The use of Convolutional Long Short-Term Memory Deep Neural Network for two-stage keywords spotting system

DRIDI Hinda[1] & OUNI Kais[2]

Research Unit Signals and Mechatronic Systems, UR13ES49
National Engineering School of Carthage, ENICarthage
University of Carthage, Tunis, Tunisia

Abstract: Recently, the deep learning architectures have been widely applied in a large variety of speech recognition tasks. Convolutional Neural Networks (CNNs) and Long Short Term Memory networks (LSTMs) have both achieved very promising results compared to Deep Neural Networks (DNNs). Due to the complementary modeling capabilities of these different deep networks, they may be combined in a unified architecture called Convolutional Long Short-Term Memory Deep Neural Network (CLDNN). In this paper we use the CLDNN architecture to introduce a systematic approach of keywords spotting (KWS) in continuous speech. This system consists of two stages, in first one the inputted utterances are decoded into phonetic flow using hybrid model combining the CLDNN architecture with the Hidden Markov Models (HMMs). This CLDNN-HMM model is built with the open source speech recognition toolkit Kaldi. And in second stage the keywords will be detected from this phonetic flow using the Classification and Regression Tree (CART). The work and experiments are conducted on TIMIT data set.

Keywords: Convolutional Neural Network (CNN); Long Short Term Memory (LSTM); Deep Neural Network (DNN); CLDNN; Keywords Spotting (KWS); two-stage; Hidden Markov Model (HMM); Classification and Regression Tree (CART); Kaldi

1. Introduction

In last few years, due to significant technological advances large volumes of spoken data have been saved, shared and available online through Internet or different databases every day. Controlling these large volumes of data is a difficult activity for human being. For that, introducing new automatic methods for accessing to these large volumes of data and extracting all the useful information contained therein is become essential. Consequently, the task of keywords spotting has been introduced and has become an interesting branch of speech processing. This task, which allows detecting and searching some pre-defined words (keywords) in utterances of continuous speech, has been widely used in many applications like, spoken document retrieval, spoken term detection, spoken data mining, telephone routing, etc…

The task of keywords spotting has been treated with different approaches over the years. Some of the earliest approaches have used the Dynamic Time Warping (DTW) algorithm to search the keywords by computing an alignment distance between a template representing the target keyword and all segments of the speech signal to look for a correspondence. The KWS approaches based on DTW have been adopted by many researchers in the 80s, but along with the evolution in speech recognition field they started showing their limitations. Such approaches have shown acceptable results in the case of isolated KWS tasks but have shown considerable drops in performances for the case of continuous ones. This drop in performances is explicated by the fact that templates couldn't represent the inherit variation of human speech. [1], [2], [5]

Consequently, different approaches have been introduced later and the most promising one was based on Hidden Markov Model (HMM). In this technique, the keywords were defined by their phonetic models and the non-keywords were defined by filler or garbage models. Introducing the garbage models was

(doi:1444-8939.2018/5-4/MRR.46)

interesting for modeling the extraneous speech and the background noises. Also, the garbage models have shown more improvements when accepting or rejecting the possible occurences of keywords over the template-based approaches. To further improve the performances of the HMM-based approach many works have proposed later the use of neural networks. Nevertheless, the main drawback of garbage models is that they are highly unconstrained and can absorb some speech segments, which correspond to relevant keywords. Also, the HMM-based keywords spotting technique presents another major drawback, which consists of processing again the recognition stage for each modification in the application vocabulary that is a very time-consuming task. [3], [4], [6]-[8]

Recently, to overcome this shortcoming the two-stage keywords spotting approaches have been investigated and have been very popular. Such approaches are based on two steps that are, indexing and search. The first stage of indexing generates for each inputted utterance a phonetic transcription using automatic speech recognition (ASR) system. This phonetic transcription is called an index. In the second stage namely, keywords search, this index will be used to quickly search the occurences of target keywords. The interesting advantage of this two-stage approach is doing a main portion of the keywords spotting task beforehand, without any prior knowledge of the keywords. The two-stage keywords spotting approach is much faster and more performing than other approaches. [7]- [8]

In this paper, we will adopt the two-stage approach to build our KWS system. In first stage, we use the Convolutional Long Short-Term Memory Deep Neural Network (CLDNN) architecture in combination with the Hidden Markov Models to decode the continuous speech utterances into phones sequences. Then, the predefined keywords will be detected from this phonetic flow using the Classification and Regression Trees.

This paper will be organized as follows: In Section 2, we describe the CLDNN deep model. In Section 3, we present an overview of our proposed KWS system. In Section 4, the experimental setup and results of both phone recognition using the "CLDNN-HMM" model, and the proposed two-stage KWS system will be presented. Finally, in Section 5 we draw some conclusions and we outline our future works.

2. An overview of the deep CLDNN model

Recently, the deep learning architectures have been widely used in a large variety of automatic speech recognition applications, like phonetic decoding. These different deep networks have been shown very promising and outperforming the traditional "GMM-HMM" model, which combines Gaussian Mixture Model (GMM) and Hidden Markov Model (HMM). In the following we will present some of the most promising deep structures and then we will present the CLDNN model.

Some of the earliest deep architectures, which are applied for phonetic decoding of continuous speech and have achieved efficient results, are the Deep Neural Networks (DNNs). A DNN is a classic Multi-Layer Perceptron (MLP) with many fully connected layers of hidden units. All the hidden units of each layer are connected to those in next layer using unidirectional connections [34]. The Deep Neural Network has been used in combination with Hidden Markov Model in a single hybrid model known as "DNN-HMM" and it has been trained in a supervised way and unsupervised one using the pre-training approach investigated by Hinton et al. in [14].

The earliest work using pre-trained DNN for phonetic decoding was proposed by Mohamed et al [26]. The deep model used in this work was able to achieve a phone error rate of 22.4% on TIMIT test set. This phone error rate was very interesting and outperforming the rates achieved with the classic "GMM-HMM" models.

Later, Convolutional Neural Network (CNN) has been used for phone recognition and has shown more performing results compared to Deep Neural Network. A standard convolutional neural network has two major parts: a convolutional stage composed by a pair of convolution and pooling layers, and a certain number of fully connected layers. In the convolution layer the neurons are arranged into feature maps, where those appertaining to one feature map share the same weights (called filters or kernels). The pooling layer is also

(doi:1444-8939.2018/5-4/MRR.46)

organized into a same number of feature maps as the convolution layer, but with smaller maps.

Abdel-Hamid et al [15] have investigated the use of a CNN with new limited-weight-sharing scheme and with convolution over frequency for phone recognition. The performances of their hybrid "CNN-HMM" model were very promising. A phone error rate of 20.36% was obtained for TIMIT test set that is outperforming the results obtained with the "DNN-HMM" model studied in previous works. Also, Loth [17] proposed "CNN-HMM" architecture with convolution over both time and frequency. In this work, the lowest phone error rate achieved on the TIMIT test set was 16.7%.

Despite showing very promising results, the convolutional neural network (CNN) can just model a limited temporal dependency. As consequent, it's not efficient to model speech signals with long-range dynamics. To overcome this weakness, an alternative structure of RNN, called Long Short-Term Memory (LSTM) has been proposed. The reccurent hidden layer in a classic LSTM is consisted of recurrently connected units called "memory blocks". Each memory block has a certain number of self-connected memory cells to store the contextual information and three multiplicative gates namely, input, output and forget gate to manage the flow of information. Another structure called Long Short-Term Memory Projected (LSTMP) has been proposed later. It has an LSTM layer followed by a separate linear projection layer. Adding the projection layer allows not only reducing the number of parameters but also providing an improved performance. To bring more amelioration to these unidirectional models their bidirectional alternatives, namely Bidirectional LSTM (BLSTM) and Bidirectional LSTMP (BLSTMP) have been introduced. A Bidirectional LSTM contains two layers; forward and backward to process the input sequence respectively in the forward and backward direction. The final output is obtained by concatenating the outputs of the two layers. [21]- [23]

Graves et al [23] proposed the first use of Long Short Term Memory (LSTM) for phone recognition. They have demonstrated in their work that a bidirectional LSTM (BLSTM) is more performing than a unidirectional one. A phone error rate of 17.7% was obtained for TIMIT test set using a deep BLSTM.

The different deep architectures (DNN, CNN and LSTM), described previously, have successfully improved the speech recognition performances; however they all have distinct strengths and weaknesses. The individual shortcomings of these various deep networks have motivated the idea of assembling them in a single CLDNN architecture to achieve further improvements. The CLDNN model may efficiently reduce the spectral variations by a frequency convolution; may reduce the long-term temporal dynamics using LSTM layers and may further reduce the final class discrimination with few DNN layers.

The baseline CLDNN architecture used in this paper is as investigated in [11]. As shown in Fig.1, we pass the input features within their temporal context into a Convolutional Neural Network to reduce the spectral variations. The CNN used in this paper is only with convolution along frequency, and it has two convolution layers. The first convolution layer is followed by a pooling layer, while no pooling was used for the second layer. Then a certain number of fully connected layers are added, each of them containing 1024 hidden units, with sigmoid activation function. Next, the output of this CNN is passed into a stack of LSTM layers, which are suitable for long-term temporal modeling. Finally, we pass the final LSTM layer output into a DNN with a few number of fully-connected feed-forward layers. These top deep layers are appropriate to produce a higher-order feature representation for an easy separation into the different classes that we want to discriminate.

(doi:1444-8939.2018/5-4/MRR.46)

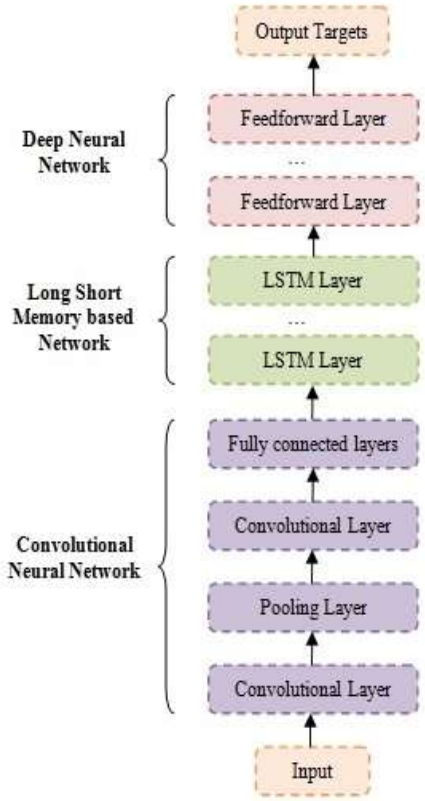

Fig. 1. An illustration of the CLDNN architecture.

The output of this CLDNN model is a probability distribution over the possible labels of the central frame. To get the phones sequence, these probabilities will be divided by the HMM states generated by the top DNN layer, and then will be passed to a Viterbi decoder. [9], [11]

3. An overview of the proposed keywords spotting system

Due to promising performances of the two-stage approach we will use it to build our keywords spotting (KWS) system. Our proposed KWS system will process as following: in first stage, called automatic processing or phonetic decoding, the inputted utterance will be presented by its corresponding phones sequence using the CLDNN architecture (described previously) in combination with the Hidden Markov Models, and in second stage, called keywords detection, the predefined keywords will be searched and identified from this phonetic flow using the Classification and Regression Tree (CART).

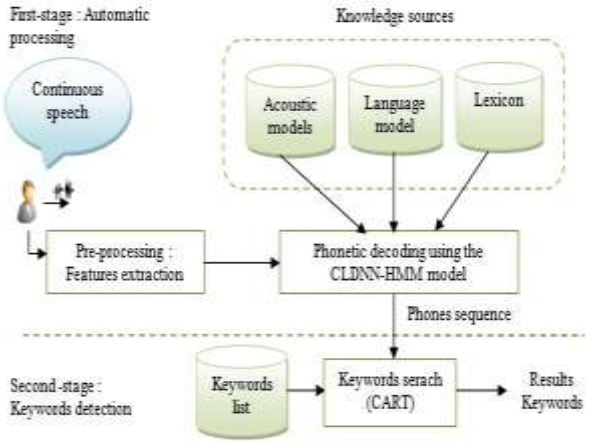

Fig. 2. The proposed two-stage Keywords Spotting system.

The CART is machine learning method which may predict how a given input may correspond to a specific output according to some contextual factors. The CART algorithm is defined by a number of yes or no questions about the contextual factor assigned to each non terminal node. For each possible answer, a new branch leading to the next question will be defined. A tree structure will be so designed using all these questions, where the specific output will be maintained by the end of this tree, known as terminal node or leaf.

Like all machine learning techniques, the CART grow necessities a training set defined by inputs or features and their corresponding labels or outputs. In our keywords spotting system, we used the CART to retrieve the grapheme corresponding to a given phoneme. First, for training an alignment between the spelling and pronunciations of the words existing in training set must be done. These alignments will generate a number of grapheme-phoneme correspondences; in such a way each phoneme in a word will get its corresponding grapheme (the null graphemic or phonemic symbol "_" is used when the spelling and pronunciation of a word are with different lengths). These different

(doi:1444-8939.2018/5-4/MRR.46)

correspondences will be used to train a number of classification and regression trees, one for each phoneme.

The CART has a number of tied nodes; for each non terminal node a contextual factor is attributed. According to the value of this contextual factor, the tree will be so crossed starting from a parent node and passing next to a child one until attending the leaf. This leaf maintains the grapheme corresponding to the phoneme. The tree building and the node splitting are defined using the minimum entropy criterion given as: [33]

$$H(L \mid node) = -\sum_{l \in L} P(l \mid node)\ \log_2 P(l \mid node) \quad (1)$$

where L corresponds to the set of admissible graphemes, and $P(l \mid node)$ corresponds to the grapheme's occurrences for a given node.

The value of this entropy differs from a node to another, because the value (label) assigned to a node is depending on the different choices made starting from the parent node. According to phoneme and grapheme contexts, the parent node will be splitted into two child nodes, where we can define the average entropy as:

$$H(L \mid child_1, child_2) = H(L \mid child_1)P(child_1) + H(L \mid child_2)P(child_2) \quad (2)$$

where $P(child_1)$ and $P(child_2)$ refer to the probabilities of reaching the two child nodes, i.e, the probabilities for which the contextual factor C_i falls respectively, into C_1^j and C_2^j, which are the two subsets obtained by the partition C_i^j of the values of C_i, where i and j denote the indexes presenting the phonemes and the partition of theirs values. [33]

The best splitting $C_{i,best}^{j,best}$ defines the maximum difference between the entropies values before and after splitting. This difference is represented by $I(L, C_i^j)$ and it corresponds to the average of mutual information between the graphemes to predict and the splitting: [33]

$$I(L, C_i^j) = H(L \mid parent) - H(L \mid child_1, child_2) \quad (3)$$

The best splitting $C_{i,best}^{j,best}$ is obtained by getting first the partition $C_i^{j,best}$ which maximizes $I(L, C_i^j)$ and then by maximizing $I(L, C_i^{j,best})$.

$$C_i^{j,best} = \arg\max_j I(L, C_i^j) \quad (4)$$

$$C_{i,best}^{j,best} = \arg\max_i I(L, C_i^{j,best}) \quad (5)$$

These two steps will be repeated for all the child nodes and be stopped when the maximum of $I(L, C_i^j)$ is inferior to a predefined threshold, for which the entropy reduction is considered insignificant. [33]

In test stage, all the CARTs trained for each phoneme will be used to retrieve the predefined keywords. Each keyword is obtained using the phonetic flow, generated by an automatic phone recognition system based on the "CLDNN-HMM" model described previously, and by taking at a time the phoneme on left and on right of the current phoneme. The CART corresponding to the current phoneme will be so traversed starting from the root node until reaching the leaf. The grapheme corresponding to the current phoneme is maintained by this leaf. These steps will be repeated for the following phoneme and its corresponding grapheme will be retrieved similarly. Finally, the searched keyword is obtained by concatenating all the predicted graphemes that have been obtained.

After testing the KWS system and having all the putative keywords occurrences, we will evaluate the obtained results. Generally, the errors of keywords spotting are due to two principles scenarios. The first one happens when the KWS system doesn't detect a keyword, which actually pronounced in the inputted utterance. And the second one happens when the KWS system detects a keyword, which is not pronounced in the inputted utterance. The first error is so a "missing" and the second error is a "false alarm". [7], [8]

- Missed Detection Rate (missing): For a given keyword, q, it can be defined as:

  $$P_{miss}(q) = \frac{N_{miss}(q)}{N_{True}(q)} \quad (6)$$

  where $N_{miss}(q)$ is the number of missed detections and $N_{True}(q)$ is the number of reference occurrences.

(doi:1444-8939.2018/5-4/MRR.46)

- The rate of total number of missed detections can be defined by the average of missed detection rates obtained for all the keywords:

$$P_{miss} = \frac{1}{K} \sum_{q=1}^{K} \frac{N_{miss}(q)}{N_{True}(q)} \quad (7)$$

where K is the total number of keywords

- Detection Rate or accuracy: is the number of references keywords occurrences correctly detected by the KWS system. For a given keyword, q, the detection rate may be computed as:

$$P_{correct}(q) = 1 - P_{miss}(q) \quad (8)$$

- False Alarm Rate: For a given keyword, q, the false alarm rate can be defined as:

$$P_{FA}(q) = \frac{N_{FA}(q)}{N_{NT}(q)} \quad (9)$$

where $N_{FA}(q)$ is the number of false alarms and $N_{NT}(q)$ is the non-target trials.

- The overall rate of false alarms produced by the KWS system can be calculated as the average of false alarm rates of all the keywords:

$$P_{FA} = \frac{1}{K} \Sigma_{q=1}^{K} P_{FA}(q) \quad (10)$$

We can take two others measures as evaluation metrics, which are recall and precision rates.

- The recall rate can be expressed in terms of the number of total detections to make $N_{True}(q)$ and the keywords correctly detected $N_{correct}(q)$.

$$\text{Recall} = \frac{100^{*} N_{correct}(q)}{N_{True}(q)} \quad (11)$$

- The precision rate can be expressed in terms of the number of keywords correctly detected $N_{correct}(q)$, and the number of false alarms $N_{FA}(q)$.

$$\text{Precision} = \frac{100^{*} N_{correct}(q)}{N_{\text{correct}(q)} + N_{FA}(q)} \quad (12)$$

After presenting our approaches for phonetic decoding of continuous speech and for keywords spotting, as well as presenting the evaluation metric used for the KWS system we will present in next section the experiments setup and the obtained results.

4. Experimental setup and results on TIMIT

4.1. Experimental results of phone recognition

The TIMIT data corpus contains 6,300 sentences produced by 630 speakers of 8 major dialects of American English. After eliminating all the SA sentences we obtain a training set with 3,696 sentences recorded by 462 speakers. The test set contains192 sentences recorded by 24 speakers. In order to validate our results and to adjust the network parameters, a random 10% of the training set was removed and considered as development (dev) set and it contains 400 sentences recorded by 50 speakers.

4.1.1. Baselines experiments

In all experiments presented in this work, a bigram language was generated using the training set. A well trained "GMM-HMM" model, with 1946 tied context dependent HMM states, was used to generate the training labels through forced alignment. The phoneme label outputs were transformed to the classic set of 39 labels. We used the Kaldi toolkit for all the following experiments [32].

We will describe firstly the baselines CNN, DNN and LSTM models experimented in this works. The CNN contains two convolution layers of 128 and 256 filters, respectively. The limited weight sharing scheme (LWS) is used. A max-pooling layer with a pooling size of 6 and a sub-sampling factor of 2 is added after the first convolution layer. And no pooling layer was added after the second convolution layer. Finally, to achieve the CNN building four fully connected layers, with 1024 hidden units, are added.

The LSTM alternative used in first step is a deep LSTMP network (DLSTMP) composed by a stack of 2 LSTMP layers, each of them with 1024 memory cells per layer and 512 projection units. For DLSTMP training the truncated back-propagation though time (BPTT) learning algorithm is used.

The DNN model used in these experiments is consisting of a few number of feed-forward layers trained in a supervised way. These deep

(doi:1444-8939.2018/5-4/MRR.46)

layers are all containing 1024 hidden units and using sigmoid activation function.

The CNN and DNN models are trained with the classic stochastic gradient decent (SGD) back-propagation algorithm. To get stable convergence, we must make a sensible choice of the initial and final learning rates. For fine-tuning, we used an initial rate of 0.0004 for the CNN and of 0.008 for the DNN. These rates will be then halved for each increasing in cross-validation frame accuracy in a single epoch lesser than 0.5%. In all experiments the SGD algorithm used mini-batches containing 256 frames.

The inputted features to all these networks are 40-dimensional filterbank features (FBANK features), along with their first and second temporal derivatives.

The phone error recognition (PER) rates of these different neural networks presented above (CNN, DNN, and DLSTMP) are illustrated in Table 1.

Table 1. Experiments with DNN, CNN and DLSTMP models

Method	***PER %* *(dev core)***	***PER %* *(test core)***
DNN (6 layers)	20.45	21.18
CNN	17.43	18.83
DLSTMP (2 layers)	17.76	18.97

These results confirm that significant improvements may be achieved using CNN over DNN, for the TIMIT phone recognition task. The success of CNNs is due to their invariance to small frequency shifts. This lets them more robust to speaker variations than DNNs. Also, the reported performances show that the two LSTMP layers (DLSTMP) may lead to phone error recognition rates more reduced than those obtained using DNNs and close to those obtained using CNNs. To improve more the performances of DLSTMP compared to CNN we must increase the number of LSTMP layers in the DLSTMP model.

4.1.2. The "CLDNN-HMM" model

In following experiments, we will use the same configurations of CNN, DNN and DLSTMP models, as described previously, to be combined in the single CLDNN model in order to make a fair comparison between all these architectures.

In first step, we start by introducing different possible combinations of CNN, LSTM and DNN, to justify the significance of the CLDNN architecture. First, we add 2 LSTMP layers after the CNN; this combination will be called "CNN-DLSTMP". Then, similarly we add 2 LSTMP layers after the DNN containing 3 fully-connected feed-forward layers and this combination will be called "DNN-DLSTMP".

Table 2. Experiments with CNN- DLSTMP vs DNN- DLSTMP

Method	***PER %* *(dev core)***	***PER %* *(test core)***
DNN-DLSTMP	20.13	20.82
CNN-DLSTMP	17.05	18.41

From these results, we can notice that the improvements contributed by CNN over DNN still existing even when it is used in combination with a DLSTMP model.

Now, we investigate the effect of adding DNN layers after the DLSTMP network. Table 3 summarizes the obtained results in terms of the number of DNN layers.

Table 3. Experiments with DLSTMP-DNN

DNN layers	***PER %* *(dev core)***	***PER %* *(test core)***
0 (DLSTMP)	17.76	18.97
1	17.62	18.83
2	17.54	18.71
3	17.48	18.66
4	17.63	18.79

We can conclude that adding more DNN layers may further reduce the phone error rates and ameliorate the performances; however adding more than 3 layers will not bring more improvements. We may also confirm that adding DNN layers after achieving the temporal modeling within the DLSTMP architecture allows mapping the output of the final LSTMP

(doi:1444-8939.2018/5-4/MRR.46)

layer to a more discriminative space and to get more easy prediction of the outputs targets.

We report now the phone error rates obtained with the CLDNN model. This model performs in three steps: first, passing the input features into a CNN, then the output of the CNN is passed to a stack of 2 LSTMP layers (DLSTMP). Finally, after performing frequency and temporal modeling, we pass the final LSTMP layer output into 3 DNN layers. Table 4 shows the results of "CNN-DLSTMP", "DLSTMP-DNN" and CLDNN models.

Table 4. Phone recognition rate with CLDNN model

Method	*PER % (dev core)*	*PER % (test core)*
DLSTMP	17.76	18.97
CNN-DLSTMP	17.05	18.41
DLSTMP-DNN	17.48	18.66
CLDNN	16.77	18.10

The results illustrated in this table indicate that the performances obtained by the CLDNN model can bring up to 0.99% improvement in the recognition rates over the DLSTMP model used alone for the dev set, and up to 0.87% for the test set. The CLDNN model takes benefits from the complementarity of the individual modeling capabilities of the three neural networks (CNN, DLSTMP and DNN) used alone; consequently, it can achieve more promising performances.

4.1.3. Further "CLDNN-HMM" experiments

In following experiments, we propose other modifications in different subnetworks of the CLDNN model, studied previously, to bring more improvements in performances.

In first step we investigate the impact of the pooling strategy used for the CNN model. We compare 2 different pooling operations, namely max and average pooling. The configurations of the two other subnetworks are kept the same as defined previously.

Table 5. Phone recognition rate of a CLDNN-HMM model with different pooling strategies

Method	*PER % (dev core)*	*PER % (test core)*
Max	16.77	18.10
Average	17.09	18.45

The results illustrated in this table indicate that the max-pooling function leads to more efficient results than the average-pooling function. The max-pooling function has the ability to emphasize the transients, in contrary of the average one which smooths them out. For that, in all the following experiments, we will use a CNN with max-pooling function in the CLDNN model.

We pass now to show the effect of using different LSTM alternatives, for that we introduce a set of experiments using in the CLDNN model a stack of 2 BLSTM layers with 1024 memory cells per hidden layer (512 memory cells in each forward and backward direction), and a stack of 2 BLSTMP layers with 1024 memory cells (512 for forward and 512 for backward layer) and with a 512-node projection layer.

For training the stack of BLSTM and BLSTMP layers, the context sensitive-chunk BPTT (CSC-BPTT) algorithm is used.

Table 6. Experiments with different deep LSTM alternatives in the CLDNN model

Method	*PER % (dev core)*		*PER % (test core)*	
	DLSTM	***CLDNN***	***DLSTM***	***CLDNN***
LSTMP – 2 layers	17.76	16.77	18.97	18.10
BLSTM – 2 layers	17.38	16.43	18.52	17.61
BLSTMP – 2 layers	17.19	16.22	18.01	17.06

The obtained results show that the CLDNN models using a stack of 2 BLSTMP layers or 2 BLSTM layers brings further improvements in phone recognition rates over the one using unidirectional LSTMP model. Unlike the unidirectional LSTMPs which are able to exploit only the past history, the bidirectional LSTM are able to exploit both the previous and future context in speech signals, consequently, they are more performing. We can also notice from these

(doi:1444-8939.2018/5-4/MRR.46)

reported results that the CLDNN model using BLSTMP layers can bring more improvements in phone recognition rates compared to the one using BLSTM layers.

After we have shown that using BLSTMP layers in the CLDNN model for the TIMIT phone recognition task gives more improved results and achieves interesting gains in performances, we pass now to investigate deeper CLDNN models by increasing the depth of BLSTMP layers used in the CLDNN architecture.

Table 7. Experiments with deeper BLSTMP and CLDNN models

Method	*PER % (dev core)*		*PER % (test core)*	
	DLSTM	***CLDNN***	***DLSTM***	***CLDNN***
BLSTMP – 2 layers	17.19	16.22	18.01	17.06
BLSTMP – 3 layers	16.70	15.66	17.53	16.49
BLSTMP – 4 layers	16.33	15.21	17.19	16.13

We can notice that a deeper CLDNN model achieves interesting improvements over a shallow one. Increasing the number of BLSTMP layers in the CLDNN architecture may efficiently model the long-range history and the non-linear relationship structures. The deep CLDNN architecture helps further to reduce the PER and to give promising recognition results when adding more BLSTMP layers, the best phone recognition rate was obtained using 4 BLSTMP layer in the CLDNN model, however increasing the number of BLSTMP layers than 4 seems to saturate the performance and can't bring more consistent improvements.

Finally, we will investigate the effect of the inputted features on the obtained results. Different experiments were done using a CLDNN model with four BLSTMP layers and using different features types. We propose to use 39 dimensional MFCC features, 40 dimensional FBANK features (as experimented previously) and LDA+STC+FMLLR features.

These later features are generated by splicing 11 frames of 13 dimensional MFCC features; a linear discriminant analysis LDA will then applied. The MFCC features are normalized using the cepstral mean-variance normalization (CMVN). Next, a semi-tied covariance (STC) transform is applied on the previous features. Finally, these features will be adapted to speaker variations using the feature-space maximum likelihood linear regression (FMLLR).

Table 8. Experiments for CLDNN model with different features types

Features	**PER % (dev core)**	**PER % (test core)**
MFCC	15.72	16.67
FBANK	15.21	16.13
FMLLR	14.74	15.81

The reported results show that a CLDNN model using four BLSTMP layers and FMLLR features on input will lead to a phone error rate of 15.81% that is the lowest phone error rate obtained along this work.

4.2. Experimental results of KWS in continuous speech using our proposed two-stage based approach

Our proposed keywords spotting system will be tested on the phonetic transcriptions generated using the "CLDNN-HMM" model (experimented previously) with four BLSTMP layers, three hidden DNN layers and using FMLLR features on input.

In these experiments we pre-define forty-two words from the TIMIT data set like keywords. These keywords may be organised into three groups: Short-length Keywords Group (SKG) for the keywords with 4, 5 and 6 phones. Medium-length Keywords Group (MKG) for the keywords with 7, 8 and 9 phones. Finally, Long-length Keywords Group (LKG) for the keywords with more than 9 phones. The results of the keywords spotting system are presented in table 9.

Table 9. Results of keywords spotting for different words length using our proposed approach

Keywords group	**Short-length Keywords**	**Medium-length Keywords**	**Long-length Keywords**
Search terms	14	14	14
Reference occurrences	109	133	102
Missed Detection Rate	0.43	0.25	0.17

(doi:1444-8939.2018/5-4/MRR.46)

Correct Detection Rate	0.56	0.74	0.82
False alarm rate	0.40	0.11	0.01
Precision %	58.49	89.62	97.67
Recall %	56.88	71.42	82.35

The reported results show that the highest detection rate is obtained for the long-length keywords and the lowest for the short-length keywords. Also, these results show that the lowest missed detection rate is obtained for the long-length keywords and the highest for the short-length keywords. Finally, we can notice that the highest false alarms rate will be obtained for the short-length keywords group compared to the two other groups.

This can be explicated by the fact that the false alarm error is generally due to three main causes. The first one is the punctuation treating problem, for example if the KWS system detects the word "home's" while we defined the word "homes" as a keyword, this detection will be considered as false alarm. The second cause, is the conflict of treating the word boundaries and the compound words, for example if the KWS system generates like detection the word "any way" while we defined the word "anyway" as a keyword, this detection will be considered as false alarm. Similarly, to detecting the word "something" while we defined the word "some thing" as a keyword. The third cause is the presence of the keyword's pronunciation in other words, for example, the pronunciation of the keyword "higher" can be included in others words as the word "hierarchies" in this case the detection will be considered as a false alarm, or the presence of the keyword's pronunciation in two consecutive words, for example the pronunciation of the word "clean" ("k l iy n") can be included in the two successive words "likely not" (pronounced "l ay k l iy n aa t"), so if the KWS generates a detection of the keyword "clean" from this words sequence it will be considered as a false alarm.

The proposed keywords spotting system will produce more false alarms for the short-length keywords, because they are the most probable to be a subpart of other keywords or to be composed by coupling two words on the speech, contrariwise to the long length keywords. Consequently, the performances obtained with longer keywords are more promising.

5. Conclusion

In this work we have presented the single deep architecture called CLDNN. We have shown that this model is more efficient than all its subnetworks; DNN, CNN and LSTM used alone. Interesting improvements have been achieved, by taking benefits from the complementary capabilities of these networks.

The lowest error rate of 15.81% for the TIMIT test set has been achieved using a CLDNN model with four BLSTMP layers and FMLLR features on input, which is a very promising result obtained for the TIMIT phone recognition task.

These efficient results of phone recognition have improved the performances of our proposed two-stage keywords spotting system.

A set of experiments was presented to test our KWS system. The obtained results have shown that we may get very interesting performances.

As a future work, we would like to ameliorate our deep CLDNN architecture by combining other neural networks in such a way we further improve the phone recognition rates and consequently improve the accuracy of the proposed KWS system.

References

1. J.S.Bridle, "An efficient elastic-template method for detecting given words in running speech", British Acoustic Metting, pp.1-4,1973.
2. A.L.Higgins and R.E.Wohlford, "keyword recognition using template concatenation", in Proceedings of the International Conference on Audio Speech and Signal Processing (ICASSP), pp. 1233-1236 (1985).
3. J. G. Wilpon, L. R. Rabiner, C.-H. Lee, and E. R. Goldman, "Automatic Recognition of Keywords in Unconstrained Speech Using Hidden Markov Models", in IEEE Trans on Acoustics, Speech, and Signal Processing 38, 1870-1878 (1990).
4. R.C.Rose and D.B.Paul, "A Hidden Markov Model based keyword recognition system",

(doi:1444-8939.2018/5-4/MRR.46)

in Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 129-132 (1990).

5. I. Szoeke, P. Schwarz, P. Matejka, L. Burget, M. Karafiat, M. Fapso and J. Cernocky, "Comparison of Keyword Spotting Approaches for Informal Continuous Speech", in Proceedings of the 9th European Conference on Speech Communication and Technology (EUROSPEECH),pp. 633-636 (2005).
6. D. R. H. Miller, M. Kleber, C. Kao, O. Kimball, T. Colthurst, S. Lowe, R. M. Schwartz, and H. Gish, "Rapid and accurate spoken term detection," in Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH), pp. 314-317 (2007).
7. I. Chen, "Resource-dependent acoustic and language modeling for spoken keyword search", Ph.D. dissertation, Electrical and Compuer. Eng. Dept., Georgia Institute of technologie, 2016.
8. J. Noguerales, "Contributions to Keyword Spotting and Spoken Term Detection For Information Retrieval in Audio Mining", Ph.D. dissertation, Eng, Inf. Dept., Madrid Univ., Madrid ESPAGNE, 2009.
9. L. Deng and J. Platt, "Ensemble Deep Learning for Speech Recognition," in Proceedings of the 15th Annual Conference of International Speech Communication Association (INTERSPEECH), pp. 1915-1919 (2014).
10. R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to Construct Deep Recurrent Neural Networks," in Proceedings of the 2nd International Conference on Learning Representation (ICLR) (2014).
11. T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, Long Short-Term Memory, Fully Connected Deep Neural Networks," in Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 4580-4584 (2015).
12. R. Zazo, T. N. Sainath, G. Simko and C. Parada, "Feature learning with raw-waveform CLDNNs for Voice Activity Detection," in Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH), pp. 3668-3672 (2016).
13. G. Dahl, D. Yu, L. Deng, and A. Acero, "Large vocabulary continuous speech recognition with context-dependent DBN-HMMs," in Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP) (2011).
14. G. E. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," Neural Comput., vol. 18, pp. 1527–1554, 2006.
15. O. A. Hamid, A. Mohamed, H. Jiang, and G. Penn, "Applying Convolutional Neural Network Concepts to Hybrid NN-HMM Model for Speech Recognition," in Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 4277-4280 (2012).
16. O. A. Hamid, L. Deng, and D. Yu, "Exploring Convolutional Neural Network Structures and Optimization Techniques for Speech Recognition," in Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH) (2013).
17. L. Tôth. "Combining time and frequency domain convolution in convolutional neural network-based phone recognition". In Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 190-194 (2014).
18. T. N. Sainath, A. Mohamed, B. Kingshury, and B. Ramabhadran, "Deep Convolutional Neural Networks for LVCSR," in Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 8614-8618 (2013).
19. D. Palaz, R. Collobert, and M. Magimai.-Doss, "Estimating phoneme class conditional probabilities from raw speech signal using convolutional neural networks," in Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH), pp. 1766-1770 (2013).
20. D. Palaz, R. Collobert, and M. Magimai. -Doss, "End-to-end Phoneme Sequence

(doi:1444-8939.2018/5-4/MRR.46)

Recognition using Convolutional Neural Networks," ArXiv e-prints, Dec. 2013.
21. H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH) (2014).
22. H. Sak, O. Vinyals, G. Heigold, A. Senior, E. McDermott, R. Monga, and M. Mao, "Sequence discriminative distributed training of long short-term memory recurrent neural networks," in Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP) (2014).
23. A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," Neural Networks, vol. 12, pp. 5–6, 2005.
24. D. Yu, L. Deng, and G. Dahl, "Roles of pre-training and fine-tuning in context-dependent DBN-HMMs for real-world speech recognition," in NIPS Workshop on Deep Learning and Unsupervised Feature Learning (2010).
25. A. Mohamed, G. Hinton and G. Penn, "Understanding how deep belief networks perform acoustic modeling". in Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp.4273-4276 (2012).
26. A. Mohamed, T. Sainath, G. Dahl, B. Ramabhadran, G. Hinton and M. Picheny, "Deep Belief Networks Using Discriminative Features for Phone Recognition," in Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 5060-5063 (2011).
27. G. E. Dahl, M. Ranzato, A. Mohamed, and G. E. Hinton, "Phone recognition with the mean-covariance restricted Boltzmann machine," in Advances in Neural Information Processing Systems 23, J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, Eds., Advances in Neural Information Processing Systems, pp. 469-477 (2010).
28. G. E. Dahl, D. Yu, L. Deng, and Al. Acero,"Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition" ,in IEEE Transactions on Audio, Speech, and Language Processing, pp. 30-42 (2012).
29. A. Mohamed, "Deep Neural Network acoustic models for ASR", Ph.D. dissertation, Computer science. Dept., Toronto Univ., Toronto, U.K., 2014.
30. N. Jaitly, P. Nguyen, AW. Senior, and V. Vanhoucken, "Application of pretrained deep neural networks to large vocabulary speech recognition", in Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH) (2012).
31. A.Graves, A. Mohammed, G. Hinton. "Speech recognition with deep recurrent neural networks". In Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 6645-6649 (2013).
32. D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi speech recognition toolkit", in IEEE 2011 Workshop on Automatic Speech Recognition and Understanding. IEEE Signal Processing Society, 2011.
33. T.Dutoit, "Introduction to text-to-speech synthesis", Kluwer academic publishers, London (1997).
34. A.M.Moghadam and M.H. Zadeh, "The Impact of Particle Swarm Optimization Method on the Improvement of Bankruptcy Predictability Using Neural Networks", MAGNT Research Report, Vol.3 (2). PP: 397-406, 2015.

(doi:1444-8939.2018/5-4/MRR.46)