made on speaker C and E. The percentages within the parentheses present improvement/decrease compared to the test done with 3 speakers.

|  |  |  |  |
| --- | --- | --- | --- |
| **lstm:** train with 10 speakers and validate with unseen speaker d | | | |
| training dataset  (10 speakers) | validate f-score (**speaker d**) | test f-score  (**speaker c**) | test f-score  (**speaker e**) |
| Tempo + Pitch + Background | 0.781 (↑ 16.7%) | 0.681 | 0.464 (↓ 18.3%) |
| Tempo + Pitch + Norm +Background | 0.765 | 0.665 | **0.532** |

The great importance of this test is in two aspects. The first aspect is that the claim that the more the model is trained with more speakers thus his F-score increases, gets approval and support. The second aspect is that our data augmentation system is an essential and very necessary tool because it allows for better results with very small amount of data, and allows to adjust the training data as in the case of speaker E.

Figure 7 shows confusing matrices of 2 tests: One for testing of speaker E on dataset 7 without the “norm” method, and the other for testing of speaker E again but with dataset 9 with the “norm” method. As shown in figure 8 the main problem is in classifying the negative example word. Precision of label 0 without “norm” method is 0.209 i.e. from all instances that predicted as negative, only 21% instances are really from that class. The “norm” method partially fix it while increasing precision to 0.52. As a result of it, classification of classes 1 and 2 were improved, but still, a lot of samples that belongs to class 3 moved to incorrectly predicted as belongs to class 0 without the “norm” effect, to incorrectly predicted as belongs to class 2 with the “norm” effect. We believe that better configuration for DAS can improved it much more.



**Fig. 7.** Left: Trained by synthesized dataset 7 with 10 speakers - Test F-score for speaker E - 0.45. Right: Trained by synthesized dataset 9 with 10 speakers - Test F-score for speaker E: 0.543.

Finally, the models seem to be more robust when using a combination of at least two effects. More precisely, in 7 from 9 cases the combination of three methods achieved the higher improvement of F-score in relation to use only original unsynthesized data. It implies that using some different settings for each effect can improve the results even more than the results we obtained in the scenarios described.

In future experiments, we will add a preprocessing phase for selecting augmentation hyperparameters, which will create effects and combination of effects with several different settings. Each setting will generate a new dataset that will be train on the model. This pre-processing can be helpful in selecting the most effective data augmentation set to the appropriate scenario. Another extension can be to add new augmentation method beyond the 4 methods that we mentioned in our work.

1. Conclusions

In this work we proposed a new problem definition for interjection classification implemented by 2 network architectures: a fully connected feedforward neural network (FNN) and Long Short-Term Memory (LSTM), which in combination with a set of audio data augmentations, produces state-of-the-art results for interjections classification. We showed that the improved performance stems from the combination of a basic classification model and an augmented training set. This combination outperformed the proposed architectures without augmentation at all.

We conducted an experiment to explore the influence of different sets of 4 data augmentation methods on unsynthesized audio samples. We observed that influence of combining between those augmentation methods gives better results than the influence of each of them apart. As shown in table 14, in validation sets, 7 from 8 cases achieved best F-Score by combination of 3 methods and 8 from 14 cases in test sets achieved best F-Score by combination of at least 2 methods. 3 of 4 cases where best F-Score achieved by 1 method were of tested speaker E. In one of this 3 cases the best F-Score achieved by the pitch method (table 9). we saw in that case that combining the norm method with the pitch method improved the best F-Score. We suggest that the performance of the model could be improved further by applying new functionality for better selection of more appropriate data augmentation set for the desired scenario.

**Table 14.** Number of used effects that achieved the higher F-Score.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| validate set | | | | test set | | | |
| 1 effect | 2 effects | 3 effects | 1 effect | | 2 effects | 3 effects |
| 0 of 2 | 0 of 2 | 2 of 2 | 2 of 4 | | 1 of 4 | 1 of 4 |
| 1 of 6 | 0 of 6 | 5 of 6 | 4 of 10 | | 3 of 10 | 3 of 10 |

As part of the experiment, we showed the positive impact of the number of trained speakers on the F-score of previously unseen speaker by training the LSTM model with 1 speaker, 2 speakers and 3 speakers. This positive impact, in addition to the importance of data augmentation have received approval after the second extension of the second scenario in which the LSTM model trained with a new dataset made by 10 different speakers recorded only once for each word.

References

1. Hoy, M. B. (2018). Alexa, Siri, Cortana, and more: an introduction to voice assistants. Medical reference services quarterly, 37(1), 81-88.‏
2. Berdasco, A., López, G., Diaz, I., Quesada, L., & Guerrero, L. A. (2019). User Experience Comparison of Intelligent Personal Assistants: Alexa, Google Assistant, Siri and Cortana. In Multidisciplinary Digital Publishing Institute Proceedings (Vol. 31, No. 1, p. 51).‏
3. Lopatovska, I., Rink, K., Knight, I., Raines, K., Cosenza, K., Williams, H., ... & Martinez, A. (2019). Talk to me: Exploring user interactions with the Amazon Alexa. Journal of Librarianship and Information Science, 51(4), 984-997.‏
4. Clark, L., Doyle, P., Garaialde, D., Gilmartin, E., Schlögl, S., Edlund, J., ... & R Cowan, B. (2019). The State of Speech in HCI: Trends, Themes and Challenges. Interacting with Computers, 31(4), 349-371.‏
5. Warden, P. (2018). Speech commands: A dataset for limited-vocabulary speech recognition. arXiv preprint arXiv:1804.03209.‏
6. Proutskova, P., Rhodes, C., Wiggins, G., & Crawford, T. (2012). Breathy or resonant-A controlled and curated dataset for phonation mode detection in singing.‏
7. Imoto, K. (2018). Introduction to acoustic event and scene analysis. Acoustical Science and Technology, 39(3), 182-188.‏
8. Zhou, Y., Xiong, C., & Socher, R. (2017). Improved regularization techniques for end-to-end speech recognition. arXiv preprint arXiv:1712.07108.‏
9. Richey, C., Barrios, M. A., Armstrong, Z., Bartels, C., Franco, H., Graciarena, M., ... & Gamble, P. (2018). Voices obscured in complex environmental settings (voices) corpus. arXiv preprint arXiv:1804.05053.‏
10. Kulkarni, K. R., & Naik, S. R. R. (2018). A Review of Music Analysis Techniques.‏
11. Zerari, N., Yousfi, B., & Abdelhamid, S. (2016). Automatic Speech Recognition: A Review. Int. Acad. Res. J. Bus. Technol., 2(2), 63-68.‏
12. Gadekar, P., Kaldane, M. H., Pawar, D., Jadhav, O., & Patil, A. (2019). Analysis of speech recognition techniques.‏
13. Majeed, S. A., Husain, H., Samad, S. A., & Idbeaa, T. F. (2015). MEL FREQUENCY CEPSTRAL COEFFICIENTS (MFCC) FEATURE EXTRACTION ENHANCEMENT IN THE APPLICATION OF SPEECH RECOGNITION: A COMPARISON STUDY. Journal of Theoretical & Applied Information Technology, 79(1).‏
14. Ghai, W., & Singh, N. (2012). Literature review on automatic speech recognition. International Journal of Computer Applications, 41(8).‏
15. Gruhn, R. E., Minker, W., & Nakamura, S. (2011). Statistical pronunciation modeling for non-native speech processing - Chapter 2. Springer Science & Business Media.‏
16. Pradhan, A. (2012). Support vector machine-A survey. International Journal of Emerging Technology and Advanced Engineering, 2(8), 82-85.‏
17. Dr. E. Chandra, K.A. Senthildevi. (2015). Keyword Spotting: An Audio Mining Technique in Speech Processing – A Survey, IOSR Journal of VLSI and Signal Processing (IOSR-JVSP).‏
18. Chen, G., Parada, C., & Heigold, G. (2014, May). Small-footprint keyword spotting using deep neural networks. In 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 4087-4091). IEEE.‏
19. Yadav, M., & Alam, A. (2018). Dynamic time warping (DTW) algorithm in speech: a review. Int. J. Res. Electron. Comput. Eng, 6.‏
20. Sainath, T. N., & Parada, C. (2015). Convolutional neural networks for small-footprint keyword spotting. In Sixteenth Annual Conference of the International Speech Communication Association.‏
21. Eronen, A. J., Peltonen, V. T., Tuomi, J. T., Klapuri, A. P., Fagerlund, S., Sorsa, T., ... & Huopaniemi, J. (2005). Audio-based context recognition. IEEE Transactions on Audio, Speech, and Language Processing, 14(1), 321-329.‏
22. Chum, M., Habshush, A., Rahman, A., & Sang, C. (2013). IEEE AASP scene classification challenge using hidden Markov models and frame-based classification. IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events.‏
23. Geiger, J. T., Schuller, B., & Rigoll, G. (2013, October). Large-scale audio feature extraction and SVM for acoustic scene classification. In 2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (pp. 1-4). IEEE.‏
24. Valenti, M., Diment, A., Parascandolo, G., Squartini, S., & Virtanen, T. (2016, September). DCASE 2016 acoustic scene classification using convolutional neural networks. In Proc. Workshop Detection Classif. Acoust. Scenes Events (pp. 95-99).‏
25. Bae, S. H., Choi, I., & Kim, N. S. (2016, September). Acoustic scene classification using parallel combination of LSTM and CNN. In Proceedings of the Detection and Classification of Acoustic Scenes and Events 2016 Workshop (DCASE2016) (pp. 11-15).‏
26. Lee, D., Lee, S., Han, Y., & Lee, K. (2017). Ensemble of convolutional neural networks for weakly-supervised sound event detection using multiple scale input. Detection and Classification of Acoustic Scenes and Events (DCASE).‏
27. Ragni, A., Knill, K., Rath, S. P., & Gales, M. (2014). Data augmentation for low resource languages.‏
28. Jaitly, N., & Hinton, G. E. (2013, June). Vocal tract length perturbation (VTLP) improves speech recognition. In Proc. ICML Workshop on Deep Learning for Audio, Speech and Language (Vol. 117).‏
29. Park, D. S., Chan, W., Zhang, Y., Chiu, C. C., Zoph, B., Cubuk, E. D., & Le, Q. V. (2019). Specaugment: A simple data augmentation method for automatic speech recognition. arXiv preprint arXiv:1904.08779.‏
30. Salamon, J., & Bello, J. P. (2017). Deep convolutional neural networks and data augmentation for environmental sound classification. IEEE Signal Processing Letters, 24(3), 279-283.‏
31. Gouda, S. K., Kanetkar, S., Harrison, D., & Warmuth, M. K. (2018). Speech Recognition: Keyword Spotting Through Image Recognition. arXiv preprint arXiv:1803.03759.‏
32. Goddard, C. (2014). Interjections and emotion (with special reference to “surprise” and “disgust”). Emotion Review, 6(1), 53-63.‏
33. Shriberg, E. (2005). Spontaneous speech: How people really talk and why engineers should care. In Ninth European Conference on Speech Communication and Technology.‏
34. Cohn, M., Chen, C. Y., & Yu, Z. (2019, September). A Large-Scale User Study of an Alexa Prize Chatbot: Effect of TTS Dynamism on Perceived Quality of Social Dialog. In Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue (pp. 293-306).‏
35. Yoon, S. A., Son, G., & Kwon, S. (2019). Fear emotion classification in speech by acoustic and behavioral cues. Multimedia Tools and Applications, 78(2), 2345-2366.‏
36. Auditok - An audio/acoustic activity detection and audio segmentation tool - <https://github.com/ramya1782/auditok>
37. Sahidullah, M., & Saha, G. (2012). Comparison of speech activity detection techniques for speaker recognition. arXiv preprint arXiv:1210.0297.‏
38. McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., & Nieto, O. (2015, July). librosa: Audio and music signal analysis in python. In Proceedings of the 14th python in science conference (Vol. 8).‏
39. Bittner, R., Humphrey, E., & Bello, J. (2016, August). Pysox: Leveraging the audio signal processing power of sox in python. In Proceedings of the International Society for Music Information Retrieval Conference Late Breaking and Demo Papers.‏
40. Khalil, R. A., Jones, E., Babar, M. I., Jan, T., Zafar, M. H., & Alhussain, T. (2019). Speech emotion recognition using deep learning techniques: A review. IEEE Access, 7, 117327-117345.‏