I Know That Voice: Identifying the Voice Actor Behind the Voice

Lior Uzan and Lior Wolf
The Blavatnik School of Computer Science
Tel Aviv University
lioruzan@mail.tau.ac.il, wolf@cs.tau.ac.il

Abstract

Intentional voice modifications by electronic or non-electronic means challenge automatic speaker recognition systems. Previous work focused on detecting the act of disguise or identifying everyday speakers disguising their voices. Here, we propose a benchmark for the study of voice disguise, by studying the voice variability of professional voice actors. A dataset of 114 actors playing 647 characters is created. It contains 19 hours of captured speech, divided into 29,733 utterances tagged by character and actor names, which is then further sampled. Text-independent speaker identification of the actors based on a novel benchmark training on a subset of the characters they play, while testing on new unseen characters, shows an EER of 17.1%, HTER of 15.9%, and rank-1 recognition rate of 63.5% per utterance when training a Convolutional Neural Network on spectrograms generated from the utterances . An I-Vector based system was trained and tested on the same data, resulting in 39.7% EER, 39.4% HTER, and rank-1 recognition rate of 13.6%.

1. Introduction

Identifying the voice of a person who deliberately modifies his natural voice is a common forensic need. Criminals often disguise their voice electronically or non-electronically by means of whispering, modifying their natural pitch, pinching their nostrils, or placing a barrier between their mouth and the telephone or the recording device. To counter such modifications, one may try to identify the type of modification and then adapt the voice recognition method.

In this work, we study speaker identification of professional voice actors playing a variety of characters. The main question we pose is whether it is possible to identify the actor playing a character by employing voice samples of the actor playing different, deliberately distinguishable, characters. As we show, this is indeed possible, at least at a moderate level of accuracy.

The voice recognition engine we use is based on a deep Convolutional Neural Network (CNN) applied to the raw spectrograms. While spectrograms have fallen out of favor in speaker recognition, and were replaced by methods such as Mel-frequency cepstral coefficients (MFCCs) [25], the ability to create convolutions that sample raw information across both frequency and time makes them suitable for CNN analysis. We focus on wide networks, with many filters per layer, in order to capture minute differences between speakers.

In section 2 we present our data collection methods, in section 3 we present the Deep Learning and I-Vector extraction frameworks we used in the experiments, Section 4 describes the CNN and I-Vector experiments, followed by a summary in section 5.

1.1. Previous work

Voice disguise is related to the problem of voice imitation (mimicry) based spoofing. While spoofing aims to elicit false acceptances in authentication applications, voice disguise and other forms of voice modification provoke missed detections. Overall, it seems that imitation by electronic means is a vulnerability of speaker verification systems [5]. However such attacks can be detected [16, 3]. Human imitation, even done by professional imitators, increases the false acceptance rate by a very moderate amount [11].

Intentional voice disguise was studied in [17], where volunteering subjects altered their voices mainly by means of whispering, murmuring, changing their pitch, or mimicking accents. The baseline system employed a Gaussian mixture model (GMM) on top of MFCC. While with the normal to normal conditions, it had a false rejection rate of 0%, this increased to 39% when users were allowed to modify their voices. However, by simply changing the acceptance threshold to match disguised voices, the rejection rate dropped to 9%. Amateur voice modifications, such as those described above, can be detected with at least a moderate level of success [24]. When the modifications are detected, it is not clear how the models would be adapted in order to

identify the speakers.

In a recent large scale study [4], the researchers applied high-technology voice conversion methods [21] in order to electronically obscure a person's voice. The equal error rates (EER) of a standard GMM model with a universal background model increased from 9% to 34%, while an i-vector based PLDA system [10, 18] increased from 3% to 10%.

Deep Neural Network (DNN) on top of speech coefficients such as MFCC or perceptual linear prediction (PLP) [13] currently dominates the performance charts for speech recognition, although recently a comparable level of performance was achieved by employing kernel methods [14]. There have been attempts to match the performance of the DNN networks using CNNs on top of spectrograms [2, 9, 26], which results in networks similar to our network.

The usage of deep networks in speaker recognition is much less studied, despite earlier interest [23, 7, 32, 12], and the field is currently dominated by I-Vectors and PLDA [10, 18]. Recently, CNNs have been explored as a supplement or possible replacement to the Ivectors/PLDA algorithm. a DNN on top of a 1,600-dimensional vector, constructed by stacking 40-dimensional log filterbank energy features extracted for a block of 40 frames, was used for text-dependent speaker verification [31]. The obtained results were inferior to that of I-Vectors. Also a CNN has been used together with Ivectors to obtain improved results in speaker recognition in [22], but we take a more direct approach using only the CNN to predict speakers. Other attempts employed Boltzmann Machines [28, 27], obtaining preliminary success.

2. Data collection

Our experiments require a large scale dataset of disguised voices. As far as we know, we present the first such dataset. In this section we describe the process of collecting speech utterances from animated sitcoms, which is summarized in Figure 1.

Animated TV shows are dubbed in recording studios with high quality microphones, and each voice actor typically dubs multiple characters. For example, in episode 3 of season 13 Dan Castellaneta plays the characters Homer Simpson, Chinese Officer, Chinese General, Barney Gumble, Professor, and Man #2. We chose to extract speech from The Simpsons and Futurama TV shows because the actors voices on these shows use their natural voices, in contrast to shows such as South Park where the characters are pitch-shifted [29].

In order to segment the speech audio subtitle timecodes are used. To label and transcribe the utterances, episode transcripts are used. For each episode subtitles are obtained from opensubtitiles.org (`www.opensubtitles.org/`), transcripts from The Simpsons Archive (`simpsonsarchive.com/`) and The Internet Movie Script Database (`imsdb.com/`).

To simplify our assumptions on the channel variance in the data, and to present a more challenging task for our system, we assume nothing regarding the handsets used to record the audio. Furthermore, we decided to keep the utterances in their native 48khz sample rate, to minimize information loss in generated spectrograms.

To label and transcribe the utterances, we match subtitles to transcript lines of a character, then use the character name to tag the utterance. Matching between subtitles and transcript quotes were done with Natural Language Toolkit [8]. To overcome differences between subtitle and transcript sentences when matching, we found that a cutoff of 0.35 average Levenshtein distance per word was most effective. Only subtitle and transcript sentences of length greater than 4 words were considered for practical reasons.

Due to internal variance of character names in transcribed episodes, character names are manually matched to official character names from IMDB (`http://www.imdb.com/`). We eliminated characters that have more than one actor from our dataset, unless the character had a significant number of appearances and was played mainly by the same actor. In that case we separate the role in two.

Once the identifying and retrieving utterances is finished, the utterances are filtered to eliminate white noise [30]. 29,733 utterances were produced with this method, at a total length of 18.9 hours, with each utterance 2.29 seconds long, on average.

An attempt to use utterance transcriptions for phonetic alignment of speech using The Penn Phonetics Lab Forced Aligner (p2fa) [33] was made. This proved problematic; p2fa uses a phonetic dictionary to decide how text is pronounced, and it was limited in scope and inflexible to the accent richness and humorous pronunciations in the shows.

We finally generate spectrogram frames in a sliding window fashion per utterance, with window width 20ms and step 10ms. All spectrograms are generated with SoX [1]. Each spectrogram shape is 513x107 pixels. Any spectrograms created in a different shape are not used.

3. Recognition method

We perform experiments with two speaker recognition methods. A Convolutional Deep Neural Net applied as a classifier to the spectrogram images, and an I-Vectors method is implemented using the bob.spear framework [19, 6].

3.1. DNN architecture and training

We train a DNN, the renowned AlexNet [20], to classify the speaker identified speaking within the spectrogram. The overall architecture is shown in Figure 2. The input is an

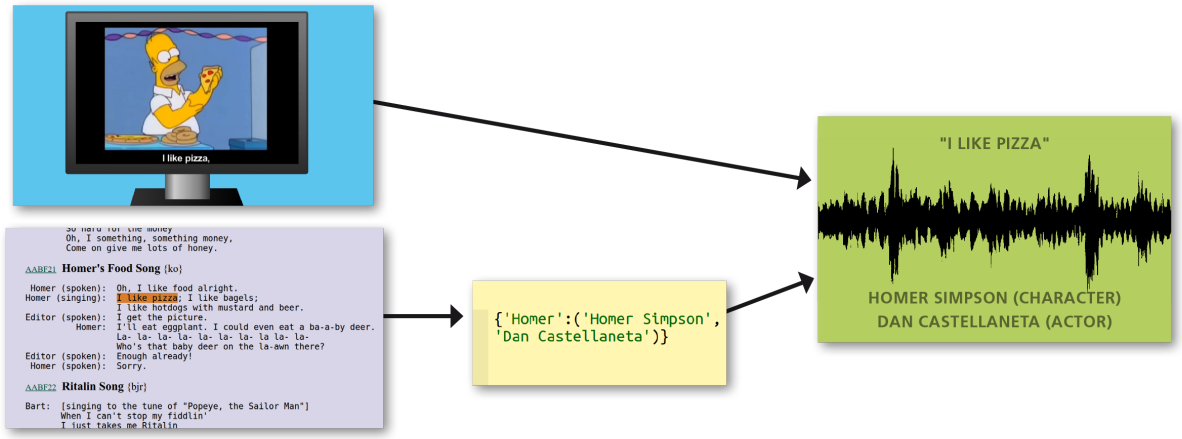

Figure 1. The data collection process. The audio we seek plays during an appearance of a subtitle. The subtitle is normalized, and located within a transcript of the episode. The name appearing in the transcript is looked up in a pre-processed dictionary containing the official IMDB names of actors and characters to identify the speaking actor and character. Finally, the time code from the subtitle is used to extract the utterance we identified.

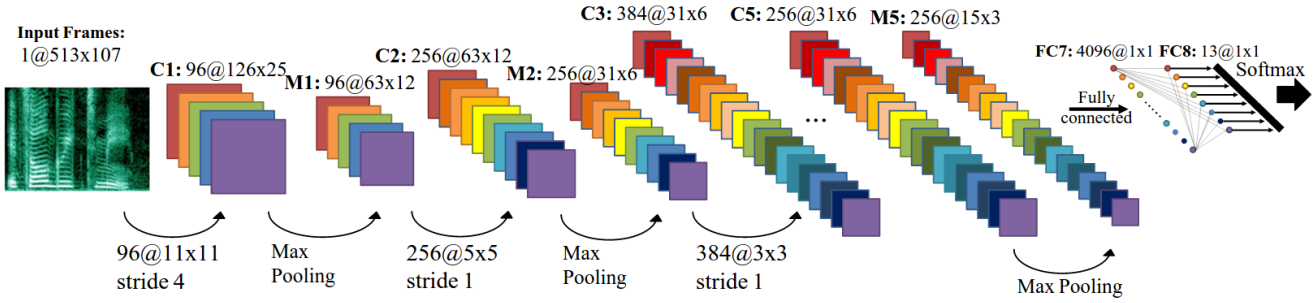

Figure 2. Outline of the architecture we have used in this work. Input is a single spectrogram, followed by interleaving sets of convolution/maximization pooling hidden layers, two fully connected hidden layers with randomized weight disabling (DropOut), and finally a softmax layer.

image of size 513 by 107- a spectrogram of a 20ms segment of an utterance. This is denoted by 1@513x107, since they are single channel images.

The input is given to a convolutional layer (C1). The filters are convolved across the image in a horizontal and vertical stride of 4. Rectified Linear Unit activations are applied to the convolution results in each step. This layer encodes patterns across frequency and time dimension in the spectrogram. The resulting feature maps, or features as we will refer to them from now on, are then fed to a max-pooling layer (M1) which takes the max pixel value over 1-overlapping 3x3 spatial neighborhoods for each feature separately.

A Local Response Normalization layer is applied. In this layer, each input is divided by $(1+(\alpha/n)\sum_i x_i^2)^\beta$ where n is the local region size, and the sum is taken over the region centered at that value. In this case, the local regions extend across nearby feature maps (they have no spatial extent). We set $n = 5$, $\alpha = 0.0001$, $\beta = 0.75$.

This is followed by another convolutional layer (C2). This layer is grouped randomly into 2 groups, where each output is connected only to inputs from its own group. In

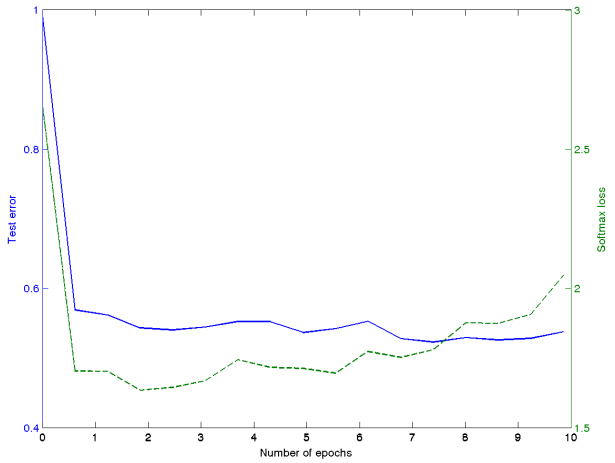

Figure 3. CNN training: the test error as a function of the training epochs (1 epoch = 1 complete pass on Train set). We stopped training at epoch 10 for practical reasons.

each step of the convolution, the filter is applied to and summed across all given inputs. As before, ReLU/max-pooling/LRN are applied in the same way.

3 Convolutional layers with ReLU non-linearity are applied, the dimensions described in the picture. C3 and C4 have no grouping, whereas C5 is with grouping like C2. Atop C5 is another max-pooling layer (M5) which considers 3x3 spatial neighborhoods with 1 pixel overlap with ReLU applied.

Next, is a fully-connected layer (FC6), whose input for every unit is all the outputs from the previous layer. Each unit proceeds to output $\sum_i w_i x_i$ where w_i represent the weights given to each input x_i. During image processing, each weight is made zero or left the same in probability 0.5. This is called a DropOut and the weights are re-selected every image. All activations are multiplied by 2. The process of FC/ReLU/DropOut is repeated with another fully-connected layer (FC7). The output from FC7 is passed on to a last fully-connected layer (FC8) with 13 outputs, which is passed to a Softmax loss layer, which is the multinomial logistic loss of the softmax of it's inputs.

The weights are trained with with Stochastic Gradient Descent and Back Propogation . The implementation is done using the Caffe [15] Deep Learning library. In order to allow complete reproducibility, we will publish the code of the entire system.

3.2. I-Vectors implementation

We use the bob.spear I-Vector implementation for speaker recognition tool chain. First, spear appliesan energy based VAD (in addition to our first VAD session) on the voice data. MFCC feature extraction is done, then UBM training. Next, subspace training I-Vector extraction is done, followed by Whitening, length normalization, and PLDA. Client models are enrolled by averaging utterance features. Scores are computed. In this stage, the models are compared to lists of probe features and a similarity score is computed for each pair of model and probe features. Finally, evaluation is done. We use Half Total Error Rate (HTER) score defined by $\frac{FAR+FRR}{2}$. We will be releasing our configuration files of bob.spear for maximal reproducibility.

4. Experiments

In order to perform our experiments, we select from within our dataset all actors that play more than 5 characters. There are 13 such actors in total, and we use the rest as auxiliary data, in ways that are discussed below. This selection stems from the need to have sufficient character diversity per actor in the train and test sets. Other voice actors, even those associated with many samples are not used during the main experiment. For example, Katey Sagal, part of Futurama's main cast, plays only one character- Turanga Leela, and is not included in the training set; Billy West, another cast member, plays Philip J. Fry, Prof. Hubert J. Farnsworth, Dr. Zoidberg and many more, and is included.

For each actor, we split their utterances by character into Test and Train sets, i.e., a specific character appears only in the train set or in the test set. 80% of the characters are taken as Train, and 20% as test. As a secondary criterion, we also make an attempt to have approximately 80% of the utterances in the Train set. This is done by employing integer programming for the characters of each actor.

We did not run separate experiments for male and female actors, because there exists some cross-gender acting in the dataset. For example, the voice of Bart Simpson is actress Nancy Cartwright. Experiment parameters and results are summarized in the following subsections, and in Figure 4.

4.1. CNN Experiment

Using the Caffe framework, we train a Convolutional Neural Net on the spectrograms we extracted in Data Collection phase. We iterate over the Train set for half an epoch, then evaluate our net on the Test set, and repeat. Figure 3 shows convergence of the test error.

Once the training is complete and all frames of Test set are classified, an utterance is classified by selecting the majority class out of it's frame classifications. The softmax output of the CNN is taken as a score vector scoring each of the possible classes. By marking the correct and incorrect classifications over this vector, then over all Test set results, the FAR, FRR, DET curve, EER and HTER of the CNN system are calculated. On the per spectrogram classification task, we achieved multi-class classification accuracy of 47.7% (52.3% error rate). After voting was employed, accuracy for utterances was 63.5% (36.5% error rate). The EER,

HTER of the system were 17.1%, 15.9%, respectively.

The above experiment details recognition results obtained in a supervised manner on the set of 13 voice actors. In order to evaluate whether the CNN learned a general representation, we also performed an unsupervised transfer-learning experiment.

In this unsupervised experiment, each spectrogram was represented by a vector of 4096 dimensions obtained as the activation of the CNN in the layer before the classification layer (FC7). The entire utterance was represented by the mean activation of all the frames. This is perhaps the simplest way to represent a set of vectors as a single vector. A more sophisticated representation might lead to improved results.

A benchmark based on the 101 actors with less than 5 characters in the datasets was constructed. It contains 2000 pairs of utterances of the same voice actor playing two different characters, and 6000 pairs of utterances by different actors. Using the L2 distance of the two vectors of mean FC7 activations as a measure of dissimilarity between the two utterances in each pair, we obtained an AUC of 68% (EER 37%). A simple baseline which predicts simply by the similarity in the gender of the actors in the pair of utterances achieves a lower AUC of 58% (EER 44%), which indicates that some non-trivial character-independent properties of the voice are encoded in the representation.

4.2. I-Vectors Experiment

We used bob.spear framework for extraction and evaluation of I-Vectors from our data, with the points of the tool chain specified in the previous section. In the experiment, we use the other 101 actors (the ones who play less than 5 characters), 4584 utterances total, to train the UBM. As in the CNN experiment, we use the same 80%/20% split by character for development and evaluation, respectively.

For development, 4665 utterances from Train set are used as probe features compared against speaker models. For evaluation, all 4046 utterances in Test set are used as probe features. 13,541 utterances from Train set are used for ZT-Norm score normalization (9871 for T-Norm , 3670 for Z-Norm). The dimension of the UBM was selected to be 8, and I-Vectors of dimension 50 were used. These parameters were chosen because of short utterance length.

The results of the experiment are 33% EER, 32.5% HTER on the development set and 39.7% EER, 39.4% HTER on the evaluation set. Out of the 13 trials done with each probe, by taking the class with maximal score as the prediction, multi-class classification accuracy was measured to be 13.6% (86.4% error rate).

5. Summary

Voice disguise is a common challenge for forensic voice recognition systems. However, the existing experiments are limited in their scope, or limited to electronic disguise. In this work, we extract from animated sitcoms a relatively large dataset of disguised voices. We compare the I-Vector method, which is currently the leading method in voice recognition, to a novel deep neural network solution.

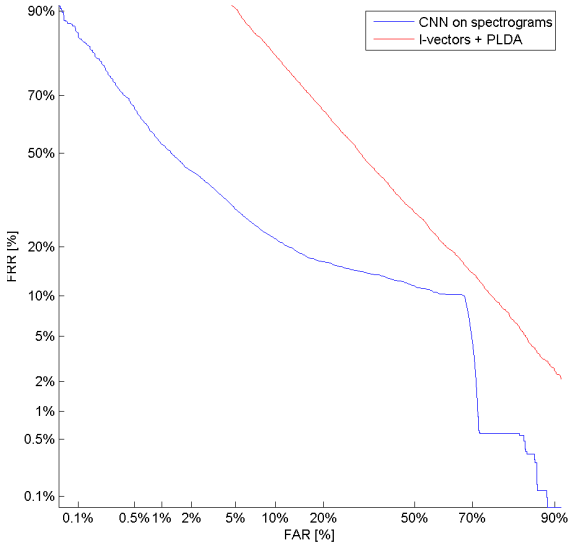

Figure 4. DET curves for CNN and I-Vectors experiment.

In our experiments, the CNN outperforms the I-Vector based system in a significant margin. This could be due to the fact that I-Vectors are usually beneficial on systems with a larger number of speakers, or the relatively short length of utterances (2.29 seconds on average). Further experimentation is required.

In the future, we seek to experiment with different CNN architectures, I-Vector parameters, and continue promoting the exploration of speaker variability introduced by the characters of professional voice actors.

References

[1] Sox - sound exchange software, version 14.4.1. `http://sox.sourceforge.net/`, 2013. [Online; accessed 01-December-2014].

[2] O. Abdel-Hamid, A. Mohamed, H. Jiang, and G. Penn. Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 4277–4280, March 2012.

[3] F. Alegre, A. Amehraye, and N. Evans. Spoofing countermeasures to protect automatic speaker verification from voice conversion. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 3068–3072, May 2013.

[4] F. Alegre, G. Soldi, and N. Evans. Evasion and obfuscation in automatic speaker verification. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 749–753, May 2014.

[5] F. Alegre, R. Vipperla, N. Evans, and B. Fauve. On the vulnerability of automatic speaker recognition to spoofing attacks with artificial signals. In *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pages 36–40, Aug 2012.

[6] A. Anjos, L. E. Shafey, R. Wallace, M. Günther, C. McCool, and S. Marcel. Bob: a free signal processing and machine learning toolbox for researchers. In *20th ACM Conference on Multimedia Systems (ACMMM), Nara, Japan*. ACM Press, Oct. 2012.

[7] Y. Bennani and P. Gallinari. Connectionist approaches for automatic speaker recognition. In *ESCA workshop on speaker recognition*, 1998.

[8] S. Bird, E. Klein, and E. Loper. *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly, Beijing, 2009.

[9] G. E. Dahl, T. N. Sainath, and G. E. Hinton. Improving deep neural networks for LVCSR using rectified linear units and dropout. In *ICASSP*, pages 8609–8613. IEEE, 2013.

[10] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P.Ouellet. Front-End Factor Analysis For Speaker Verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 13(4):788–798, May 2011.

[11] R. G. Hautamki, T. Kinnunen, V. Hautamki, T. Leino, and A.-M. Laukkanen. I-vectors meet imitators: on vulnerability of speaker verification systems against voice mimicry. In *Interspeech*, 2013.

[12] L. P. Heck, Y. Konig, M. K. Sönmez, and M. Weintraub. Robustness to telephone handset distortion in speaker recognition by discriminative feature design. *Speech Commun.*, 31(2-3):181–192, June 2000.

[13] G. E. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Process. Mag.*, 29(6):82–97, 2012.

[14] P.-S. Huang, H. Avron, T. Sainath, V. Sindhwani, and B. Ramabhadran. Kernel methods match deep neural networks on timit. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 205–209, May 2014.

[15] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[16] Q. Jin, A. R. Toth, A. W. Black, and T. Schultz. Is voice transformation a threat to speaker identification? In *ICASSP*, pages 4845–4848, 2008.

[17] S. S. Kajarekar, H. Bratt, E. Shriberg, and R. de Leon. A study of intentional voice modifications for evading automatic speaker recognition. In *Speaker and Language Recognition Workshop, 2006. IEEE Odyssey 2006: The*, pages 1–6, June 2006.

[18] P. Kenny. Bayesian Speaker Verification with Heavy-Tailed Priors. In *Proc.˜Odyssey-10*, 2010.

[19] E. Khoury, L. El Shafey, and S. Marcel. Spear: An open source toolbox for speaker recognition based on Bob. In *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2014.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[21] D. Matrouf, J.-F. Bonastre, and C. Fredouille. Effect of speech transformation on impostor acceptance. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 1, pages I–I, May 2006.

[22] M. McLaren, Y. Lei, N. Scheffer, and L. Ferrer. Application of convolutional neural networks to speaker recognition in noisy conditions. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[23] J. Oglesby and J. Mason. Optimisation of neural models for speaker identification. In *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on*, pages 261–264 vol.1, Apr 1990.

[24] P. Perrot and G. Chollet. Helping the forensic research institute of the french gendarmerie to identify a suspect in the presence of voice disguise or voice forgery. In A. Neustein and H. A. Patil, editors, *Forensic Speaker Recognition*, pages 469–503. Springer New York, 2012.

[25] L. R. Rabiner and B.-H. Juang. *Fundamentals of speech recognition*, volume 14. PTR Prentice Hall Englewood Cliffs, 1993.

[26] T. N. Sainath, B. Kingsbury, A. Mohamed, G. E. Dahl, G. Saon, H. Soltau, T. Beran, A. Y. Aravkin, and B. Ramabhadran. Improvements to deep convolutional neural networks for LVCSR. *CoRR*, abs/1309.1501, 2013.

[27] M. Senoussaoui, N. Dehak, P. Kenny, R. Dehak, and P. Dumouchel. First attempt of boltzmann machines for speaker verification. In *Odyssey: The Speaker and Language Recognition Workshop*, 2012.

[28] T. Stafylakis, P. Kenny, M. Senoussaoui, and P. Dumouchel. Preliminary investigation of boltzmann machine classifiers for speaker recognition. In *Odyssey: The Speaker and Language Recognition Workshop*, 2012.

[29] M. R. Stone. South park blog september 05, 2001. `http://southpark.cc.com/blog/2001/09/05/390-breayle-september-05-2001`, 2001. [Online; accessed 01-December-2014].

[30] L. N. Tan, B. Borgstrom, and A. Alwan. Voice activity detection using harmonic frequency components in likelihood ratio test. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 4466–4469, March 2010.

[31] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez. Deep neural networks for small footprint text-dependent speaker verification. In *ICASSP*, 2014.

[32] B. Yegnanarayana and S. P. Kishore. Aann: An alternative to gmm for pattern recognition. *Neural Netw.*, 15(3):459–469, Apr. 2002.

[33] J. Yuan and M. Liberman. Speaker identification on the scotus corpus. In *Acoustics*, 2008.