Towards deeper understanding: Deep convex networks for semantic utterance classification

Article *in* Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on · March 2012

DOI: 10.1109/ICASSP.2012.6289054

4 authors:

Gokhan Tur
Microsoft
163 PUBLICATIONS **3,523** CITATIONS
SEE PROFILE

li Deng
Zhejiang Normal University
400 PUBLICATIONS **15,833** CITATIONS
SEE PROFILE

Dilek Hakkani-Tur
Microsoft
245 PUBLICATIONS **4,631** CITATIONS
SEE PROFILE

Xiaodong He
Microsoft Research & University of Washington
164 PUBLICATIONS **5,596** CITATIONS
SEE PROFILE

TOWARDS DEEPER UNDERSTANDING: DEEP CONVEX NETWORKS FOR SEMANTIC UTTERANCE CLASSIFICATION

Gokhan Tur[1,2] *Li Deng*[2] *Dilek Hakkani-Tür*[1,2] *Xiaodong He*[2]

[1]Microsoft Speech Labs, [2]Microsoft Research

gokhan.tur@ieee.org, deng@microsoft.com, dilek@ieee.org, xiaohe@microsoft.com

ABSTRACT

Following the recent advances in deep learning techniques, in this paper, we present the application of special type of deep architecture — deep convex networks (DCNs) — for semantic utterance classification (SUC). DCNs are shown to have several advantages over deep belief networks (DBNs) including classification accuracy and training scalability. However, adoption of DCNs for SUC comes with non-trivial issues. Specifically, SUC has an extremely sparse input feature space encompassing a very large number of lexical and semantic features. This is about a few thousand times larger than the feature space for acoustic modeling, yet with a much smaller number of training samples. Experimental results we obtained on a domain classification task for spoken language understanding demonstrate the effectiveness of DCNs. The DCN-based method produces higher SUC accuracy than the Boosting-based discriminative classifier with word trigrams.

Index Terms— deep convex networks, spoken language understanding, domain detection, semantic utterance classification, deep learning

1. INTRODUCTION AND MOTIVATION

Spoken language understanding (SLU) in human/machine spoken dialog systems aims to automatically identify the domain and intent of the user as expressed in natural language and to extract associated arguments or slots [1] to achieve a goal.

Given an utterance, SLU in dialog systems extracts semantic information from the output of an automatic speech recognizer (ASR). The dialog manager (DM) then determines the next machine action given the SLU output. In the last decade, a variety of practical goal-oriented spoken dialog systems have been built for limited domains. Three key tasks in such targeted dialog and understanding applications are domain classification, intent determination and slot filling [2]. Domain classification is often completed first in SLU systems, serving as a top-level triage for subsequent processing.

Similar to intent determination, domain detection is often framed as a semantic utterance classification (SUC) problem, where the target domain is a discrete set of semantic classes without structural relationship among them. Usually, supervised classification methods are used to estimate conditional probabilities, and a set of labeled utterances is used in training. Such systems typically use established classification algorithms, such as Boosting [3], support vector machines (SVMs) [4], or maximum entropy models [5].

It is only very recently that researchers have started experimenting with deep belief networks (DBNs) for SUC (e.g., [6]). While deep networks are shown to have better modeling capacity than most established classifiers, they have not been popular until recently. This is partly due to the problem of scalability with larger data sets for complex tasks. In particular, DBNs [7] have recently gained popularity in various areas of information processing applicaitons. DBNs are stacks of Restricted Boltzmann Machines (RBMs) followed by fine tuning. RBM is a two-layer network, which can be trained reasonably efficiently in an unsupervised fashion. Following the introduction of this RBM learning and layer-by-layer construction of deep architectures, DBNs have been successfully used for numerous tasks in speech and language processing, including acoustic modeling [8, 9, 10] and intent determination [6].

Following the success of DBN, Deng and Yu have proposed the use of Deep Convex Net (DCN), which directly attacks the scalability issue of DBN-like deep learning techniques [11]. In this paper, we present experimental results with this more recent deep learning technique for SUC. DCN is shown to be superior to DBN, not only in terms of accuracy, but also in training scalability and efficiency [11, 12]. DCN can also incorporate the strength of DBN by training the weights at the lowest level using RBM. While DCN has been applied to acoustic modeling successfully, SUC poses significant challenges for DCN. Two important issues are the size of the input feature space, and the typically much smaller amounts of training data available. For example, typical acoustic models use 39 standard MFCC features, with millions of frames as training samples. In our training data set with only 16,000 utterances, there are already as many as 125,000 unique trigrams as potential features, forming a very sparse space. In this paper, we present experimental results using the DCN technique for domain detection, and present methods that reduce the feature space by relying on a set of n-grams chosen as weak learners by Boosting.

In the next section, we describe the task of semantic utterance classification and the related work in more detail. Then in Section 3, we briefly present the deep convex network learning technique and present how DCNs are used for semantic classification. In Section 4 we show the experimental results before concluding in Section 5.

2. SEMANTIC UTTERANCE CLASSIFICATION

The semantic utterance classification (SUC) task aims at classifying a given speech utterance X_r into one of M semantic classes, $\hat{C}_r \in \mathcal{C} = \{C_1, \ldots, C_M\}$ (where r is the utterance index). Upon the observation of X_r, $\hat{C}_r$ is chosen so that the class-posterior probability given X_r, $P(C_r|X_r)$, is maximized. More formally,

$$\hat{C}_r = \arg\max_{C_r} P(C_r|X_r). \quad (1)$$

Semantic classifiers need to allow significant utterance variations. A user may say *"I want to fly from San Francisco to New York next Sunday"* and another user may express the same information by saying *"Show me weekend flights between JFK and SFO"*. In spite

of this variability, utterances in such applications have a clear structure that binds together the specific pieces of information. Not only is there no *a priori* constraint on what the user can say, these systems also need to generalize well from a tractably small amount of training data. For example, the phrase *"Show all flights"* and *"I want to fly"* should be interpreted as variants of a single semantic class "*Flight.*" On the other hand, the command *"Show me the weekend snow forecast"* should be interpreted as an instance of another semantic class, say, "*Weather*." In order to do this, the selection of the feature functions $f_i(C, W)$ aims at capturing the relation between the class C and word sequence W. Typically, binary or weighted n-gram features, with $n = 1, 2, 3$, to capture the likelihood of the n-grams, are generated to express the user intent for the semantic class C [1]. As an example, binary bigram feature functions are in the form of:

$$f^{BG}_{c,w_x w_y}(C_r, W_r) = \begin{cases} 1, \text{ if } c = C_r \wedge w_x w_y \in W_r, \\ 0, \text{ otherwise.} \end{cases} \quad (2)$$

That is, if two consecutive words, $w_x w_y$, appear in the word sequence W and if the class is one of interest, C_r, then the binary feature function takes on a value of one; it is zero otherwise. Once the features are extracted from the text, the task becomes a text classification problem. Traditional text categorization techniques devise learning methods to maximize the probability of C_r, given the text W_r; i.e., the class-posterior probability $P(C_r|W_r)$.

Early work on spoken utterance classification has been done mostly for call routing or intent determination system, such as the AT&T How May I Help You? (HMIHY) system [13], relying on salience phrases, or the Lucent Bell Labs vector space model [14]. With advances in machine learning, especially in discriminative classification techniques, in the last decade, researchers have been able to apply off-the-shelf classification algorithms. Typically word n-grams are used as features after preprocessing with generic entities, such as dates, locations, or phone numbers. Because of the very large dimensions of the input space, large margin classifiers such as SVMs [4] or Boosting [3] were found to be very good candidates. In this paper, instead, we explore the use of DCNs for this task.

3. DEEP CONVEX NETWORK FOR SEMANTIC CLASSIFICATION

In this section, we first briefly describe the general DCN architecture and then present how features are extracted to be fed to the DCN for our SUC task.

3.1. General DCN architecture

A typical DCN architecture, shown in Fig.1, includes a variable number of layered modules, wherein each module is a specialized neural network consisting of a single hidden layer and two trainable sets of weights. Specifically, the lowest module in the DCN comprises a first linear layer with a set of linear input units, a non-linear layer with a set of non-linear hidden units, and a second linear layer with a set of linear output units. In this paper, the input units are associated with features computed from the input utterances.

The hidden layer of the lowest module of a DCN comprises a set of non-linear units that are mapped to the input units by way of a first, lower-layer weight matrix, which we denote by $W_i, i = 1, 2, 3, 4$ in Fig. 3. For instance, the weight matrix may comprise a plurality of randomly generated values between zero and one, or the weights of a restricted Boltzmann machine (RBM) trained separately. The non-linear units may be sigmoidal units that are configured to perform non-linear operations on weighted outputs from the input units.

The second, linear layer in any module of a DCN includes a set of output units that are representative of the targets of classification. For instance, if the DCN is configured to perform digit recognition, then the plurality of output units may be representative of the values 1, 2, 3, and so forth up to 10, with a 0-1 coding scheme. If the DCN is configured to perform speech recognition, then the output units may be representative of phones, the hidden Markov model (HMM) states of phones, or the context-dependent HMM states of phones. In this paper, the output units are the semantic classes associated with the input utterances. The non-linear units in each module of the DCN may be mapped to a set of the linear output units by way of a second, upper-layer weight matrix, which we denote by $U_i, i = 1, 2, 3, 4$ in Fig. 3. This second weight matrix can be learned by way of a batch learning process, such that learning can be undertaken in parallel. Convex optimization can be employed in connection with learning U_i. For instance, U_i can be learned, based at least in part, on the first weight matrix W_i, the values of the coded classification targets, and the values of the input units.

The DCN includes a set of serially connected, overlapping, and layered modules, wherein each module includes the aforementioned three layers – a first linear layer that includes a set of linear input units whose number equals the dimensionality of the input features, a hidden layer that comprises a set of non-linear units whose number is a tunable hyper-parameter, and a second linear layer that comprises a plurality of linear output units whose number equals that of the target classification classes. The modules are referred to herein as being "layered" because the output units of a lower module are a subset of the input units of an adjacent higher module in the DCN. More specifically, in a second module that is directly above the lowest module in the DCN, the input units can include the output units of the lower module(s). The input units can additionally include the raw training data. In other words, the output units of the lowest module can be appended to the input units in the second module, such that the input units of the second module also include the output units of the lowest module.

The pattern discussed above can continue for many modules. A resultant learned DCN may then be deployed in connection with an automatic classification task such as frame-level speech phone or state classification, as performed in [11], and classification of semantic categories of input utterances, as is the focus of this paper.

3.2. Training Algorithm for DCN

The convex optimization technique discussed here applies to all layers of a DCN. Implementation of the technique differs for distinct layers, mainly in ways of setting the weight matrices $\mathbf{W}$, which varies its dimensionality across layers before applying the technique presented below.

We assume a supervised learning setting where both the training data $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_N]$ and the corresponding labeled target vectors $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_i, \ldots, \mathbf{t}_N]$, where each target $\mathbf{t}_i = [\mathbf{t}_{1i}, \ldots, \mathbf{t}_{ji}, \ldots, \mathbf{t}_{Ci}]^T$, are available. We use the loss function of mean square error to learn weight matrices $\mathbf{U}$, assuming $\mathbf{W}$ is given. That is, we aim to minimize the error of

$$E = Tr[(\mathbf{Y} - \mathbf{T})(\mathbf{Y} - \mathbf{T})^T], \quad (3)$$

where $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_i, \ldots, \mathbf{y}_N]$. Importantly, if the weight matrix W is determined already (e.g., via judicious initialization), then the hidden layer values $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_i, \ldots, \mathbf{h}_N]$ are also determined. Consequently, the weight matrix $\mathbf{U}$ can be determined by setting the gradient

$$\frac{\partial E}{\partial \mathbf{U}} = 2\mathbf{H}(\mathbf{U}^T\mathbf{H} - \mathbf{T})^T \quad (4)$$

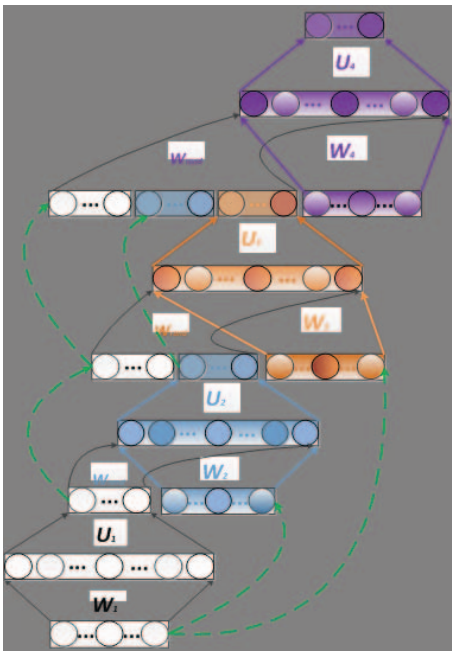

Fig. 1. A typical DCN architecture with four modules illustrated in four separate colors.

to zero. This is a well established convex optimization problem and has a straightforward closed-form solution, known as pseudo-inverse:

$$\mathbf{U} = (\mathbf{H}\mathbf{H}^T)^{-1}\mathbf{H}\mathbf{T}^T. \quad (5)$$

A further, batch-mode, parallelizable fine tuning algorithm improves classification accuracy upon the basic algorithm described above. In contrast to fine tuning for DBN of [7], our fine tuning adjusts the weight matrices **W** and **U** layer by layer, not involving any global fitting over the entire architecture. The essence of the DCN fine tuning is to exploit the structural relationship between **W** and **U** in each layer, as expressed in Eq.5, in computing the gradient of the loss function with respect to the weight matrix **W**. Details of the fine tuning algorithm can be found in [11]. Some additional regularization is also developed and applied to handle the small training data problem.

3.3. Feature Extraction for Semantic Classification

For domain detection, the input is a sequence of words uttered by the user and as recognized by a speech recognizer. Then, typically word n-grams are extracted as features for classification.

In earlier studies using deep networks for text categorization, *the curse of dimensionality* is a very well known problem. These networks are not designed for millions of dimensions. To this end, for example, [15] tackled this problem by transforming the word n-grams into lower dimensional spaces first. This is analogous to class-based models, where *similar words* are clustered or latent semantic indexing (LSI) or latent Dirichlet allocation (LDA) based methods.

In this paper, instead of feature transformation, we relied on feature selection. While there are a number of feature selection mechanisms proposed in the machine learning literature [16, among others], we instead rely on the baseline Boosting classifier. For DCN, the input feature space is shrunk using the n-grams selected by the Boosting classifier. Boosting is an iterative algorithm; on each iteration, t, a weak classifier, h_t, is trained on a weighted training set, and at the end, the weak classifiers are combined into a single classifier [17]. For example, for semantic categorization, one can use word n-grams as features, and each weak classifier (e.g., decision stump, which is a single node decision tree) can check the absence or presence of an n-gram. This feature selection technique corresponds to selecting the most discriminative n-grams for the task. However, the weights coming with the decision stump are ignored, only binary features indicating absence or presence are used.

	No. Utt.	Avg. No. Words
Training	16,000	7.60
Development	2,000	7.66
Test	1,902	7.58

Table 1. Data sets used in the experiments.

Layer	Dev	Test
Chance (Majority)	77.45%	76.71%
Baseline (Boosting)	13.15%	13.35%
1	15.30%	15.29%
2	14.05%	13.14%
3	**13.45%**	**12.67%**
4	14.25%	13.77%
5	15.10%	14.45%

Table 2. Semantic classification error rates using deep convex nets with varying number of stacked DCN modules, compared to the Boosting baseline. RBM is used to initialize lowest-level network weights using the discriminative features selected by Boosting.

4. EXPERIMENTS AND RESULTS

In order to perform experiments with the DCNs, we compile a dataset of utterances from the users of a spoken dialog system. Table 1 shows the properties of the data sets and the (relative) frequencies of the two types of queries in each data set. Each of the utterances in these data sets is manually labeled with one of 25 domain categories. The domains were chosen to cover specific target domains such as restaurants, calendar, or movies, generic user intents such as greeting or frustration, and an *other* category for the remaining ones. Note that the *other* domain can include web-related utterances as well, such as *search for the inventor of kaleidescope*. In this set, the utterances were enforced to belong to a single domain in order to avoid multi-labeling issues.

For evaluation, the error rate of the top scoring class is used. The baseline performance is obtained using only word trigrams with Boosting method of [3]. We did not experiment with other classification algorithms, as Boosting has already been shown to give comparable results [6, 18]. Table 2 summarizes the Boosting and DCN experimental results. This data contains about 125K word trigrams, so we applied Boosting-based filtering as explained earlier, with 1,000 iterations, reducing the input feature space size to 917 unique salient n-grams. These features are then fed to a single-hidden-layer RBM to train weights for the lowest module of DCN. The optimal number of epochs for RBM training is determined to be 60, using the development set, empirically. No other RBM parameter is optimized over the RBM training method used in [19, 20]. To initialize weights for higher levels, fine-tuned weights from lower modules are used. One behavior worth noting is that, while the error rates decrease as the number of layers increase, over-fitting occurs, at layer 4, in this case. This has been observed in earlier work on speech processing as well [11].

Figure 2 shows learning curves for the semantic utterance clas-

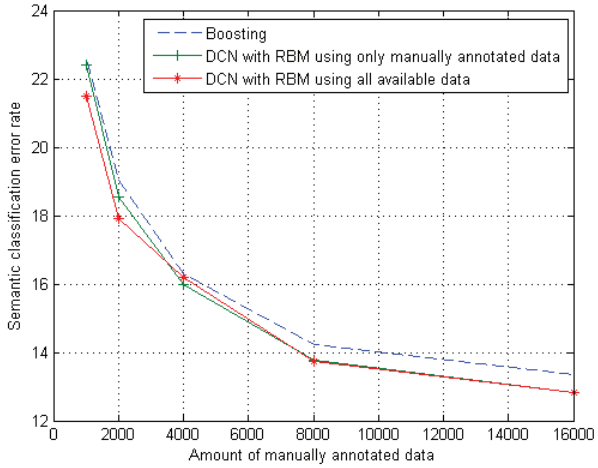

Fig. 2. Learning curves comparing Boosting with DCN with bottom layer initialized with RBM using only the annotated vs. all data.

Model	Dev	Test
Baseline (Boosting)	10.70%	10.40%
DCN	11.50%	**10.09%**

Table 3. Semantic utterance classification error rates using optimal number of features for the Boosting baseline system.

sification task, comparing the performance of Boosting with DCN with varying amounts of manually annotated data. This figure justifies the use of RBM for estimating the weights at the bottom layer as it can use all the data in an unsupervised fashion. This difference is best seen when there are only 1,000 or 2,000 manually annotated samples.

In a further experiment, DCN was fed with the optimal number of features (iterations) used for Boosting baseline system. This gives a total of 4,809 features, which were learned over 10,000 iterations. While the performance of DCN for the held-out set is lower, DCN gives better performance for the test set, validating the effectiveness of DCN's regularization mechanisms.

5. CONCLUSIONS AND FUTURE WORK

We have presented promising experimental results on semantic utterance classification using deep convex networks. Our results indicate that, even without optimizing most of the network parameters, this method already reached better performance than a non-trivial baseline, established using a state of the art discriminative classifier, Boosting, with word trigrams.

In the current work, we assume the input of the SUC system, or a module of the SLU system, is a perfect speech script. However, in real-world scenarios, the SUC or SLU module in the full SLU system including an ASR module has to deal with input with ASR errors. Therefore, joint optimization of the ASR module and the SUC module, in similar ways to the schemes developed in [5, 21], is desirable. In our previous work, we have shown that it is possible to perform an end-to-end optimization where the model parameters in all sub-systems are jointly learned via optimizing the objective function directly tied to the final performance measure defined by the full system. We have demonstrated that such an end-to-end optimization approach can give better results for complex information systems such speech translation involoving multiple sub-systems working together [22, 23]. We expect in the future similar approach can be applied to improve the end-to-end performance of SLU when either the SUC or the ASR module of the full system, or both, are constructed using the DCN-like deep architecure as described in this paper.

6. REFERENCES

[1] G. Tur and L. Deng, *Intent Determination and Spoken Utterance Classification, Chpater 3 in Book: Spoken Language Understanding*, John Wiley and Sons, New York, NY, 2011.

[2] M. Gilbert, J. G. Wilpon, B. Stern, and G. Di Fabbrizio, "Intelligent virtual agents for contact center automation," *IEEE Signal Processing Magazine*, vol. 22, no. 5, pp. 32–41, September 2005.

[3] R. E. Schapire and Y. Singer, "Boostexter: A boosting-based system for text categorization," *Machine Learning*, vol. 39, no. 2/3, pp. 135–168, 2000.

[4] P. Haffner, G. Tur, and J. Wright, "Optimizing SVMs for complex call classification," in *Proceedings of the ICASSP*, Hong Kong, April 2003.

[5] S. Yaman, L. Deng, D. Yu, Y.-Y. Wang, and A. Acero, "An integrative and discriminative technique for spoken utterance classification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 6, pp. 1207–1214, 2008.

[6] R. Sarikaya, G. E. Hinton, and B. Ramabhadran, "Deep belief nets for natural language call-routing," in *Proceedings of the ICASSP*, Prague, Czech Republic, 2011.

[7] G. E. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets," *Advances in Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[8] G. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained dnns for large vocabulary speech recognition," *IEEE Trans. Audio, Speech, and Lang. Proc.*, Jan. 2012.

[9] 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*, December 2010.

[10] N. Jaitly and G. E. Hinton, "Learning a better representation of speech sound waves using restricted boltzmann machines," in *Proceedings of the ICASSP*, Prague, Czech Republic, 2011.

[11] L. Deng and D. Yu, "Deep convex nets: A scalable architecture for speech pattern classification," in *Proceedings of the Interspeech*, Florence, Italy, 2011.

[12] L. Deng, D. Yu, and J. Platt, "Scalable stacking and learning for building deep architectures," in *Proc. ICASSP*, Kyoto, Japan, 2012.

[13] A. L. Gorin, G. Riccardi, and J. H. Wright, "How May I Help You?," *Speech Communication*, vol. 23, pp. 113–127, 1997.

[14] J. Chu-Carroll and B. Carpenter, "Vector-based natural language call routing," *Computational Linguistics*, vol. 25, no. 3, pp. 361–388, 1999.

[15] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 4, no. 2, pp. 1137–1155, 2003.

[16] Y. Yang and J. Pedersen, "A comparative study on feature selection in text categorization," in *Proceedings of the ICML*, 1997.

[17] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[18] M. Karahan, D. Hakkani-Tur, G. Riccardi, and G. Tur, "Combining classifiers for spoken language understanding," in *Proceedings of the IEEE ASRU Workshop*, U.S. Virgin Islands, December 2003.

[19] L. Deng, M. Seltzer, D. Yu, A. Acero, A. Mohamed, and G. Hinton, "Binary coding of speech spectrograms using a deep auto-encoder," in *Proc. Interspeech*, September 2010.

[20] A. Mohamed, D. Yu, and L. Deng, "Investigation of full-sequence training of deep belief networks for speech recognition," in *Proc. Interspeech*, September 2010.

[21] X.D. He and L. Deng, "Speech recognition, machine translation, and speech translation a unified discriminative learning paradigm," *IEEE Signal Processing Magazine.*, 2011.

[22] X.D. He, L. Deng, and A. Aceero, "Why word error rate is not a good metric for speech recognizer training for the speech translation task?," in *Proc. ICASSP*, Prague, 2011.

[23] Y. Zhang, L. Deng, X. He, and A. Acero, "A novel decision function and the associated decision-feedback learning for speech translation," in *Proc. ICASSP*, Prague, 2011.