DE GRUYTER OPEN

Journal of ELECTRICAL ENGINEERING, VOL 67 (2016), NO2, 124–130

DOMAIN ADAPTATION OF DEEP NEURAL NETWORKS FOR AUTOMATIC SPEECH RECOGNITION VIA WIRELESS SENSORS

Gábor Gosztolya [*] [**] — **Tamás Grósz** [*]

Wireless sensors are recent, portable, low-powered devices, designed to record and transmit observations of their environment such as speech. To allow portability they are designed to have a small size and weight; this, however, along with their low power consumption, usually means that they have only quite basic recording equipment (e.g. microphone) installed. Recent speech technology applications typically require several dozen hours of audio recordings (nowadays even hundreds of hours is common), which is usually not available as recorded material by such sensors. Since systems trained with studio-level utterances tend to perform suboptimally for such recordings, a sensible idea is to adapt models which were trained on existing, larger, noise-free corpora. In this study, we experimented with adapting Deep Neural Network-based acoustic models trained on noise-free speech data to perform speech recognition on utterances recorded by wireless sensors. In the end, we were able to achieve a 5% gain in terms of relative error reduction compared to training only on the sensor-recorded, restricted utterance subset.

K e y w o r d s: wireless sensors, speech recognition, deep neural networks, domain adaptation

1 INTRODUCTION

Wireless sensors and wireless sensor networks have become increasingly popular recently. Their main application is the monitoring of their environment like motion detection [1], measuring light and temperature [2] and object localization [3,4]. They are also capable of recording audio data [5], which calls for porting a number of speech processing applications (*eg* automatic speech recognition [6], speaker verification [7] and emotion recognition [8, 9]). However, these applications tend to be quite sensitive to the recording conditions, which usually do not match those created via wireless sensors. That is, while standard audio databases are typically recorded in environments having only a minimal amount of background noise and using a decent microphone, utterances recorded by wireless sensors often have a high level of background noise, and the microphone installed on such sensors is generally of mediocre or poor quality.

To overcome these problems, the most straightforward approach is perhaps to collect an audio database recorded by the wireless sensors which is large enough to train an accurate model for the actual speech processing application. However, since recording and annotating dozens of hours of utterances is quite a tedious process, we should look for a more sophisticated solution, especially if we already have a large audio database that has been annotated. Another idea might be to play a noise-free audio dataset on some speaker, and record it via the sensors: this way we could avoid the need for collecting ne audio recordings, and, more importantly, the existing annotation of the recordings could be used for the newly recorded utterances (or they could be converted cheaply). Still, this is not a quick and easy solution either.

A third option might be to train our model (classification or regression method) on the noise-free database, and adapt it to the acoustic conditions of the sensors. This approach has the advantage that, compared to the previous approaches, only a fraction of sensor-recorded utterances are required, and we can benefit from the large noise-free audio database which could be used for acoustic model training.

In this paper, we will focus on the adaptation of deep neural networks (DNNs) trained as the acoustic units for automatic speech recognition (ASR). For this, we re-record a subset of a speech recognition database on the wireless sensor, and experiment with three strategies for constructing an acoustic model for sensor-based utterances: DNN training on this subset, DNN model adaptation, and joint training (training on both the noise-free and the sensor-recorded utterances)

The structure of this paper is as follows. First, we describe the automatic speech recognition task. Next, we turn to Deep Rectifier Networks, and describe the common techniques used for acoustic model adaptation. Then we turn to the description of our experimental setup (*eg* the properties of the wireless sensors used, our way of evaluating the performance, and the database and feature sets used). Lastly, we present and analyze our results, and draw some conclusions.

[*] University of Szeged, Szeged, ggabor@inf.u-szeged.hu; , groszt@inf.u-szeged.hu [**] MTA-SZTE Research Group on Artificial Intelligence of the Hungarian Academy of Sciences, Hungary

DOI: 10.1515/jee-2016-0017, Print (till 2015) ISSN 1335-3632, On-line ISSN 1339-309X

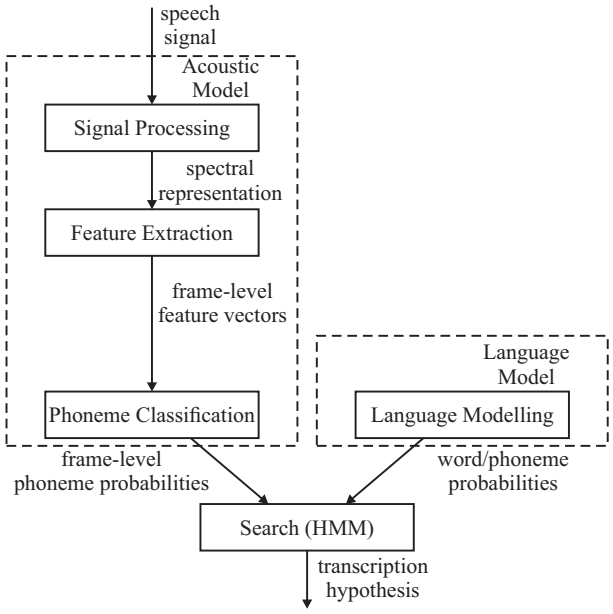

Fig. 1. The workflow of automatic speech recognition

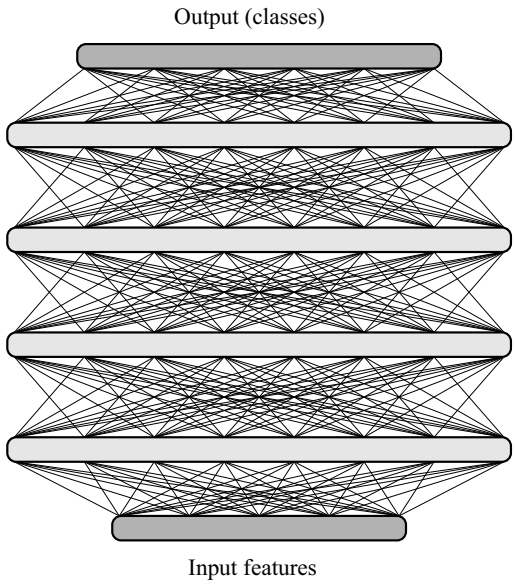

Fig. 2. The layout of a deep neural network (DNN) with four hidden layers

2 AUTOMATIC SPEECH RECOGNITION

In automatic speech recognition we are given a sound recording of the speech of a user (an *utterance*), and our aim is to produce its written transcription. Although it is undoubtedly a machine learning task, over the decades researchers have developed dedicated tools for it, most notably the hidden Markov model (HMM) [10]. This is mainly because speech recognition is special in the sense that it has to handle an input of a varying size, and the output is not of fixed length. To overcome this, the input is usually divided into small, equal-sized portions, but the sequence of these samples still has to be processed jointly in some way. In particular, each small portion of the speech signal (*frame*) has to be identified as one of the possible phonemes in the given language, and then combined somehow into a probable *phoneme sequence*.

After some signal processing and feature extraction steps, we have to classify the individual frames, based on the frame-level feature vectors calculated so far, into one of the possible phonemes in the given language. This step is called the *phoneme classification* task, and here general machine learning methods could be used. Recall that we perform speech recognition to get the transcript of the *whole utterance*, so phoneme classification is only of use to us if it can lead to high-precision transcriptions. Therefore the frame-level results of phoneme classification have to be combined, which is usually done via a HMM. For this step, instead of the resulting classes (*eg* phonemes) for each frame, their estimated likelihoods are considered; so phoneme classification has the requirement that besides achieving a good frame-level classification accuracy, we also have to obtain precise class conditional probabilities. (Due to this, this subtask is also often referred to as *phoneme posterior estimation*.)

The steps outlined so far make up the *acoustic model* of a speech recognizer, which describes the relation between the phoneme sequences and the speech signal. To achieve state-of-the-art recognition accuracy scores, however, we usually have to incorporate some general knowledge about the structure of the given human language [6]; this information is encoded by the separate *language model*. Although language models could be constructed in several ways (*eg* by applying context-free grammars [11] or by relying on word similarity [12]), in practice the most commonly used solution is n-grams. The concept of n-grams is that the probability of the nth word depends solely on the preceding $n-1$ words, where the corresponding probability values are estimated by using statistical methods on huge (textual) datasets. It is common to employ bigrams ($n=2$) and trigrams ($n=3$); here we will employ word trigrams in our experiments.

For the above reasons, usually Gaussian mixture models (GMM) [13] and artificial neural networks (ANN) [10] were applied as acoustic models. Recently a variation of ANN called deep neural networks (DNN) has emerged, and it has now become the dominant solution for acoustic modeling in ASR. Next, we will describe DNNs in more detail.

3 DEEP NEURAL NETWORKS

Since the invention of deep neural networks in 2006, their role has become evermore important in the phoneme classification subtask of speech recognition. Deep neural networks differ from traditional artificial neural networks in that besides the input and the output layers, the latter have only one or two hidden layers, while DNNs have several (three or more) hidden layers. (See Fig. 2) The efficient training of a deep network with several hidden layers was not possible though, as the traditional back-propagation method was unable to train the bottom layers. The reason for this is that with the standard sigmoid

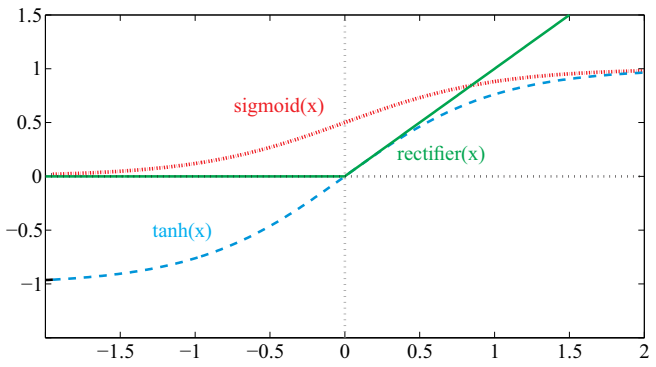

Fig. 3. The sigmoid, tanh and linear rectifier functions

and tanh activation functions, the gradients in the lower layers tend to be close to zero ("vanishing gradient effect"), hence the weights in those layers barely change and cannot be trained.

All this changed when Hinton et al. invented the method they called "DBN Pre-Training" [14]. This efficient unsupervised algorithm can be used for learning the connection weights of a Deep Belief Network (DBN), which consists of several layers of restricted Boltzmann machines (RBMs). As the name implies, RBMs are a variant of Boltzmann machines, with the restriction that their neurons must form a bipartite graph. They have an input layer representing the features of the given task, a hidden layer which has to learn some representation of the input, and each connection in an RBM must be between a visible unit and a hidden unit.

Hinton *et al* showed that the weights resulting from the unsupervised pre-training algorithm can be used to initialize the weights of a deep, but otherwise standard, feed-forward neural network. After this initialization step, we can readily apply the backpropagation algorithm to fine-tune the network weights by utilizing a supervised criterion.

"Discriminative pre-training" was proposed by Seide *et al* [15] as an alternative to DBN pre-training. It is a simple algorithm where we first train a network with one hidden layer to full convergence using backpropagation; then we replace the softmax output layer by another randomly initialized hidden layer and a new softmax layer on top, and we train the network again. This process is repeated until we reach the desired number of hidden layers. This is very similar to the greedy layer-wise training algorithm of Bengio *et al* [16], but differs in that Bengio only updates the newly added hidden layers. Seide et al. found that this method yields the best results if one performs only a few iterations of backpropagation in the pre-training phase (instead of training to full convergence) with an unusually large learn rate. In their article, they concluded that this simple training strategy performs just as well as the much more complicated DBN pre-training method described above [15].

In the next method it is not the training algorithm that is slightly modified, but the neurons. Instead of the usual sigmoid activation function, here we apply the rectifier function $\max(0, x)$ for all hidden neurons [17] (see Fig. 3). There are two main differences between the sigmoid and the rectifier functions. One is that the output of rectifier neurons does not saturate as their activity gets higher. Glorot *et al* conjecture that this is very important in explaining their good performance in deep nets: because of this linearity, there is no vanishing gradient effect [17]. The other difference is the hard saturation at 0 for negative activity values: because of this, only a subset of neurons are active for a given input. One might argue that this could harm optimization by blocking gradient backpropagation, but the experimental results do not support this view. It seems that the hard nonlinearities do no harm so long as the gradient can propagate along some path. In fact, some claim that this sparsity of the hidden layers offers many advantages [17], such as allowing the model to be more robust and less sensitive to small changes (*eg* noise) present in the input features.

The principal advantage of deep rectifier nets is that they can be trained with the standard backpropagation algorithm, without any pre-training. In our previous experiments on phoneme classification using several databases, they were found to yield phone recognition scores similar to those of sigmoid networks pre-trained with the DBN algorithm [18, 19], but their training times were much shorter. Therefore, in the experiments performed in our study, we decided to just employ deep rectifier neural networks.

4 ADAPTATION OF DNN ACOUSTIC MODELS

It is a well-known fact that DNN based acoustic models trained on clean speech perform very badly on speech recorded in a different, noisy environment (*eg* [20, 21]). To make the DNNs more robust to noise two approaches could be used, namely feature enhancement and model adaptation.

Feature enhancement methods generally attempt to reduce or remove the effects of the corrupting noise in the audio signal by creating new features based on the noisy ones, on which the unaltered acoustic model is evaluated. Some techniques seek to find noise-resistant features [22], while others try to remove the noise from the speech data [23] or learn a transformation from the noisy speech features to the clean one [24]. As these methods do not modify the acoustic model, the computational cost of these methods tends to be quite low; however, based on the results of experiments, their potential is limited.

Model-domain methods use the inputs unaltered and modify the DNN parameters to be more representative of the observed speech [25]. To be able to adapt the DNN, we need to record some noisy speech, which we can use as training data. While model adaptation techniques typically achieve higher accuracy scores than enhancement methods do, they usually require significantly larger noisy datasets, and their computational cost is higher as well.

As for wireless sensors the amount of audio data is substantial enough to allow model adaptation, hence we

Fig. 4. The spectral representation of the same speech excerpt recorded by the TV tuner card (left) and by the wireless sensor (right)

opted for this technique. That is, we first trained a DNN until it attained full convergence on clean speech data. This network was then trained further using the dataset recorded by the wireless sensors. Note that adaptation methods usually focus on adapting to the actual speaker; now we would like to retain speaker-independence, but our recording conditions differ significantly from those of the training database, so we will perform domain adaptation.

It is well known (*eg* [24]) that the lower layers of a deep neural network are responsible for low-level feature extraction, while the higher layers perform more abstract and more task-dependent functions. As in our case only the acoustic conditions change, while the task remains the same, it seems reasonable that it might be enough to train just a few lower layers of the network instead of adapting all the weights. This way, we might achieve the same level of accuracy with faster training.

We will also test a method similar to the multicondition training (*eg* [20, 26]) by mixing the clean and noisy speech data together and training the acoustic DNN on this joint dataset. This way, we may train a network to perform well in both environments [26], and we will rate this strategy by the accuracy score measured on the utterances recorded via the wireless sensors.

5 EXPERIMENTAL SETUP

5.1 The Wireless Sensors Used

In our study we used Crossbow Iris sensor nodes (motes) that have a 7.37 MHz processor with a RAM of 8K bytes and a programmable flash memory of 128K bytes. The microphone and other input peripherials are located on a piece of hardware that can be attached to the mote, on the so-called sensor board. We had Crossbow MTS300 sensor boards, which, besides the microphone, also contain light and temperature sensors.

5.2 The Method of Evaluation

To measure the performance of a method for a standard classification task, usually just the classification accuracy is calculated as the ratio of correctly classified examples. If the classes are unbalanced, this method can be extended to calculate the classification accuracy for each class, and then we take the mean of these scores. This approach could be followed with an isolated word recognizer, where word-level accuracy scores can be readily determined as the number of exact word matches; however, measuring the performance of a continuous speech recognition application is a bit more complicated, since we would like to differentiate between a sentence which contains only a small mistake and a completely misunderstood one.

In the case of sentence recognition the common method is to calculate the word-level edit distance (or Levenshtein-distance) of the two word sequences (the reference and the resultant); that is, we construct the resulting sentence from the real transcript by using the following operations: inserting and deleting words, and replacing one word with another one. These operations have some cost (in speech recognition the common values chosen are 7, 7 and 10, respectively), and then we pick an operation set having the lowest cost. Now we can calculate the accuracy metric as

$$\text{Accuracy} = \frac{N - S - D - I}{N}\,, \tag{1}$$

where N is the total number of words in all the original utterances, S is the number of substitutions, D is the number of deletions and I is the number of insertions. (Note that in theory this score can be negative, as there is no theoretical upper bound for the number of insertions, although this is so only for quite inefficient recognizer configurations.)

5.3 The Database Used

The speech corpus of Hungarian broadcast news [18] was collected from eight TV channels via a TV tuner card. From the 28 hours of recordings, 22 hours are used as the training set, on which the acoustic neural networks are trained. Another 2 hours of recordings was assigned to the development set (used for setting the meta-parameters of a speech recognition system), and 4 hours to the test set (used for final model evaluation). We built a trigram language model from a corpus of about 50 million words taken from the www.origo.hu news portal, using the language modelling tools of HTK [27]. As Hungarian is an

Table 1. The frame-level and word-level accuracy scores obtained by the different training and domain adaptation strategies, for both feature sets

Feature set	Method	Frame	Word	
		Dev	Dev	Test
FBANK	DNN Adaptation	62.3 %	73.2 %	73.0 %
	Joint training	60.8 %	72.6 %	71.5 %
	Random initialization	60.2 %	71.9 %	71.6 %
MFCC	DNN Adaptation	60.9 %	72.8 %	72.2 %
	Joint training	60.0 %	71.9 %	69.9 %
	Random initialization	59.0 %	71.5 %	70.9 %

agglutinative language with a lot of word forms, we kept only those words that occurred at least twice in the corpus, reducing the recognition dictionary to about half a million words. Afterwards, the pronunciations of these words were obtained from the 'Hungarian Pronunciation Dictionary' [28].

We re-recorded a smaller subset of this database using our wireless sensors: about 5 hours of the training, about 1 hour of the development set and about 2 hours of the test set. (We will refer to this subset as the *restricted* set.) Due to the quite different recording hardware, the two versions differed significantly: while the original utterances had a sampling rate of 16000 Hz, this value for the utterances recorded via the wireless sensors was 8861 Hz. Furthermore, the characteristics of the two microphones were also very different. Figure 4 shows the spectrum of an utterance recorded by the tuner card (left) and by the wireless sensor (right). It is clear that the higher frequencies were not captured by the wireless sensor (indicated by the blank region at the top) due to the lower sampling rate; furthermore, this spectrogram appears more distorted, and there is a constant buzz which is displayed as a continuous horizontal line. Here, we will refer to this re-recorded data subset as the *sensor-recorded* set.

5.4 Experimental Setup

We tested two feature sets, namely 40 mel filter bank energies ("FBANK") and 12 mel-frequency spectral coefficients ("MFCC", [6]), along with energy, and their first and second order derivatives. Decoding and evaluation was performed by applying a modified version of HTK [27]. We employed our custom neural network implementation, which achieved outstanding results earlier on several datasets (*eg* [29, 30]). Following preliminary tests, we opted for five hidden layers, each one containing 1000 rectified neurons, and we applied the softmax activation function in the output layer. As weight regularization, we employed the L2 normalization technique.

The recognition accuracy score of an ASR system can be improved if we use different models for the same phone pronounced in a different context. These context-dependent models are called triphones (or context-dependent states, CD-states), and their application in ASR is standard practice nowadays. We constructed the set of CD states by using the Kullback-Leibler divergence-based state tying method proposed in [31]. The final set contained 1843 CD states.

6 RESULTS

We tested three approaches for carrying out ASR on the sensor-recorded utterances: training only on the sensor-recorded (restricted) utterance set (as here we train on this subset after randomly initializing the weights of the DNN, we call this strategy *random initialization*), training on the utterances of both the full clean training set and the sensor-recorded one (*joint training*), and training on the full clean utterance set and then adapting the DNN acoustic model on the sensor-recorded set (*DNN adaptation*).

Table 1 shows the accuracy scores obtained by using both feature sets with these strategies. (The best values are underlined.) It can be seen that, for both feature sets, DNN adaptation resulted in the highest word-level accuracy score among the three tested strategies. Using the FBANK feature set, although the 73.2 % score is only slightly above the 72.6 % score obtained by the joint training strategy, it is far above that got via training only the restricted subset recorded by the sensors (71.9 %), resulting in a roughly 5 % improvement in terms of Relative Error Reduction (RER). On the more important test set, the improvements are more significant as the latter two approaches perform pretty similarly (71.5 % and 71.6 %, joint training and random initialization, respectively), while the 73.0 % achieved by DNN adaptation is significantly higher than these values (over 5 % RER). At the frame-level, the DNN adaptation strategy is also more successful than the other two tested methods. (Note that, although both noise-free and sensor-recorded utterances were present in the development set of the joint training strategy, we just list the scores measured on the latter utterances.)

The results got by using the MFCC feature set display similar trends as we found by training on the FBANK feature set: the accuracy scores achieved by the DNN adaptation strategy are higher than those of the random initialization and joint training ones. However, the values are generally lower by 1–2 %, due to the well-known fact that neural networks tend to perform better on raw features such as FBANK than on traditional spectral ones such as MFCC.

The fact that DNN adaptation outperformed the random initialization strategy, in our opinion indicates that by first training on a larger speech corpus recorded under different acoustic conditions, then adapting to the acoustic specialities of the wireless sensor, we can efficiently make use of the more training instances present in the noise-free database.

Although the DNN acoustic models trained via this joint strategy can be expected to perform well under both conditions, they are suboptimal on each of them. If it is unlikely that we have to handle both kinds of utterances

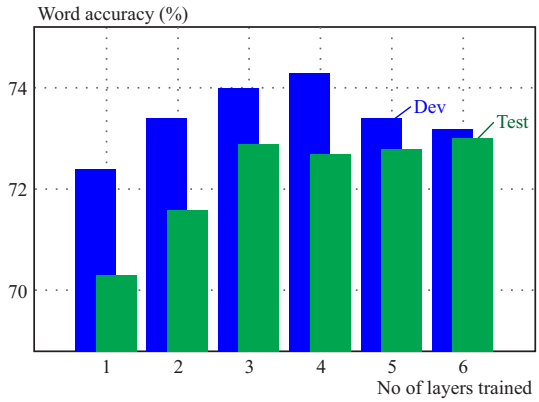

Fig. 5. The accuracy scores obtained as a function of the number of layers adapted, using the FBANK feature set

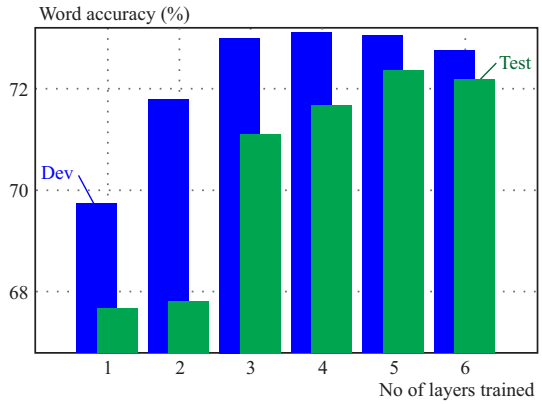

Fig. 6. The accuracy scores obtained as a function of the number of layers adapted, using the MFCC feature set

Table 2. The frame-level and word-level accuracy scores obtained by the different training and domain adaptation strategies, using the FBANK feature set

Method	Layers	Frame	Word	
		Dev %	Dev %	Test %
DNN Adaptation	1	57.6	72.4	70.3
	2	61.9	73.4	71.6
	3	61.7	74.0	72.9
	4	62.0	74.3	72.7
	5	62.3	73.4	72.8
	All	62.3	73.2	73.0

Table 3. The frame-level and word-level accuracy scores obtained by the different training and domain adaptation strategies, using the MFCC feature set

Method	Layers	Frame	Word	
		Dev %	Dev %	Test %
DNN Adaptation	1	55.9	69.7	67.8
	2	59.0	71.8	67.8
	3	60.2	73.0	71.1
	4	60.8	73.1	71.7
	5	60.9	73.1	72.4
	All	60.9	72.8	72.2

with the same acoustic model, it is best to adapt the acoustic deep neural networks.

6.1 Partial DNN Adaptation

As it is known that the hidden layers of a Deep Neural Network perform different functions within the same task, it may happen that we do not need to adapt all of them for optimal performance, or even that we can get better results by adapting only a subset of the layers. To this end, we also experimented with following this strategy. As now we perform domain adaptation, it was logical to adapt only the lower layers; we did this using both feature sets (FBANK and MFCC).

Tables 2 and 3 show the accuracy scores obtained this way; the best values and the ones within a 0.3% threshold are underlined. (Note that we had five hidden layers, which, together with the input and output layers results in seven layers overall, leading to six layer-layer connections to adapt.) We can see that to maximize the frame-level accuracy it is best to adapt all layers; this is logical, though, as we perform DNN training (and adaptation) on the level of frames. Surprisingly, word-level accuracy on the development set is optimal when we adapt the connections only by the fourth hidden layers, but this does not hold for the test set, where the training of further layers is required for optimal performance. Examining figures 5 and 6 we may deduce that at least the first three layers have to be adapted to get close-to-optimal accuracy scores. When relying on the FBANK feature set, adjusting the weights between the further layers yielded only a slight or no improvement over this score, while when we used the MFCC feature vectors, optimality on the test set was achieved by adapting all connections except the ones between the last hidden layer and the output layer.

7 CONCLUSIONS

Wireless sensors are low-powered devices designed to monitor their environment. In this study we examined ASR performance on utterances recorded via wireless sensors. Due to the small microphone installed on these devices, the sound quality of utterances recorded by such sensors differs significantly from that of the larger audio databases usually used for acoustic DNN training, typically recorded in sound-proof environments and with high-quality microphones. As it is quite expensive to record a large speech database on wireless sensors, it is sensible to make use of these larger speech corpora even when we seek to carry out speech recognition using such sensors. To this end, besides training on a database subset recorded by the wireless sensors, we tested two ways of making use of a noise-free, larger audio corpus: joint training on both corpora in parallel, and training the acoustic Deep Neural Network on this larger, clean dataset, and then adapting it to the acoustic conditions of the one recorded via the sensors. In the end we found

that joint training was not significantly better than training on the sensor-recorded, noisy database subset, while DNN adaptation turned out to perform significantly better. By following this strategy, we were able to achieve a 5% improvement in terms of relative error reduction.

Acknowledgment

The Titan X graphics card used for this research was donated by the NVIDIA Corporation.

References

[1] BURCHFIELD, T. R.—VENKATESAN, S.: Accelerometer-based Human Abnormal Movement Detection in Wireless Sensor Networks, Proceedings of ACM SIGMOBILE Workshop (2007), 67–69.

[2] HAYES, J.—BEIRNE, S.—LAU, K. T.—DIAMOND, D.: Evaluation of a Low Cost Wireless Chemical Sensor Network for Environmental Monitoring, IEEE Sensors Journal **64** No. 06 (2008), 530–533.

[3] GOGOLÁK, L.—PLETL, SZ.—KUKOLJ, D.: Neural Network-based Indoor Localization in WSN Environments, Acta Polytechnica Hungarica **10** No. 06 (2013), 221–235.

[4] GOGOLÁK, L.—KUKOLJ, D.—FÜRSTNER, I.: Wireless Sensor Network Based Localization in Industrial Environments, Analecta **8** No. 1 (2014), 91–96.

[5] GOSZTOLYA, G.—TÓTH, L.: Improving the Sound Recording Quality of Wireless Sensors Using Automatic Gain Control Methods, Scientific Bulletin of "Politehnica" University of Timisoara, Transactions on Automatic Control and Computer Science **56** No. 2 (2011), 47–56.

[6] RABINER, L.—JUANG, B. H.: Fundamentals of Speech Recognition, Prentice Hall, Upper Saddle River, NJ, USA, 1993.

[7] FURUI, S.: Cepstral Analysis Technique for Automatic Speaker Verification, Acoustics, Speech and Signal **29** No. 2 (1981), 254–272.

[8] TÓTH, SZ. L.—SZTAHÓ, D.—VICSI, K.: Speech Emotion Perception by Human and Machine, Proceedings of COST Action (2012), 213–224.

[9] GOSZTOLYA, G.—BUSA-FEKETE, R.—TÓTH, L.: Detecting Autism, Emotions and Social Signals Using AdaBoost, Proceedings of Interspeech (2013), 220–224.

[10] MORGAN, M.—BOURLARD, H.: An Introduction to Hybrid HMM/Connectionist Continuous Speech Recognition, Signal Processing Magazine (May 1995), 1025–1028.

[11] NEDERHOF, M.-J.: Practical experiments with regular approximation of context-free languages, Journal of Computational Linguistics **26** No. 1 (2000), 17–44.

[12] VARGA, I.—OHTAKE, K.—TORISAWA, K.—DESAEGER, S.—MISU, T.—MATSUDA, S.—KAZAMA, J.: Similarity Based Language Model Construction for Voice Activated Open-Domain Question Answering, Proceedings of IJCNLP (2011), 535–544.

[13] DUDA, R. O.—HART, P. E.: Pattern Classification and Scene Analysis, John Wiley & Sons, New Jersey, 1973.

[14] HINTON, G. E.—OSINDERO, S.—TEH, Y.-W.: A Fast Learning Algorithm for Deep Belief Nets, Neural Computation **18** No. 7 (2006), 1527–1554.

[15] SEIDE, F.—LI, G.—CHEN, X.—YU, D.: Feature Engineering in Context-Dependent Deep Neural Networks for Conversational Speech Transcription, Proceedings of ASRU (2011), 24–29.

[16] BENGIO, Y.—LAMBLIN, P.—POPOVICI, D.—LAROCHELLE, H.: Greedy Layer-Wise Training of Deep Networks, Advances in Neural Information Processing Systems **19** (2007), 153–160.

[17] GLOROT, X.—BORDES, A.—BENGIO, Y.: Deep Sparse Rectifier Networks, Proceedings of AISTATS (2011), 315–323.

[18] GRÓSZ, T.—TÓTH, L.: A Comparison of Deep Neural Network Training Methods for Large Vocabulary Speech Recognition, Proceedings of TSD (2013), 36–43.

[19] TÓTH, L.: Phone Recognition with Deep Sparse Rectifier Neural Networks, Proceedings of ICASSP (2013), 6985–6989.

[20] SELTZER, M.—YU, D.—WANG, Y.: An Investigation of Deep Neural Networks for Noise Robust Speech Recognition, Proceedings of ICASSP (2013), 7398–7402.

[21] KOVÁCS, GY.—TÓTH, L.: Joint Optimization of Spectro-Temporal Features and Deep Neural Nets for Robust Automatic Speech Recognition, Acta Cybernetica **22** No. 1 (2015), 117–134.

[22] JAIN, P.—HERMANSKY, H.—KINGSBURY, B.: Distributed Speech Recognition Using Noise-Robust MFCC and TRAPS-estimated Manner Features, Proceedings of Interspeech (2002), 473–476.

[23] AGARWAL, A.—CHENG, Y. M.: Two-Stage Mel-Warped Wiener Filter For Robust Speech Recognition, Proceedings of ASRU (1999), 12–15.

[24] GAO, T.—DU, J.—DAI, L.-R.—LEE, C.-H.: Joint Training of Front-end and Back-end Deep Neural Networks for Robust Speech Recognition, Proceedings of ICASSP (2015), 4375–4379.

[25] LIAO, H.—GALES, M. J. F.: Adaptive Training with Joint Uncertainty Decoding for Robust Recognition of Noisy Data, Proceedings of ICASSP (2007), 389–392.

[26] HUANG, Y.—SLANEY, M.—SELTZER, M. L.—GONG, Y.: Towards Better Performance with Heterogeneous Training Data in Acoustic Modeling Using Deep Neural Networks, Proceedings of Interspeech (2015), 845–849.

[27] YOUNG, S.—EVERMANN, G.—GALES, M. J. F.—HAIN, T.—KERSHAW, D.—MOORE, G.—ODELL, J.—OLLASON, D.—POVEY, D.—VALTCHEV, V.—WOODLAND, P. C.: The HTK Book, Cambridge University Engineering Department, Cambridge, UK, 2006.

[28] ABARI, K.—OLASZY, G.—ZAINKÓ, CS.—KISS, G.: Hungarian Pronunciation Dictionary on Internet (in Hungarian), Proceedings of MSZNY (2006), 223–230.

[29] TÓTH, L.: Phone Recognition with Hierarchical Convolutional Deep Maxout Networks, EURASIP Journal on Audio, Speech, and Music Processing **2015** No. 25 (2015), 1–13.

[30] GRÓSZ, T.—BUSA-FEKETE, R.—GOSZTOLYA, G.—TÓTH, L.: Assessing the Degree of Nativeness and Parkinson's Condition Using Gaussian Processes and Deep Rectifier Neural Networks, Proceedings of Interspeech (2015), 1339–1343.

[31] GOSZTOLYA, G.—GRÓSZ, T.—TÓTH, L.—IMSENG, D.: Building Context-Dependent DNN Acousitc Models Using Kullback-Leibler Divergence-Based State Tying, Proceedings of ICASSP (2015), 4570–4574.

Received 20 November 2015

Gábor Gosztolya (PhD), received his MSc and PhD degrees from the University of Szeged, Szeged, Hungary in 2001 and 2010, respectively. He is working at the University of Szeged, and at the Research Group on Artificial Intelligence of the Hungarian Academy of Sciences and University of Szeged. His research interests are speech recognition, various speech technologies, and applied machine learning.

Tamás Grósz received his BSc and MSc degrees from the University of Szeged, Szeged, Hungary in 2011 and 2013, respectively. He is currently a PhD student at the University of Szeged. His research interests are speech recognition, deep learning technologies and image processing.