Journal of the Brazilian Computer Society

RESEARCH Open Access

Theoretical learning guarantees applied to acoustic modeling

Christopher D. Shulby[1,2*], Martha D. Ferreira[2], Rodrigo F. de Mello[2] and Sandra M. Aluisio[2]

Abstract
In low-resource scenarios, for example, small datasets or a lack in computational resources available, state-of-the-art deep learning methods for speech recognition have been known to fail. It is possible to achieve more robust models if care is taken to ensure the learning guarantees provided by the statistical learning theory. This work presents a shallow and hybrid approach using a convolutional neural network feature extractor fed into a hierarchical tree of support vector machines for classification. Here, we show that gross errors present even in state-of-the-art systems can be avoided and that an accurate acoustic model can be built in a hierarchical fashion. Furthermore, we present proof that our algorithm does adhere to the learning guarantees provided by the statistical learning theory. The acoustic model produced in this work outperforms traditional hidden Markov models, and the hierarchical support vector machine tree outperforms a multi-class multilayer perceptron classifier using the same features. More importantly, we isolate the performance of the acoustic model and provide results on both the frame and phoneme level, considering the true robustness of the model. We show that even with a small amount of data, accurate and robust recognition rates can be obtained.

Keywords: Acoustic modeling, Convolutional neural networks, Shallow learning, Speech recognition, Statistical learning theory, Support vector machines

Introduction

Most speech-processing applications rely on acoustic models which build the bridge between the audio signal and its phonetic transcription. After a sentence prompt has been phonetically transcribed, the task of learning which phonemes belong to certain audio segments is far from trivial, but essential to modern speech applications, since problems at this stage are likely to propagate, even with the help of a robust language model. For automatic speech recognition (ASR), it is essential that phonemes are transcribed completely and in the correct order; in other words, all phonemes in sequence should be recognized. Normally, an ASR system consists of an acoustic model, a pronunciation model and a language model. The acoustic model attempts to match the signal to its probable phoneme(s), and the posterior values are used in the pronunciation model to find the most likely words for each segment, which works as an input to the language model to determine the probable chunks or phrases.

ASR results are generally reported with all of these components included; however, it is important to accurately model the acoustic properties so that errors do not propagate in the other models. Reliable phoneme-level error detection is still a great need [1, 2] for automatic pronunciation training where a pronunciation model can actually damage the desired results.

Recent work in state-of-the-art speech recognition has benefited greatly from deep learning techniques. While results have become significantly better, it is fair to assume a high risk for overfitting of the data and difficult to provide solid learning guarantees. Beyond these obvious issues, deep learning algorithms are notoriously poor performers on small datasets or datasets with noise. We believe this is due to the lack of generalization capabilities of such networks and that this could be avoided if the care to establish learning guarantees was taken during development phases.

The long time state-of-the-art Gaussian mixture models - hidden Markov models (GMM-HMM) acoustic

*Correspondence: cshulby@icmc.usp.br
[1]Samsung SIDI Institute, Rua Aguaçu, 171, 13098-321 Campinas, SP, Brazil
[2]Institute of Mathematical and Computer Sciences, University of São Paulo, Avenida Trabalhador São-Carlense, 400, 13566-590 São Carlos, SP, Brazil

models have become less popular since the deep learning movement in the last 5 years. In many cases, deep learning has significantly reduced the error rates of ASR systems [3]. GMM-HMM models have the advantage that "good" models can be built using relatively little data (about 10 h of speech) and can be trained quickly (in a couple of hours at most). They also do not require the enormous corpora needed for training deep neural networks [4]. The trade-off in quality vs. quantity seems to be somewhere on the spectrum between these two methods for most applications. Besides the training time, the real problem lies in the availability of training data for under-resourced languages or specific applications. On the other hand, the complexity of neural network models, especially in brute-force approaches, can be a problem, since they can require huge corpora. Recent corpora released for deep models contain over a thousand hours of audio data [5, 6] and can take months to train [7]. More recently, multi-conditional training (MCT) and similar techniques have made their way to state-of-the-art implementations [8, 9]. In an effort to account for noise, the database is exponentially augmented which requires even more training time (several months with an immense GPU-cluster infrastructure). While these enormous datasets and augmentation techniques usually implement regularizing techniques like dropout and bottlenecking, it is clear that they cannot provide solid learning guarantees [10].

In this paper, we propose a method, useful for low-resource training environments, which provides solid learning guarantees, using a shallow-shallow-convolutional neural network (CNN)-hierarchical tree support vector machine (HTSVM) architecture. This architecture combines several techniques which have already shown success in ASR tasks. We have done this by taking a knowledge-driven slant on the typical machine learning approach. First, we take advantage of the well-known CNN ability to deal with images [11, 12] which extracts the features from a spectrogram generated from the audio signal. It is important to note that we use the CNN only as an extractor and prove that we can extract meaningful features for classification in a robust way. The selection of the CNN is in the spirit of a knowledge-driven manual classification task where trained specialists in acoustics and laboratory phonetics [13] are able to classify phonemes even when the spectrogram is fairly noisy. This is due to the visual representation given to us via fast Fourier transform.

Humans are able to see a "picture" based on the relative heights and intensities of formants and energy concentrations across spectrogram frequencies. While the actual frequency values can vary greatly from speaker to speaker, depending on multiple factors including the length of the vocal tract, the "picture" is always similar and recognizable to a human speech scientist. Therefore, it makes sense that a computer vision algorithm would be suitable in this case. In previous years, a lot of feature engineering was used for GMM-HMM acoustic models which attempted to represent the spectrogram using features like mel-frequency cepstrum coefficients (MFCC), band filter banks, pitch, and strength, among other features. Here, our goal is to treat this as a computer vision problem where certain visual clues are useful, depending on the context of the spectrogram. The CNN makes for an interesting extractor, especially with the down-sampling of features acquired through max pooling. This is actually inspired by the mammal's primary visual cortex as described in [14], where the orientation of selective simple cells with overlapping local receptive fields, or sub regions were identified. The network operates locally [15] convolving with filters over an image and is able to recognize similar avatars, regardless of their actual position. We believe that the use of filter masks could be another useful feature which the CNN offers to phoneme recognition. These are some advantages over other neural networks which are unable to deal with this type of translational variance of local distortions from the input, as is pointed out in [16]. This gives the CNN a higher level of robustness against distortions due to speaker variability and noise, both of which are much needed in the current state-of-the-art ASR applications.

We classify the extracted features using a hierarchical tree with predefined articulatory groups where each node contains an SVM. The articulatory groups were inspired by the hierarchical grouping suggested by Peter Ladefoged [13]. The SVM was selected because it provides supervised learning guarantees due to the Vapnik–Chervonenkis (VC) theory [17] and the principle of structural risk minimization. The hierarchical tree structure was chosen for two reasons: (i) to overcome the unbalanced data problem and (ii) to deal with the sample and feature sizes generated in pre-processing and feature extraction phases. We believe that a good way to work with these problems is to combine a knowledge-driven recipe with a machine learning algorithm, thus simplifying the classification space while saving time, since a SVM with a large amount of samples can become prohibitively costly to train. When less classes are used for classification, we can more confidently implement data augmentation techniques to balance the dataset based on specific, discriminative features. In other words, a two or three class problem creates a simpler hypothesis space than a 40 class problem. Also, in higher nodes we can make a better sample selection and use a voting system or similar approaches, whereas in the lower nodes, smaller problems with multiple machines parallelizes the training set which has been divided up among discriminative phonemic classifications. These strategies for the CNN and SVM optimize the power of each, thus adherent to the paper's goal, to present an acoustic model with high accuracy given limited resources, both in the senses of computational processing power as well as training data.

In this paper, we will first discuss the related work in phoneme recognition, divided into several subsections, in an attempt to present a triage of results to better understand the state of the art in this field, being (1) CNN, (2) HTSVM, and (3) studies which are most impacted by acoustic model results.

Then, we will go into the relevant details about the corpus used for this study and explain the procedures used when treating the raw data and the spectrograms generated from it. After that, we will discuss the feature extraction via CNN and classification done by the HTSVM used to produce the following two sections which present experiments and results and then the convergence analysis of the models. Finally, we will have a discussion to sum up this work followed by the conclusion and speculations about future work.

For the interested reader, this work expands on the study presented in [18]. We have added a great many details about the process which were not present in the conference paper and have provided statistical learning guarantees via convergence analysis to evidence the robustness of this method.

Related work

Phoneme recognition is not a new task as explored in [19–21], but greater success has been achieved only in the last 5 years [3] and still remains far from a solved problem. One of best known studies proving the capabilities of CNN for this task is [22], where a hybrid CNN-HMM model using local filtering and max pooling in the frequency domain is proposed to deal with the translational invariance problem present in other DNN. In [23], the optimal CNN architecture is explored including the number of convolutional layers and hidden units needed, as well as the optimal pooling strategy and feature type for the CNN and the best results are achieved using large corpora (300–400 h) and a two-convolutional layer DNN with with 424 hidden units and four fully connected layers with 2048 hidden units each, followed by a softmax layer with 512 output targets.

State of the art using CNN for speech recognition

Abdel-Hamid, et al. [15] revisits the issue of robustness in speaker and environment variation with a CNN-HMM where the HMM deals with the issue of distortions of over time, while the CNN convolves over the frequency to take advantage of its ability to deal with variation among speakers in this domain. This study serves as a baseline on TIMIT for the state-of-the-art deep CNN with a 21.6% PER (phone error rate). It should be noted that the network in that paper initializes pre-trained weights from another network trained on the much larger Google voice search database with 18 h of speech data. The authors do not explain how the strings are generated for comparison with the original TIMIT annotations but one can assume that some post-processing is done to reduce the frames to phonemes. A general issue, for proper comparison of acoustic models, is that state-of-the-art methods, which use large deep networks with thousands of units and often thousands of hours of training data, do not show frame-level results as in [15, 22–27]. As outlined in [3], they often employ a number of resources like pronunciation models, language models, and other post-processing/data smoothing techniques which are of great help for the end speech-recognition applications; however, they also mask the true recognition accuracy achieved by the acoustic model.

State-of-the-art using HTSVM for speech recognition

Hierarchical classification has not been often applied to the phoneme recognition task but some notable exceptions exist, like [28–31]. In [28], phoneme classification is treated as an optimization problem where a hierarchical tree structure, which divides groups of phonemes as nodes in the tree. The authors found that the tree would tolerate small tree-induced errors while avoiding gross errors as a standard multi-class classifier would be prone to commit. In the last 3 years, the HTSVM has been employed using data from speech corpora and applied to a phoneme recognition task as presented in [29–31]. In [30], an experiment on stop and fricative consonants using the Lithuanian LTDIGITS corpus, containing over 25,000 phonemes, is presented. The most important findings were a 3% gain in the overall accuracy, a total of 68.4%, while reducing 52–55% of the computational time taken for classification with SVM. In [29], a Tamil corpus of repeated words was developed and 2400 phoneme instances were tested resulting in about 67% total accuracy on obstruent and sonorant sounds using MFCC features. In [31], a study on the TIMIT corpus presents MFCC classification results for each phoneme and major confusions classified by SVM as well. Most of the phonemes fall between the accuracy range of 30% and 60%. The authors point out that due to the multiple dialects present in the TIMIT corpus, many phonemes are pronounced similarly to others depending on the speaker, increasing the confusion rate for similar phonemes, where the SVM was not capable of efficiently classifying phonemes in the lower nodes of the tree.

State of the art in acoustic modeling

We will divide this topic in two lines of research and three subsections: (i) forced alignment (FA) and (ii) phone error rate (PER), and related to PER PER frame error rate

(FER) will have its own section. We want to draw special attention to the FA results since these experiments are not impacted by a language model; still, they do use pronunciation models, which are known to be essential for producing good results. So the division is done between FA results and metrics for acoustic modeling. In scope of this this section, we focus on stand-alone methods which present a training set and a validation set to generate results for the acoustic model without pretraining from larger databases or great adaptation techniques.

Forced alignment (FA)

This section is indirectly related to acoustic modeling, but since it is free of post-processing and great data manipulations, it is an interesting area to look at from our perspective. This is a logical choice because we can assume that without errors generated by a language model and assuming that the pronunciation model is well designed, FA results would present the best results possible for an acoustic model, since the actual target orthography is presented a priori. Phonetic alignments can be generated in one of two ways: (i) manually, usually with the help of specialized software or (ii) automatically, as is done with FA. Manual alignment guarantees a high level of quality but is not free of subjectivity and is a high cost activity. Normally, at least two specialist transcribers are required to generate a kappa score or something similar and it is time consuming. According to [32], manual transcription requires more than 13 h/min of audio. Often, researches opt for automatic solutions like FA. One the most used tools in the industry is HTK's HVite [33]. Studies like [34] show that forced alignment can be as accurate as 75.09% for errors of less than or equal to 10 ms and 93.92% for errors of less than or equal to 20 ms for English. These results are interesting because they give some insight to the actual frame accuracy of these acoustic-pronunciation model systems and are both reliable and reproducible.

Phone error rate

PER is the industry standard for measuring the accuracy of acoustic models and is calculated by the the Levenshtein distance [35] where the number of insertions, deletions, and substitutions are added and divided by the total number of phonetic units in the string.

Often, word error rate (WER) is calculated in the same way using the number of recognized words in a sentence; however, good WER results come from a good tandem of both acoustic and language models. This task is still far from resolved but has enjoyed great improvement in the last couple of years from 27.3% in 2009 [36] to 17.7% in 2013 [37], both tested on the TIMIT database. Again, it is important to point out that the results presented there are reported for entire systems and not for the acoustic model in isolation, also [37] uses a pre-trained network. Since it is difficult to find reliable state-of-the-art phoneme recognition results which do not use a great amount of data isolate the acoustic model, the next subsection presents FER.

Frame error rate

Some studies do report FER which represents the true accuracy of the acoustic model. In [38], a hybrid CNN and layer-fused multi-layer perceptron (MLP) with inputs of 11 frames of 25ms (step of 10 ms) as context from the TIMIT database and presents both PER and FER results, where the best network, with 1024 neurons in the input layer and 512 in the hidden layer, was able to obtain a FER of 43.04%. In [39], a hierarchical broad-phoneme MLP-HMM hybrid classifier was used with window widening which achieved impressive results. For a 90 ms window the FER was 61% using their past-future method that achieved 42% using 170 ms and 17 frames for context training every other frame along the right and left side of the central frame for that period. It should be noted that these experiments were done with the full 61 phoneme set and results for smaller windows with 39 phonemes are not presented.

Dataset, features, and machine learning algorithms

In the following sub-sections, the HTSVM architecture used in this work will be explained, step by step, from the corpus used, pre-processing steps and labeling to the feature extraction, classification, and finally post-processing procedures. One of the goals of this paper is to be as transparent as possible in order to better understand what is required to build robust acoustic models. The full pipeline, which will be explained in detail throughout this section, can be seen in Fig. 1.

Dataset: TIMIT automatic segmentation

TIMIT corpus

The TIMIT corpus, developed by Texas Instruments and the Massachusetts Institute of Technology [40], is a speech corpus, meant to be used for speech research and to serve as a gold standard for results comparison. TIMIT contains phonetically balanced prompted recordings of 2342 unique sentences (two dialect sentences (SA), 450 phonetically compact sentences (SX) and 1890 phonetically diverse sentences (SI)) from 630 speakers of eight major dialects of American English, each reading ten phonetically rich sentences to total 6300 utterances (5.4 h). The full training set contains 4620 utterances. The test set contains 1344 utterances from 168 speakers. With the exception of SA sentences which are usually excluded from tests, the training and test sets do not overlap and follow the suggested corpora splits outlined in [40]. TIMIT is considered a "balanced" corpus with respect to the distribution of phonemes and triphones found in the English language but it also follows a typical Zipf curve, as one

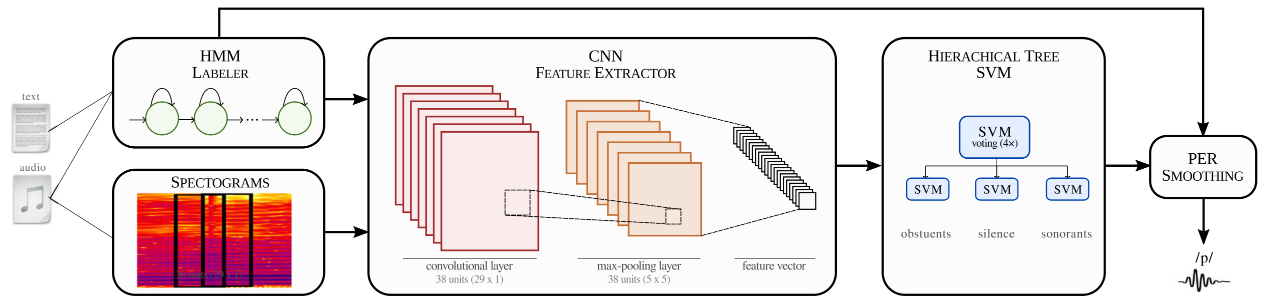

Fig. 1 *CNN-HTSVM architecture defined for the experiments*

would expect to find in a speech corpus, where some phoneme sequences dominate a large portion of the samples and a large number of phonemes are less frequent. From a machine learning perspective, this is an unbalanced dataset. Normally, as we have done in this paper, the phone set is collapsed into 39 monophones as suggested in [41] for training and testing. The decision to collapse them from the beginning was made in order to make the hypothesis space less confusing with such little data. The data is considered "clean" and has no great variations in amplitude or clipping.

TIMIT automatic segmentation

In order to build a full pipeline for low-resourced language situations, the label generation was delegated to an automatic labeler, pre-trained with only one hour of data using HTK [33].

To improve the performance of this tool, some adjustments were made, mainly a rule-based script was created to generate multiple pronunciations for each word in the pronunciation dictionary, as shown in Table 1. This was done to account for co-articulation and the eight dialects used in the TIMIT database and some light manual revision of difficult cases was done.

Then, the labeler was trained using FA provided by HTK's HVite tool on the rest of the TIMIT training set.

Spectrogram images

For each audio file, sampled at 16 kHz, spectrograms were created by sox [42], using the fast Fourier transform (FFT) from time windows of 25 ms as Hann windows with a stride of 10 ms. This means that each window originally consisted of 400 samples (25 ms $\times$16kHz). The images were then resized to 5 $\times$ 128 pixel images resulting in a DFT size of 254. As explained earlier, this was done in order to reduce the number of features extracted to a manageable dimensionality while prioritizing the frequency domain (which is simply reduced in the number of pixels equally over the entire image) and not the time domain since the time domain is handled by the HMM labeler. When a sentence ended in a number of milliseconds which is not divisible by 25, the last frame was squeezed into the penultimate frame (in all cases, this was silence). This process yielded 1,447,869 images for training and 482,623 for testing.

Feature extraction

CNN is a deep learning technique, which presents good results in several domains [16], including speech processing [15, 22, 23].

As mentioned earlier, the biological motivations prompted us to employ a CNN to extract relevant features from the spectrogram images, conserving only the most important information. The advantage of the CNN in this application is the identification of local recurrence information and its invariant translation in data [11, 43, 44]. CNNs are tolerant to distortion as they combine local receptive fields, shared weights, and spatial sub-sampling. All three of which are useful in phoneme recognition [22]. The trick comes in balancing the spatial resolution reduction with the representational richness of the images in

Table 1 Example excerpt from the augmented pronunciation model

Word	Transcription
Your	CH AO R
Your	CH ER
Your	CH OW R
Your	CH UH R
Your	JH AO R
Your	JH ER
Your	JH OW R
Your	JH UH R
Your	Y AO R
Your	Y ER
Your	Y OW R
Your	Y UH R

order to generate the most useful feature maps at a low classification cost with high accuracy [11]. We perform this in two ways:

1 We rescale the image to a smaller size. The goal was to find the smallest images possible, while still preserving the most important information for phoneme classification. The a priori decision was that the frequency domain was the most important since the time domain is not handled here[1]. Some experiments were done with small portions of data (about 10,000 samples) to arrive at the final size of 5×128 pixels. We also experimented with several window types, especially those which least distort the edges like Hann and Blackman-Harris windows. The final decision was to use the Hann window after experiments.
2 We optimize the feature maps by searching for the best sized masks. The size of the masks and number of neurons were estimated according to the adapted false nearest neighbors (FNN) algorithm proposed in [45]. We generated the best five configurations found by the algorithm and then selected the one with the largest mask size and fewest neurons. This selection was mostly systematic although the decision between the final configurations was slightly subjective as we preferred larger mask sizes to fewer neurons where the difference was small. This "feeling" was guided by the logic that the largest avatar which best represents the data is the best choice for strong generalizations. The final configuration is composed of 38 convolutional units with a 29×1 mask size. In the case of the sub-sampling layer, max pooling was used with 38 units and sized at 5×5, without overlapping. ReLU (rectified linear unit) was applied as the activation function due to its widespread adoption in the literature. This function avoids negative values and maintains the scale of output values. The CNN was trained using the Keras [46] package developed with the TensorFlow library. The 988 feature dimensions were generated using half or "same" padding as $\left(ceiling\frac{128\times5}{5\times5}\right) \times 38 = 988$. Where 128×5 is the size of the input images, 5×5 is the pooling size and 38 is the number of neurons used. Observe that the half padding always rounds up using the ceiling function.

We should also mention that we generated only single frames for classification and not nine or eleven for context as in other studies [38, 47]. We leave those experiments for future work on purpose, because our goal here is to focus on the usefulness of the CNN as a feature extractor for small datasets, contrary to the popular belief that the CNN requires a lot of data. To satisfy the readers curiosity, after 30 training epochs, the CNN settled at a cross-entropy loss of 1.0533. Also, it took less than half an hour to execute the CNN on a computer with 8 GB of RAM and conventional hardware (Intel i7). We believe this information is important because it shows that this careful technique can be applied even in low-resource scenarios.

Classification

The proposed method takes advantage of the machine learning techniques used for CNN and SVM and attempts to improve accuracy and minimize their disadvantages with a knowledge-based hierarchical tree structure.

The features produced by the CNN were classified using a SVM, since it has been used in literature combined with CNN and has provided good results in many domains. In addition, SVM provides a strong learning guarantee according to statistical learning theory (SLT) and large-margin bounds [17, 48]. The SVM parameters were found empirically after several experiments. The selected kernel for final experiments was a fourth-order polynomial kernel with coef0= 1 (as a non-homogeneous kernel) and a cost $C = 10,000$. As in other studies on natural language processing problems, like [49, 50], we found a polynomial kernel to be more useful for this task than a radial basis function (RBF) kernel. This is not surprising since the Zipf-like class distributions for these tasks are similar. For this task, namely speech recognition on the TIMIT dataset, we have two main problems to solve. First, the number of CNN extracted features combined with the number sample frames, since the SVM training time increases quadratically as the number of examples increases. The second problem, is the unbalanced nature of the dataset, a common characteristic for most speech corpora.

For the first issue, the hierarchical structure is what makes the SVM a viable option. The cost of training a sequential SVM on this dataset of almost 1.5 million images with our 988 dimensions would be prohibitive and even with a great deal of work in data reduction techniques, it would still likely take several months to train the model. By dividing the task into several hierarchical levels based on the knowledge of articulatory phonetic classifications in English as described in [13], we are able to turn the problem into a binary, ternary, or quaternary classification instead of the original 40 classes. Ladefoged suggests a hierarchical structure necessary for English features in the last chapter of [13]. Although his tree is not a binary decision tree, but rather features 129 overlapping leaves with multiple paths, the idea of using articulatory features to classify phonemes was inspired by that approach. This served as the primary source for our tree which was derived as possible questions to classify each phoneme so that they contain the features necessary for classification. This makes our classification space much more simple when creating the support vectors. It should be noted that the first layer classifying obstruents, silence, and sonorants is built using four individually trained machines on equal chunks of data where the prediction for this layer is made by a simple voting system where the mode is taken as the final prediction. This was done to further reduce the training time since at this stage we have the largest amount of data in the tree and the simplest hypothesis space, given that the features are most distinct between silence, obstruents, and sonorants than in lower nodes of the tree. The hierarchical structure is presented in Fig. 2. Note that the final nodes in the HTSVM produce the respective classes within the final category for a total of 40 classes. This number comes from the suggested 39 monophones from [41] plus the silence class. It should be noted that silence frames are not used in the classification results to be consistent with the literature.

In the second issue, we deal with an unbalanced dataset where even minimal pairs can have a large difference in frequency as in the example of /k/ and /g/ which are both velar stops. In the training set, the phoneme /k/ appears in 60,433 frames, whereas /g/ is found in only 17,727. In order to build a robust system, it is important to learn this phonemic distinction and minimize the influence of probability in the training set. We were able to deal with this by using the synthetic minority over-sampling technique (SMOTE) [51] data augmentation technique. The synthetic creation of minority samples allows us to treat the classification task with more confidence. This was also made possible by the hierarchical approach because each node has a much smaller number of samples than the original dataset.

PER smoothing

In order to obtain PER, the classified frames must be converted to phonemes. This was done by taking the mode of all of the SVM classifications for each GMM-HMM-generated boundary, initially produced by the HMM labeler as seen in Fig. 1, thus collapsing repeated frame classifications into single phonemes. In other words, the values of the initial labels are not actually used, just the boundaries so that a decision can be made for each one. Since each boundary contains multiple frames classified by the SVM, a single value is calculated by taking the mode of all frames within the respective boundary. In the case where classifications across boundaries produced the same results, that boundary ceased to exist. The acoustic model used in the HMM labeler is the model trained by [34] on the SCOTUS corpus. Smoothing is an important step because it avoids likely misclassifications from the frame level, for example: a single frame classified as a phoneme A between two frames classified as a phoneme B seems extremely unlikely. Also in the case of a long co-articulation or heavy aspiration the beginning or ending of a phoneme could be confused with an entirely different sound but as time goes on this should become more clear.

Experiments

As a baseline comparison to the proposed method, we used one of the most popular ASR toolkits for a database the size of TIMIT, the HTK toolkit [33]. A triphone HTK model was trained on the same TIMIT training set used in our method, and recognition was performed on the test set with a zero-gram language model, with only the individual monophones as the pronunciation model. This was done in order to obtain only the posterior values from the acoustic model predictions without the influence of

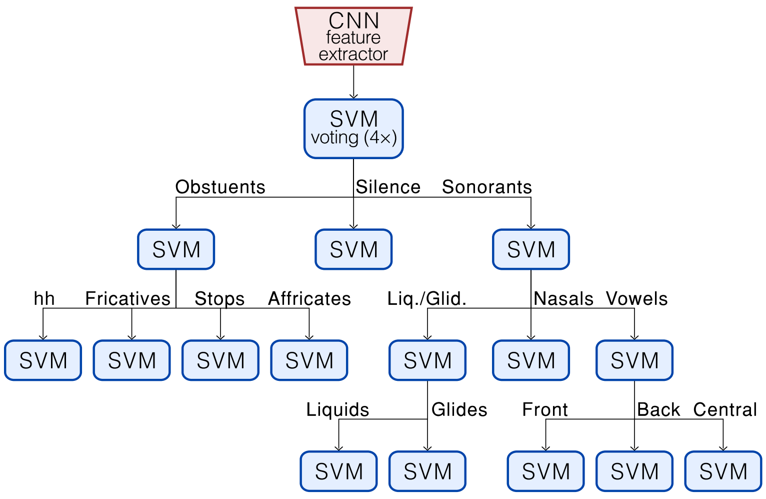

Fig. 2 *HTSVM architecture defined for the experiments*

a language or pronunciation model for fair comparison, since we are only evaluating the accuracy of the acoustic model. We used 31 Gaussian components because this number is one which we have found useful in the past. It is higher than what is recommended by the voxforge tutorial at [52], which uses 15 and is similar to the models used by Keith Vertanen at [53], where he uses a maximum of 32 for the Wall Street Journal and TIMIT datasets together. The model was trained using MFCC 0DANZ acoustic features, where 0 use zeroth cepstral coefficient, D is for the delta coefficients, A for acceleration coefficients, computed as delta of delta coefficients, N is for absolute energy suppression and Z is zero mean normalization. The predictions were then segmented in the same fashion as the proposed method with 25 ms sliding windows and a step of 10 ms, in order to make a frame by frame comparison.

Along with the HTK model, we also trained a simple MLP with one hidden layer, 100 neurons, a ReLu activation function and an Adam solver. The choice for 100 neurons was made because it does not saturate the network. We experimented with larger configurations like 1000 neurons but no more than an overall accuracy gain of 0.6% was obtained, and the loss did not improve at 1.992. After running multiple experiments, it seems that this was due to chance since some experiments were slightly lower and some higher. Deeper architectures with two or three layers presented worse results of about a 5 to 8% drop in accuracy over several experiments. We believe that this is due to the complexity of the network and is an expected result due to the small dataset. The learning rate and all other hyperparameters used the default values from the MLPClassifier from [54] which was heavily based on the work in [55]. The network is also very similar to the one used in [39]. Again, for consistency, we chose to use a strong baseline network without hyperparameter tuning. This was done to show the gain provided by the HTSVM structure over another widely used classifier for the classification of CNN features.

Results

For each model, we calculated the FER and F1 scores since accuracy can be misleading at times when dealing with a number of less frequent phonemes. This was done first on a frame by frame analysis. Then, we smoothed the data by removing frame repetitions in order to collapse them into sequential phonemes. For the latter, we were also able to calculate the PER, the industry standard for measuring the accuracy of acoustic models; it is calculated by the Levenshtein distance [35] where the number of insertions, deletions, and substitutions were added and divided by the total number of phonetic units in the string. It should be noted that for fair comparison, we use the same 39 phonemes, which has become the standard for evaluation [56], omitting silence. T test results yielded less than 0.01 between classifiers, evidencing that the results were significantly different.

Table 2 presents the F1 scores and FER of the GMM-HMM, CNN-MLP, and CNN-HTSVM models for frame classifications as well as the PER in the case of phoneme classification. The FER was calculated as an accuracy score, PER was calculated by the conventional Levenshtein edit distance [35] and the F1 score was calculated as $2 \times \frac{precision \times recall}{precision + recall}$.

Independent of the model's accuracy, it is also important to understand what sort of errors the model is actually committing. Table 3 lists the 15 most frequent errors committed by each system, including the true values, predicted values, and the confusion percentage.

The confusion percentage for each true class is defined as the percentage of all occurrences which were misclassified as the predicted class. In the table, it becomes clear that our method is only confused in the most turbulent shattering cases were a great overlap in discriminative features occurs.

Convergence analysis

As previously mentioned, an important part of our work deals with learning guarantees. In this section, we discuss the learning convergence of the CNN network and the SVM model employed in this paper. SLT provides theoretical support for such convergence proofs in terms of how supervised learning algorithms generalize examples. Equation 1 defines the main principle of SLT which is the empirical risk minimization [48] to bound the divergence ϵ between the empirical risk R_{emp}, i.e., the error measured in a sample, and the expected risk $R(f)$, i.e., the expected error while assessing the joint probability distribution of examples and their respective classes, as the sample size n tends towards infinity. Still describing the equation, the right-most term is known as the Chernoff bound, f is a given classifier, and $\mathcal{F}$ is the space of admissible functions provided by some supervised algorithm, a.k.a. the algorithm bias [17, 48, 57].

$$P\left(\sup_{f \in \mathcal{F}} \left|R(f) - R_{emp}(f)\right| \geq \epsilon\right) \leq 2e^{-n\epsilon^2/4} \tag{1}$$

Vapnik [17] proved a bound for supervised learning algorithms considering the shattering coefficient

Table 2 F1 Scores in frames, frame error rates and phone error rates for each model

Classifier	F1 Score	FER%	PER%
GMM-HMM	0.166	76.36	75.17
CNN-MLP	0.225	56.97	52.90
CNN-HTSVM	*0.491*	*37.04*	*35.41*

Table 3 Most frequent FER confusion percentages in GMM-HMM and CNN-HTSVM models where the true phoneme was confused as being the predicted phoneme

GMM-HMM			CNN-HTSVM		
True	Pred	Conf (%)	True	Pred	Conf (%)
s	z	33.14	s	z	15.16
ih	uw	16.00	ay	ae	39.64
t	ch	17.58	ao	aa	26.58
er	r	32.46	r	er	18.84
ao	l	28.00	sh	s	26.01
iy	y	14.23	aa	ae	16.07
s	sh	10.09	ah	ae	14.79
ae	t	14.32	t	s	7.61
ih	z	10.07	iy	ih	6.48
w	ao	45.52	er	r	7.64
iy	uw	12.80	er	r	10.64
k	eh	11.55	z	s	18.01
ih	t	7.70	ay	aa	15.78
ah	l	16.67	t	s	10.61
d	t	14.46	iy	ih	9.48

$\mathcal{N}(\mathcal{F}, 2n)$, as defined in Eq. 2. Such a coefficient is a measure function to compute the complexity of the algorithm bias, i.e., the cardinality of functions contained in the space $\mathcal{F}$ that produce different classification outputs, provided a sample size n. Throughout our formulation, we employ the generalization bound defined in Eq. 4, a further result obtained from Eq. 2, to ensure that the expected risk is bounded by the empirical risk plus an additional term associated to the shattering coefficient and some probability δ (Eq. 3).

$$P\left(\sup_{f\in\mathcal{F}} \left|R\left(f\right) - R_{emp}\left(f\right)\right| \geq \epsilon\right) \leq 2\mathcal{N}(\mathcal{F}, 2n)e^{-n\epsilon^2/4} \tag{2}$$

$$P\left(\sup_{f\in\mathcal{F}} \left|R\left(f\right) - R_{emp}\left(f\right)\right| \geq \epsilon\right) \leq \delta \tag{3}$$
$$\delta = 2\mathcal{N}(\mathcal{F}, 2n)e^{-n\epsilon^2/4}$$

$$R(f) \leq R_{emp}(f) + \sqrt{4/n\left(\log(2\mathcal{N}(\mathcal{F}, n)) - \log(\delta)\right)} \tag{4}$$

In the case of the SVM, the same bound is formulated as shown in Eq. 5, in which c is some constant, R corresponds to the dataset radius, and ρ represents the maximal margin.

$$R(f) \leq R_{emp}(f) + \sqrt{c/n\left(R^2/\rho^2 - \log(1/\delta)\right)} \tag{5}$$

In this context, we assessed our CNN and SVM to understand the sample size they require to ensure learning in the context of speech recognition, allowing to estimate their expected risk value over unseen examples. The CNN architecture used is composed of one convolutional layer with 38 units whose mask size is 29 × 1, as estimated using the adapted FNN [45]. The mask size and the number of units are important parameters to estimate the shattering coefficient for a single CNN layer used in our experiments. Considering the formulation proposed in [58], Eq. 6 defines the shattering coefficient for a single unit in the CNN layer, in which h is the space dimensionality and n corresponds to the sample size. Thus, Eq. 7 corresponds to the shattering coefficient for all 38 units in this layer, in which p is the number of hyperplanes.

$$f(n) = 2\sum_{i=0}^{h}\binom{n-1}{i} = 2\sum_{i=0}^{29\times 1}\binom{n-1}{i} \tag{6}$$

$$CNN(n) = 2\sum_{i=0}^{h}\binom{n-1}{i}^{p} = 2\sum_{i=0}^{29\times 1}\binom{n-1}{i}^{38} \tag{7}$$

Now, we proceed by computing the generalization bound for the CNN (Eq. 4), as shown in Eq. 8. Considering $\delta = 0.05$, that represents a probability of 0.95 (i.e., 95%) to ensure that the empirical risk $R_{\text{emp}}(f)$ is a good estimator for the expected risk $R(f)$, meaning the error results measured for our classifier indeed work on unseen examples. Observe that our CNN requires at least 216,640 examples to converge, while we had in practice 1,447,869 examples in training set.

In addition, we can employ another result from [17] to prove that our CNN converges. Equation 9 considers the most relevant term to prove the learning convergence, analyzing Eq. 8 [17, 48]. Notice that as $\frac{\log CNN(n)}{n}$ approaches zero, term $\sqrt{4/n(\log(2CNN(n)) - \log(0.05))}$ from Eq. 8 goes to zero, remaining the empirical risk as an assessment measure of the learning performance.

$$R(f) \leq R_{\text{emp}}(f) + \sqrt{4/n(\log(2CNN(n)) - \log(0.05))} \tag{8}$$

$$\lim_{n\to\infty} \frac{\log\{CNN(n)\}}{n} \approx 0, \tag{9}$$

Next, the SVM is also analyzed considering Eq. 5. In this case, we have an accuracy of 0.61 leading to $\nu(f) = 1 - \text{accuracy} = 1 - 0.61 = 0.39, R = 3,332,567$ (the radius we estimated for the whole dataset) and $\rho = 173,869,050$

as the maximal margin found. Consequently, the generalization bound for our SVM is defined in Eq. 10. Also, considering $\delta = 0.05$ as before, and $c = 4$ as taken in the default formulation (Eq. 4), notice our SVM requires at least 1476 examples to converge, while we had in practice 1,447,869 examples in the training set.

$$R(f) \leq 0.39 + \sqrt{4/n(3,332,567^2/173,869,050^2 - \log(1/\delta))} \quad (10)$$

Based on [58], we formulated the shattering coefficient for our CNN architecture that was composed of a single convolutional layer. The shattering coefficient of the SVM was also calculated, according to SLT [17, 48]. Those formulations ensure our framework presents learning guarantees in the context of speech recognition.

Discussion

Most errors committed by the HTSVM were made between similar phonemes as found in [31], many of which unlikely would be perceived by human listeners. For example, the confusion between /s/ and /z/ are likely produced by the database itself where speakers may mix the two or use them interchangeably, where it is semantically unimportant [59], or even produce a partially vocalized fricative. This shows that the algorithm is quite promising, since it avoided gross errors which are common in many modern acoustic modeling algorithms like GMM-HMM. The issue between similar phonemes seems to be one which could be corrected by a robust language and/or pronunciation model. Also, since such a small window of 25 ms was used, some of the vowels, especially diphthongs like /ay/ were damaged. Studies like [29] use nine frames for classification and [60] suggest at least 100 ms for context which prompted [38] to use 11 frames (110ms) as context for classification. While our system is more accurate in frame classification than the system in [38], our PER is worse. This is probably also due to the small window size where unnecessary insertions are made in the phoneme strings.

The combination of features extracted from single frames by a simple and shallow CNN classified by the HTSVM produced similar accuracy results to that study on a more difficult task as we were classifying full utterances which include pauses, co-articulation, and a greater variation in pronunciation and not only single words and likely more careful speech. It is also important to note that some errors, like the case of the vowels which are close in the vowel space, could have been caused by the TIMIT transcriptions themselves, where, in some dialects, these sounds could be very similar or even some sound in between. It was also interesting that the results are not far from the state-of-the-art forced alignment results where a pronunciation dictionary is employed for disambiguation. This shows that not only is the recognition robust but also that the CNN-HTSVM seems promising for automatic segmentation as well, either as a stand-alone algorithm or a post-processor. We believe that the take home point of this paper is that a quite robust acoustic model can be built even with a simple architecture and dataset when carefully constructed. Robust means that the results are similar to related recent studies with a 37.04% FER, and the convergence analysis provides sufficient evidence that the model will infer unseen data well, guaranteeing that the results were not obtained by chance.

Conclusions and future work

We have shown that even with the large dimensionality of the CNN features, a shallow CNN-HTSVM architecture can be useful in scenarios where a large abundance of data is not available. Our method shows similar or better results on the same corpus than the most recent HTSVM methods and outperforms other works using few frames for classification as well as the traditional GMM-HMM classifier and an MLP classifier using the same CNN features. We also make a contribution towards the comparison of acoustic models by presenting a breakdown of frame and phoneme accuracy with F1 scores which were not available in many of the previous studies, as well as an analysis of the convergence of the model, providing more information about the true robustness, facilitating the assessment of state-of-the-art acoustic models.

As future work, our CNN-HTSVM pipeline could take advantage of some fine-tuning. Since most errors were made between very similar phonemes and diphthongs, it would be interesting to investigate this issue further. As a first step, one could implement something along the lines of a backtracking system in the tree where after a preliminary decision is made in the node (especially in the final node), a second node could check for likely errors according to predefined rules. As a second solution to this problem, one could add more frames to the classification stage as employed in other studies. It also would be useful to investigate other features which could be added to better disambiguate some of the more frequent errors, for example, acoustic features to indicate voicing for sonorants and some fricatives or voice onset time between stops and fricatives. We also believe that the CNN will be even more valuable when applied to a noisy database or a database with more classes and we are currently exploring this scenario. We plan to revise the CNN architecture by adding more convolutional layers and possibly greater max pooling to reduce the dimensionality. This would allow for more efficient experiments. The sample size could also be reduced by calculating the distances from support vectors, using only the most important examples.

Therefore, we believe that the error rates should decrease significantly even on small datasets like TIMIT and similar datasets for other languages. A future work should explore the use of different language models which should smooth the errors produced by the acoustic model.

Endnote

[1] The time domain is to be handled by the HMM labeler.

Abbreviations

ASR: Automatic speech recognition; CNN: Convolutional neural network; FER: Frame error rate; FFT: Fast fourier transform; FA: Forced alignment; FNN: False nearest neighbors; GMM-HMM: Gaussian mixture models - hidden Markov models; HTSVM: Hierarchical tree support vector machine; MCT: Multi-conditional training; MFCC: Mel-frequency cepstrum coefficients; MLP: Multi-layer perceptron; PER: Phone error rate; RBF: Radial basis function; SLT: Statistical learning theory; SMOTE: Synthetic minority over-sampling technique; VC: Vapnik-Chervonenkis; WER: Word error rate

Acknowledgements

We would like to thank our colleagues in the NILC laboratory for their input and support.

Funding

We would like to thank CNPq for the grants 441583/2014-8, 303051/2014-0 and 302077/2017-0 as well as FAPESP for the grants 2013/07375-0 and 2017/16548-6, all to RM. Also, we thank Capes for the grant PROEX-7901561/D to MF.

Availability of data and materials

All material used in this paper is open-source and can be downloaded for uses according to the public licenses provided their origins. The TIMIT corpus can be found at the Linguistic Data Consortium: https://catalog.ldc.upenn.edu/ldc93s1. For the machine learning algorithms, we used the CNN from Keras: https://keras.io/layers/convolutional/ and the SVM from scikit learn: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC. The SMOTE algorithm can be found in the imbalanced learn repository: https://contrib.scikit-learn.org/imbalanced-learn/stable/generated/imblearn.over_sampling.SMOTE.html. All non-default parameters for these algorithms are given here in this article. For FA and HMM training HTK can be obtained at https://htk.eng.cam.ac.uk/. The htklabel plugin was used for manual revision and can be found in the CPrAN repository at: https://cpran.net/plugins/htklabel/ or the bleeding edge version in my github repository: https://github.com/CShulby/plugin_htklabel.

Authors' contributions

CS has developed the acoustic model and pipeline presented in this paper as a core part of his PhD thesis. MF developed the algorithm for intelligent CNN mask selection based on her adaptations of the false-nearest-neighbors algorithm which is part of her PhD thesis. RM provided the convergence analysis, which is part of his current research. SM contributed in the literature review and organization of the machine learning algorithms presented as the first author's advisor. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Received: 18 March 2018 Accepted: 21 October 2018
Published online: 04 January 2019

References

1. Witt SM (2012) Automatic error detection in pronunciation training: where we are and where we need to go. Proc IS ADEPT 6:1–8
2. Li K, Qian X, Meng H (2017) Mispronunciation detection and diagnosis in l2 english speech using multidistribution deep neural networks. IEEE/ACM Trans Audio Speech Lang Process 25(1):193–207
3. Hinton G, Deng L, Yu D, Dahl GE, Mohamed A-R, Jaitly N, Senior A, Vanhoucke V, Nguyen P, Sainath TN, et al (2012) Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Proc Mag 29(6):82–97
4. Chan A (2005) 10 Common Pitfalls of using SphinxTrain. http://www.cs.cmu.edu/~archan/10CommonPitfallsST.html. Accessed: 12 Oct 2016
5. Cieri C, Miller D, Walker K (2004) The Fisher corpus: a resource for the next generations of speech-to-text. In: LREC, vol.4. pp 69–71
6. Panayotov V, Chen G, Povey D, Khudanpur S (2015) Librispeech: an ASR corpus based on public domain audio books. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE. pp 5206–5210
7. Chen X, Eversole A, Li G, Yu D, Seide F (2012) Pipelined back-propagation for context-dependent deep neural networks. In: Interspeech, Portland. pp 26–29
8. May T (2017) Robust speech dereverberation with a neural network-based post-filter that exploits multi-conditional training of binaural cues. In: IEEE/ACM Trans Audio, Speech, and Lang Process
9. Kim TY, Han CW, Kim S, Ahn D, Jeong S, Lee JW (2016) Korean LVCSR system development for personal assistant service. In: Consumer Electronics (ICCE), 2016 IEEE International Conference On. IEEE, Las Vegas. pp 93–96
10. Zhang C, Bengio S, Hardt M, Recht B, Vinyals O (2016) Understanding deep learning requires rethinking generalization. In: CoRR. https://doi.org/abs/1611.03530
11. LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324
12. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems. pp 1097–1105
13. Ladefoged P, Disner SF (2012) Vowels and consonants. 3rd. Wiley-Blackwell, Malden, MA
14. Hubel DH, Wiesel TN (1962) Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. J Physiol 160(1): 106–154
15. Abdel-Hamid O, Mohamed A-R, Jiang H, Deng L, Penn G, Yu D (2014) Convolutional neural networks for speech recognition. IEEE/ACM Trans Audio Speech Lang Process 22(10):1533–1545
16. LeCun Y, Bengio Y (1995) Convolutional networks for images, speech, and time series. Handb Brain Theory Neural Netw 3361(10):1995
17. Vapnik V (2013) The Nature of Statistical Learning Theory. In: Paperback, 2nd. Springer, New York
18. Shulby CD, Ferreira MD, de Mello RF, Aluísio SM (2017) Acoustic modeling using a shallow CNN-HTSVM architecture. In: 2017 Brazilian Conference on Intelligent Systems (BRACIS). IEEE, Uberlândia. pp 85–90
19. Waibel A, Hanazawa T, Hinton G, Shikano K, Lang KJ (1989) Phoneme recognition using time-delay neural networks. IEEE Trans Acoust Speech Signal Proc 37(3):328–339
20. Lee H, Pham P, Largman Y, Ng AY (2009) Unsupervised feature learning for audio classification using convolutional deep belief networks. In: Advances in Neural Information Processing Systems. pp 1096–1104
21. Hau D, Chen K (2011) Exploring hierarchical speech representations with a deep convolutional neural network. In: UKCI 2011 Accepted Papers. p 37
22. Abdel-Hamid O, Mohamed A-R, Jiang H, Penn G (2012) Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition. In: 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, Kyoto. pp 4277–4280
23. Sainath TN, Mohamed A-R, Kingsbury B, Ramabhadran B (2013) Deep convolutional neural networks for LVCSR. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, Vancouver. pp 8614–8618
24. Mohamed A-R, Dahl GE, Hinton G (2012) Acoustic modeling using deep belief networks. IEEE Trans Audio Speech Lang Process 20(1):14–22
25. Graves A, Jaitly N (2014) Towards end-to-end speech recognition with recurrent neural networks. In: ICML. vol. 14. pp 1764–1772
26. Maas AL, Hannun AY, Jurafsky D, Ng AY (2014) First-pass large vocabulary continuous speech recognition using bi-directional recurrent DNNS. In: CoRR. https://doi.org/abs/1408.2873

27. Tóth L (2015) Phone recognition with hierarchical convolutional deep maxout networks. EURASIP J Audio Speech Music Proc 2015(1):25
28. Dekel O, Keshet J, Singer Y (2004) An online algorithm for hierarchical phoneme classification. In: International Workshop on Machine Learning for Multimodal Interaction. Springer, Martigny. pp 146–158
29. Karpagavalli S, Chandra E (2015) A hierarchical approach in tamil phoneme classification using support vector machine. Indian J Sci Technol 8(35):57–63
30. Driaunys K, Rudžionis V, Žvinys P (2015) Implementation of hierarchical phoneme classification approach on LTDIGITS corpora. Inf Technol Control 38(4):303–310
31. Amami R, Ellouze N (2015) Study of phonemes confusions in hierarchical automatic phoneme recognition system. In: CoRR. https://doi.org/abs/1508.01718
32. Schiel F, Draxler C, Baumann A, Ellbogen T, Steffen A (2012) The production of speech corpora. epub uni-muenchen
33. Young S, Evermann G, Gales M, Hain T, Kershaw D, Liu X, Moore G, Odell J, Ollason D, Povey D, et al (2002) The HTK book. Cambridge Univ Eng Dept 3:175
34. Yuan J, Liberman M (2008) Speaker identification on the SCOTUS corpus. J Acoust Soc Am 123(5):3878
35. Levenshtein VI (1966) Binary codes capable of correcting deletions, insertions, and reversals
36. Hifny Y, Renals S (2009) Speech recognition using augmented conditional random fields. IEEE Trans Audio Speech Lang Process 17(2):354–365
37. Graves A, Jaitly N, Mohamed A-R (2013) Hybrid speech recognition with deep bidirectional lstm. In: Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop On. IEEE, Olomouc. pp 273–278
38. Lombart J, Miguel A, Lleida E (2013) Articulatory feature extraction from voice and their impact on hybrid acoustic models. In: Advances in Speech and Language Technologies for Iberian Languages. Springer, Las Palmas de Gran Canaria. pp 138–147
39. Lopes C, Perdigão F, et al (2009) Phonetic recognition improvements through input feature set combination and acoustic context window widening. In: 7th Conference on Telecommunications, Conftele. Citeseer, Porto. pp 449–452
40. Garofolo J, Lamel L, Fisher W, Fiscus J, Pallett D, Dahlgren N (1990) The DARPA TIMIT acoustic-phonetic continuous speech corpus, NTIS speech disc. NTIS order number PB91-100354
41. Lee K-F, Hon H-W (1989) Speaker-independent phone recognition using hidden Markov models. IEEE Trans Acoust Speech Signal Process 37(11):1641–1648
42. Bagwell C (2018) Sox(1) - Linux man page. https://linux.die.net/man/1/sox. Accessed: 01 Mar 2018
43. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444
44. Ian Goodfellow AC, Yoshua Bengio (2016) Deep learning. The MIT Press, Cambridge. http://goodfeli.github.io/dlbook/. Accessed 18 Mar 2018
45. Ferreira MD, Corrêa DC, Nonato LG, de Mello RF (2018) Designing architectures of convolutional neural networks to solve practical problems. In: Expert Systems with Applications 94(Supplement C). pp 205–217. https://doi.org/10.1016/j.eswa.2017.10.052
46. Chollet F, et al (2015) Keras. https://keras.io. Accessed 18 Mar 2018
47. Bromberg I, Qian Q, Hou J, Li J, Ma C, Matthews B, Moreno-Daniel A, Morris J, Siniscalchi M, Tsao Y, Wang Y (2017) Detection-based ASR in the automatic speech attribute transcription project. In: Proceedings of The Interspeech 2017, 18th Annual Conference of the International Speech Communication Association. ISCA, Stockholm, Sweden. pp 1829–1832. https://doi.org/10.21437/Interspeech.2017
48. von Luxburg U, Schölkopf B (2011) Statistical learning theory: models, concepts, and results, vol. 10. Elsevier, North Holland, Amsterdam, Netherlands. Max-Planck-Gesellschaft
49. Chang Y-W, Hsieh C-J, Chang K-W, Ringgaard M, Lin C-J (2010) Training and testing low-degree polynomial data mappings via linear svm. J Mach Learn Res 11(Apr):1471–1490
50. Goldberg Y, Elhadad M (2008) splitSVM: fast, space-efficient, non-heuristic, polynomial kernel computation for NLP applications. In: Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers. Association for Computational Linguistics, Columbus. pp 237–240
51. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) Smote: synthetic minority over-sampling technique. J Artif Intell Res 16:321–357
52. MacLean K (2018) Tutorial: create acoustic model - manually. http://www.voxforge.org/home/dev/acousticmodels/linux/create/htkjulius/tutorial/triphones/step-10. Accessed: 1 Mar 2018
53. Vertanen K (2018) HTK acoustic models. https://www.keithv.com/software/htk/us/. Accessed: 1 Mar 2018
54. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in Python. J Mach Learn Res 12:2825–2830
55. Hinton GE (2011) Connectionist learning procedures. Artif Intell 40(1-3):185–234. https://doi.org/10.1016/0004-3702(89)90049-0
56. Lopes C, Perdigão F (2012) Phone recognition on the TIMIT database. In: Speech Technologies. https://doi.org/10.5772/17600
57. de Mello RF, Ferreira MD, Ponti MA (2017) Providing theoretical learning guarantees to deep learning networks. In: CoRR. https://doi.org/abs/1711.10292
58. de Mello FR, Antonelli Ponti M, Grossi Ferreira CH (2018) Computing the shattering coefficient of supervised learning algorithms. ArXiv e-prints. http://arxiv.org/abs/1805.02627
59. Hoffmann S, TIK E (2009) Automatic phone segmentation. Corpora 3:2–1
60. Yang HH, Van Vuuren S, Sharma S, Hermansky H (2000) Relevance of time–frequency features for phonetic and speaker-channel classification. Speech Comm 31(1):35–50