Novel Generative Classifier for Acoustic Events

Paul M Baggenstoss, Kevin Wilkinghoff
Fraunhofer FKIE
Fraunhoferstraße 20, 53343 Wachtberg, Germany
p.m.baggenstoss@ieee.org, kevin.wilkinghof@fkie.fraunhofer.de

***Abstract*—In this paper, we describe a novel generative classifier for audio events based on the projected belief network (PBN). The PBN is a layered generative network formed from a feed-forward network, so it can be simultaneously trained as a generative model for a given class, and as a discriminative classifier against "all other classes". A PBN can also be shortened to any number of layers, allowing the output features of the shortened network to be modeled using an arbitrary probability density function (PDF) estimator. We exploit these properties of PBN to model the features from the output of an early convolutional layer, where a time dimension is still present, using a hidden Markov model (HMM). The special generative/discriminative training of the PBN produces generative features that are also high in discriminative information, forming a generative classifier combining (a) a discriminative deep network and (b) a generative neural network, and (c) a HMM classifier rooted in classical Bayesian approaches. The approach is demonstrated in the task of acoustic event classification.**

***Index Terms*—projected belief network, generative models, acoustic event classification**

I. Introduction

A. Motivation and Main Contributions

In the past years, acoustic event detection has shifted from using generative models such as hidden Markov models (HMMs) [1]–[3] to almost exclusively using discriminative deep learning based models such as convolutional neural networks (CNNs) [4], [5]. Still, there remains room for improvement in the design of classifiers for acoustic events, and this needed improvement may come from generative approaches. Despite the almost universal advantage of discriminative classifiers in a one-to-one comparison, generative approaches do have a place, especially in open-set problems [6], [7]. It is clear that given equal performance, a generative classifier is preferred because it possesses an inherent model of the data generation process, which can serve to create synthetic data and can help detect out-of-set events. Especially the latter is important for acoustic event detection, since for most applications the set of all sounds that may appear are not known a priori and thus only training a closed-set classification model is impractical and not sufficient.

A projected belief network (PBN) [8]–[11] is a generative model implemented by working backwards (back-projecting) through a feed-forward neural network (FFNN). The method of normalizing flows [12] has recently become popularand, similar to a PBN, can define a probability density function (i.e. generative model) from a network. But NF is a special case of PDF projection in which the network is composed of only 1:1 (bijective) transformations. Since most neural networks are composed of dimension-reducing layers, the more general PBN is required to create generative models from these.

Recently, we introduced a combined generative-discriminative architecture called discriminative alignment of projected belief network (PBN-DA) [13], in which the performance rivaled discriminative classifiers. In this paper, we propose a modification to PBN-DA, called PBN-DA-HMM, in which the features in the early convolutional layers of a PBN-DA network, where data still has a time dimension, are tapped off and passed to a HMM for generative modeling, forming a shortened PBN network with HMM feature distribution. In the PBN-DA training of the early-stages of the network, discriminative information is present. In acoustic event classification, the proposed method is shown to have about the same classification error rate as the conventional feed-forward deep neural network (DNN) and PBN-DA, while yielding about the same performance as a CNN. Moreover, when combining PBN-DA-HMM with a CNN the number of errors is significantly reduced showing that both models learn different representations and thus have a different view on the data.

B. Related Work

Similar to the presented approach, there are several works on combining autoencoders with other models to improve the performance for different tasks based on acoustic data. One example is jointly training variational autoencoders and HMMs for unsupervised acoustic unit discovery of speech [14], [15]. In [16], the intermediate representation of a discriminative model is used as input to an autoencoder for speech emotion recognition. Another example is unsupervised learning of representations for acoustic event detection [17]. Here, first an autoencoder that predicts the next frame of a Mel-spectrogram is trained and second a pairwise loss is used to take inter-sample similarity of the representations into account and thus yield more meaningful audio representations. In contrast, the proposed approach combines the generative task (i.e. autoencoder or generative model) with the discriminative task in a single network using PBN.

This work was supported jointly by the Office of Naval Research Global and the Defense Advanced Research Projects Agency under Research Grant - N62909-21-1-2024

II. Mathematical Background

A projected belief network (PBN) is unique among layered generative networks because it operates implicitly by backing-up through a feed-forward neural network (FFNN). The FFNN is the dual network for the generative process [8], [11]. The PBN is based on PDF projection, which is related to normalizing flows (NF) [12]. In NF, the data distribution is seen as a 1:1 (dimension-preserving) transformation of a simple probability density function (PDF), so its likelihood function (LF) is easily computed. For dimension-reducing transformations, which most neural networks consist of, PDF projection is required.

A. Review of PDF Projection

Subject to mild constraints, any fixed dimension-reducing transformation, $\mathbf{y} = T(\mathbf{x})$, together with the known or assumed feature distribution $g(\mathbf{y})$, corresponds to a probability density function (PDF) on the input data [18]–[20] given by

$$G(\mathbf{x}) = \frac{p_{0,x}(\mathbf{x})}{p_{0,x}(\mathbf{y})} g(\mathbf{y}), \tag{1}$$

where $p_{0,x}(\mathbf{x})$ is a prior distribution and $p_{0,x}(\mathbf{y})$ is its mapping to $\mathbf{y}$ through $T(\mathbf{x})$ (in our simplified notation, the argument of the distribution defines its range of support, and the variable in the subscript defines the original range where the distribution was defined). If $p_{0,x}(\mathbf{x})$ is selected for maximum entropy, then $G(\mathbf{x})$ is unique for a given transformation and $g(\mathbf{y})$. The PDF of the input data can be estimated by training the parameters of the transformation to maximize the mean of $\log G(\mathbf{x})$, resulting in a transformation that extracts sufficient statistics and maximizes information [21]. We say that $G(\mathbf{x})$ is the "projection" of $g(\mathbf{y})$ back to the input data. We call the term $J(\mathbf{x}) \triangleq \frac{p_{0,x}(\mathbf{x})}{p_{0,x}(\mathbf{y})}$ the "J-function" because in the special case of dimension-preserving transformations (i.e. normalizing flows [12]), $J(\mathbf{x})$ is the determinant of the Jacobian of $T(\mathbf{x})$. To generate data from $G(\mathbf{x})$ in (1), one draws a sample $\mathbf{y}$ from $g(\mathbf{y})$, then draws a sample $\mathbf{x}$ from the set $\{\mathbf{x} : T(\mathbf{x}) = \mathbf{y}\}$, weighted by the prior distribution $p_{0,x}(\mathbf{x})$.

PDF projection admits a chain-rule by applying the idea recursively to stages of a transformation. Consider a cascade of two transformations, $\mathbf{y} = T_1(\mathbf{x})$, and $\mathbf{z} = T_2(\mathbf{y})$. Then, applying (1) recursively,

$$G(\mathbf{x}) = \frac{p_{0,x}(\mathbf{x})}{p_{0,x}(\mathbf{y})} \frac{p_{0,y}(\mathbf{y})}{p_{0,y}(\mathbf{z})} g(\mathbf{z}), \tag{2}$$

which can be extended to any number of stages. The data generation is also cascaded. Note that $p_{0,x}(\mathbf{y})$ and $p_{0,y}(\mathbf{y})$ are two different distributions on the range of $\mathbf{y}$. While $p_{0,y}(\mathbf{y})$ is originally defined on the range of $\mathbf{y}$, $p_{0,x}(\mathbf{y})$ can be written $T_1[p_{0,x}(\mathbf{x})]$.

Generative models based on multiple feature extraction approaches are possible. Consider two feature extraction chains, resulting in features $\mathbf{z}_1$, $\mathbf{z}_2$, with distributions $g_1(\mathbf{z}_1)$ and $g_2(\mathbf{z}_2)$. Then, a maximum likelihood classifier can be constructed by comparing $G_1(\mathbf{x})$ and $G_2(\mathbf{x})$, where these functions are separately defined using (2) in the obvious way.

TABLE I

MaxEnt priors and activation functions as a function of input data range. TG="Trunc. Gauss.". TED="Trunc. Expon. Distr". $\mathcal{N}(x) \triangleq \frac{e^{-x^2/2}}{\sqrt{2\pi}}$ and $\Phi(x) \triangleq \int_{-\infty}^{x} \mathcal{N}(x)$.

$\mathbb{X}^N$	MaxEnt Prior $p_{0,x}(\mathbf{x})$	$\lambda(\alpha)$
$\mathbb{R}^N$	$\prod_{i=1}^{N} \mathcal{N}(x_i)$ (Gaussian)	α (Linear)
$\mathbb{P}^N$	$\prod_{i=1}^{N} 2\mathcal{N}(x_i),\ \ 0 < x_i$ (TG)	$\alpha + \frac{\mathcal{N}(\alpha)}{\Phi(\alpha)}$ (TG)
$\mathbb{U}^N$	$[0,\ 1]^N$ (Uniform)	$\frac{e^{\alpha}}{e^{\alpha}-1} - \frac{1}{\alpha}$ (TED)

B. Review of PBN

When PDF projection is applied to a FFNN layer-by-layer, this results in the projected belief network (PBN) [11]. To illustrate the sampling process in one layer, let $\mathbf{y} \in \mathbb{R}^M$ be the hidden variable at the output of a given layer of a FFNN, and let $\mathbf{x} \in \mathbb{R}^N$ be the layer input, where $N > M$. Let $\mathbf{y} = \lambda(\mathbf{b} + \mathbf{W}'\mathbf{x})$, where $\lambda(\ )$ is a strictly monotonic increasing (SMI) element-wise activation function. We seek to sample $\mathbf{x}$ given $\mathbf{y}$. Because the SMI activation function and bias are invertible, we can work with the output of the linear transformation, denoted by $\mathbf{z} = \mathbf{W}'\mathbf{x}$. Specifically, $\mathbf{x}$ is drawn randomly from the manifold $\mathcal{M}(\mathbf{z}) = \{\mathbf{x} : \mathbf{W}'\mathbf{x} = \mathbf{z}\}$, i.e. we draw $\mathbf{x}$ from the set of samples that map to $\mathbf{z}$, randomly from $\mathcal{M}(\mathbf{z})$ proportional to a prior distribution $p_{0,x}(\mathbf{x})$, which is selected according to the principle of maximum entropy [22] (MaxEnt), while meeting any constraints that include the range of the variable $\mathbf{x}$, denoted by $\mathbb{X}^N$, determined by which activation function was used in the up-stream layer. We consider three canonical ranges, $\mathbb{R}^N : x_i \in (-\infty, \infty), \forall i$, $\mathbb{P}^N : x_i \in [0, \infty), \forall i$, and $\mathbb{U}^N : x_i \in [0, 1], \forall i$. For a MaxEnt distribution to exist in the first two cases, we need constraints on the variance [1].

The canonical MaxEnt priors for the ranges $\mathbb{R}^N$, $\mathbb{P}^N$, and $\mathbb{U}^N$, are the Gaussian, truncated Gaussian (TG), and uniform distributions, respectively, which are listed in Table I. The uniform distribution is a special case of the truncated exponential distribution (TED). Additional details can be found in [8], [9]. As an aside, there is a deterministic way to sample a PBN, which results in a kind of auto-encoder [8].

C. Discriminative Alignment of PBN (PBN-DA)

It is generally assumed that the discriminative approach to classification is superior to the generative approach, because estimating the class distributions at high dimension is much harder than predicting the class label [23], [24]. Looking at a PBN, one can come to a different conclusion: the parameters of a PBN (the layer weights and bias values) simultaneously define a FFNN (which can be a discriminative classifier), and a generative model, i.e. with LF given by (2). One can think of it as a two-way street, information conceptually flows forward through the network to form a discriminative classifier, and backward to form a generative model. However, a discriminative classifier is trained on all data classes, while

[1] In the case of $\mathbb{P}^N$, a constraint on the mean is adequate, but this leads to an uninteresting solution.

generative models must be trained separately on each class. This dilemma was solved by the method of discriminative alignment [13] in which each class-dependent network is trained simultaneously as a generative model for the given class, but also as a discriminative model against "all other classes". This approach tends to "align" the network weights, giving the generative model high selectivity against the other classes, getting the best of both the generative and discriminative approaches. Despite the benefits, a PBN cannot use max-pooling in convolutional layers, limiting the performance as a classifier. Despite this, the resulting generative classifier has been shown to rival the performance of discriminative classifiers, and when combined with them, produces results that exceed either individual approach [9], [13].

D. PBN-DA-HMM

Since a PBN is a recursive structure, if it is broken in the middle, the second half acts as a PDF estimate for the hidden variables coming out of the first half, denoted by $\mathbf{h}$. Let the true distribution of $\mathbf{h}$ be written $p(\mathbf{h})$. The second part of the PBN implements $G(\mathbf{h})$, which is an estimate of $p(\mathbf{h})$, implemented by PDF projection, e.g. equation (2). However, it can be that the dimension of $\mathbf{h}$ is small enough that one can use well-known PDF estimation methods, such as HMM, to replace, and possibly improve upon $G(\mathbf{h})$. The Markovian assumption, exploited by the hidden Markov model (HMM) while over-simplified, provides an excellent compromise between tractability and generative modeling accuracy.

In traditional HMM classifier, one uses a conventional feature extractor, such as log-MEL spectrograms, then estimates class-dependent LFs using HMM. Instead, we propose to pre-train a full PBN on a given class, applying discriminative alignment, to give the feature extractor (i.e. the first part of the PBN) discriminative information against "all other classes", then, combine the class-depedent LFs using the concept at the end of Section II-A. There are two advantages to this, (a) the use of class-dependent feature extraction can potentially improve class selectivity in the generative classifier, and (b) the use of discriminative alignment in the pre-training lends discriminative information to the features.

III. Experimental Results

A. Data Selection, Feature Selection, and Feature Extraction

The ESC50 data set [25] consist of 50 data classes, with of 40 ten-second recordings in each data class. The classes are diverse, and it is difficult to represent them well by one feature extraction approach alone. As outlined in Section II-A, PDF projection allows the use of multiple feature extraction approaches in a single common generative model. But, before attempting to conduct an experiment on all 50 classes using multiple features, we wanted to conduct a limited feasibility experiment on a small data subset. To select a suitable data subset, we examined all 50 classes and extracted log-MEL-band features using overlapped Hanning-weighted processing windows, then reconstructed the time-series from the features using maximum-entropy feature inversion [26], followed by overlap-add time-series reconstruction. The time-series were played back and compared acoustically with the original. We chose appropriate FFT size, numbers of MEL bands, as well as either MEL or linear band spacing to optimize the perceived quality of the reconstructed sound. We found that of the 50 classes, 23 classes were well represented by the following features: FFT size 768, 2/3 overlap, 48 log-MEL-spaced bands. The features for each 10-second event consisted of 624 time samples, resulting in a 624×48 (time × freq) matrix. The 23 classes (classes are numbered 0-49) were 0,1,6,9,10,11,13,16-19,21,23-25,27-30,36,45,47,49. In order to obtain the most meaningful experiment, we selected the most challenging subset of these 23 classes by conducting a classification experiment using a conventional DNN. We found hat a set of 8 classes caused the bulk of the inter-class errors. These classes were: 0 (Dog bark), 13 (Crickets), 21 (Sneeze), 23 (Breathing), 24 (Coughing), 25 (Footsteps), 30 (Door knock), 49 (Sawing). We then conducted a feasibility experiment for PBN-DA-HMM using just the chosen features and these 8 classes. There were then 8×40 total events.

To partition the data, we used random 4:1 random data holdout, selecting 10 testing samples of the 40 samples of each class class at random, and trained on the remaining 30. We did this four times, independently. The partitions were designated by letters A-D. We provide these features and holdout folds online for reproducibility [27].

B. Network Architectures

We used three network architectures in the experiments, (a) the "PBN networks", which were trained separately on each of the 8 classes, (b) a "DNN classifier" which was almost identical to the networks used for the PBNs but was trained as a classifier on all classes, and (c) a state of the art "CNN classifier".

PBN networks. The seven-layer PBN network had three convolutional and 4 dense layers, ending with a classifier layer of 8 neurons. The input data is 624×48 (time × freq). Kernels in the three convolutional layers were 8 12×16 kernels, 30 10×5 kernels, and 60 21×3 kernels respectively. Downsampling was 3×4, 3×2 , and 3×1. Convolutional border modes were "valid". The dense layers had 256, 64, 16, and 8 neurons. The last layer is the cross-entropy classifier (output) layer. The output of the third convolutional layer has dimension 16×60, which is tapped off for HMM processing. For the HMM, the data is seen as having 16 time steps and a feature dimension of 60. The eight class-dependent PBN networks used linear activation at the output of the first 3 layers, in order to form a Gaussian group [9]. The remaining layers used the truncated Gaussian (TG) activation [28], similar in behavior to softplus, not unlike leaky Relu [29], but continuous (see Table I).

DNN classifier. For the DNN classifier, we used the same network as for the PBN networks, but used TG activation function in all layers, and max-pooling instead of down-sampling in the convolutional layers. We also used dropout regularization in layers 4 and 5.

CNN classifier. For the CNN, we used the system presented in [30], which is designed for anomalous sound detection but can also be used for classifying acoustic data by choosing the auxiliary task for training the system to be the classification task as done in [31]. More concretely only the sub-model for spectrogram representations, which is based on a modified ResNet architecture [32], is used. The model is trained using the sub-cluster AdaCos loss [33] with 4 sub-clusters to learn mapping the input samples to discriminative embeddings. This loss is an angular margin loss with an adaptive scale parameter similar to AdaCos [34] but uses multiple instead of a single centers for each class, called sub-clusters. When training the model, dropout with a probability of 50% applied to the hidden representations before the last dense layer [35]. Additionally, we used two data augmentation techniques to improve the performance of the CNN. As a first technique, we applied mixup [36] to the input samples and their corresponding classes using a mixing coefficient sampled from a uniform distribution. Secondly, we applied random cyclic temporal shifts to the spectrograms. The CNN is trained with a batch size of 8 for 100 epochs using adam [37] and is implemented in Tensorflow [38].

After training, the embeddings for each sample are extracted and normalized with respect to the Euclidean norm. Then, for each class k-means is applied to the embeddings of the training samples to get 4 mean embeddings. Then, the maximum cosine similarity over these 4 mean embbedings is determined and used as a similarity score for this particular class.

C. Experimental Approach and Results

For PBN-DA-HMM, we followed the method described in Section II-D, summarized as follows: (a) We first train a separate PBN on each class m using a combined cost function consisting of negative log-likelihood PBN cost, and cross-entropy classification cost for the given class against "all others", scaled by a factor of 1000, (b) then shorten the networks by tapping off the output of the third layer, with data shape 16×60 (seen as 16 time steps, 60-dimensional feature), (c) train an HMM to estimate the probability distribution $p_m(\mathbf{h}_m)$ of this output map, (d) create a generative classifier using the class-dependent feature approach at the end of Section II-A by multiplying $p_m(\mathbf{h}_m)$ by the corresponding J-function (see beginning of Section II-A). The classification is made by choosing the PBN network with highest log-likelihood, and this is calculated from the log of the combined J-function for the first 3 layers, plus the LF of the HMM. An example of this for a 2-layer network is equation (2), where $g(\mathbf{z})$, is replaced by the HMM distribution. Because the HMM was significantly dependent on the random initialization, we always conducted three trials, and averaged the results. For data augmentation during training, we used random time shifts with a maximum of ±10 time segments.

The number of errors for each of the 4 data partitions are shown in Table II for DNN classifier, PBN-DA (the unshortened PBN networks), PBN-DA-HMM, and CNN classifier. It can be seen that DNN performs worst followed by a better performing PBN-DA. Furthermore, PBN-DA-HMM and CNN have a very similar performance that is significantly better than both other classifiers.

TABLE II
NUMBER OF ERRORS ON EACH OF THE 4 DATA PARTITIONS.

	Partition				
Algorithm	A	B	C	D	mean
DNN	24	24	33	26	26.75
PBN-DA	26	18	18	12	18.5
PBN-DA-HMM	10.7	4.3	14.3	6	8.8
CNN	7.3	9	11.3	8	8.9
two CNNs	5	6.7	9	5.7	6.6
PBN-DA-HMM and CNN[2]	7.7	5	7	2	5.4

It is well-known that combining the output of several models usually improves the classification performance, especially if the combined classifiers (a) have comparable performance, and (b) they are based on different methods or views of the data. This is especially true for CNN and PBN-DA-HMM. The resulting number of errors is shown in Figure 1 as a function of the linear combining factor. It is clearly visible that combining both approaches significantly reduces the number of errors. Moreover, when comparing the results of this combination to the ones obtained when combining two CNNs, which slightly improves performance as shown in Tab. II, the resulting performance is better verifying that both models, CNN and PBN-DA-HMM, indeed must have different views of the data.

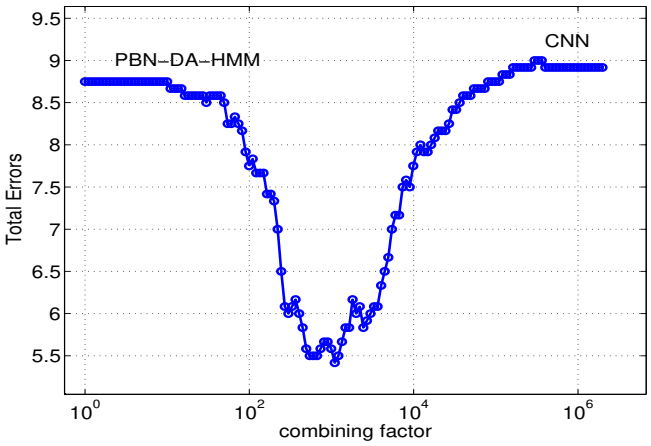

Fig. 1. Mean of errors over all four partitions when combining PBN-DA-HMM with CNN.

D. Reproduceability

Because we used a non-standard subset of ESC50 data, and non-standard data partitions, we make the feature data available, as well as software and instructions to reproduce the results in this paper as [27].

[2]These numbers are obtained using an optimal combination factor with a value of 1100 in this case.

IV. Conclusion and Future Work

In this paper, a new generative classifier has been described. PBNs are trained separately on each class using discriminative alignment, ensuring that the generative models are selective against the other data classes. Then, the PBNs were shortened, tapping off at a convolutional layer, where the time dimension is present. The probability density of these features are estimated using HMMs, which leverage the Markov assumption to create good probability density density estimates. The resulting generative model is shown to rival the performance of a state of the art discriminative classifier. Furthermore, when additively combined, the error rate was cut in half relative to each approach by itself, a verification of the independent nature of the two competing approaches.

For future work, it is planned to extend the results to the full ESC50 data set.

References

[1] X. Zhuang, X. Zhou, T. S. Huang, and M. Hasegawa-Johnson, "Feature analysis and selection for acoustic event detection," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2008, pp. 17–20.

[2] A. Mesaros, T. Heittola, A. J. Eronen, and T. Virtanen, "Acoustic event detection in real life recordings," in *18th European Signal Processing Conference (EUSIPCO)*. IEEE, 2010, pp. 1267–1271.

[3] P. M. Baggenstoss, "Acoustic event classification using multi-resolution HMM," in *26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 972–976.

[4] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, "Detection and classification of acoustic scenes and events: Outcome of the DCASE 2016 challenge," *IEEE ACM Trans. Audio Speech Lang. Process.*, vol. 26, no. 2, pp. 379–393, 2018.

[5] A. Mesaros, T. Heittola, T. Virtanen, and M. D. Plumbley, "Sound event detection: A tutorial," *IEEE Signal Process. Mag.*, vol. 38, no. 5, pp. 67–83, 2021.

[6] R. Yoshihashi, W. Shao, R. Kawakami, S. You, M. Iida, and T. Naemura, "Classification-reconstruction learning for open-set recognition," in *Conference on Computer Vision and Pattern Recognition (CVPR)*. CVF / IEEE, 2019, pp. 4016–4025.

[7] P. Perera, V. I. Morariu, R. Jain, V. Manjunatha, C. Wigington, V. Ordonez, and V. M. Patel, "Generative-discriminative feature representations for open-set recognition," in *Conference on Computer Vision and Pattern Recognition (CVPR)*. CVF / IEEE, 2020, pp. 11 811–11 820.

[8] P. M. Baggenstoss, "A neural network based on first principles," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 4002–4006.

[9] P. M. Baggenstoss and F. Kurth, "Using the projected belief network at high dimensions," in *30th European Signal Processing Conference (EUSIPCO)*. IEEE, 2022, pp. 1526–1530.

[10] P. M. Baggenstoss, "Applications of projected belief networks (PBN)," in *27th European Signal Processing Conference (EUSIPCO)*. IEEE, 2019, pp. 1–5.

[11] ——, "On the duality between belief networks and feed-forward neural networks," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 30, no. 1, pp. 190–200, 2019.

[12] I. Kobyzev, S. J. D. Prince, and M. A. Brubaker, "Normalizing flows: An introduction and review of current methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 11, pp. 3964–3979, 2021.

[13] P. M. Baggenstoss, "Discriminative alignment of projected belief networks," *IEEE Signal Process. Lett.*, vol. 28, pp. 1963–1967, 2021.

[14] J. Ebbers, J. Heymann, L. Drude, T. Glarner, R. Haeb-Umbach, and B. Raj, "Hidden markov model variational autoencoder for acoustic unit discovery," in *18th Annual Conference of the International Speech Communication Association (Interspeech)*. ISCA, 2017, pp. 488–492.

[15] T. Glarner, P. Hanebrink, J. Ebbers, and R. Haeb-Umbach, "Full bayesian hidden markov model variational autoencoder for acoustic unit discovery," in *19th Annual Conference of the International Speech Communication Association (Interspeech)*. ISCA, 2018, pp. 2688–2692.

[16] J. Santoso, T. Yamada, K. Ishizuka, T. Hashimoto, and S. Makino, "Performance improvement of speech emotion recognition by neutral speech detection using autoencoder and intermediate representation," in *23rd Annual Conference of the International Speech Communication Association (Interspeech)*. ISCA, 2022, pp. 4700–4704.

[17] M. Meyer, J. Beutel, and L. Thiele, "Unsupervised feature learning for audio analysis," in *5th International Conference on Learning Representations (ICLR)*. OpenReview.net, 2017.

[18] P. M. Baggenstoss, "The PDF projection theorem and the class-specific method," *IEEE Trans. Signal Process.*, vol. 51, no. 3, pp. 672–685, 2003.

[19] ——, "Beyond moments: Extending the maximum entropy principle to feature distribution constraints," *Entropy*, vol. 20, no. 9, p. 650, 2018.

[20] ——, "Maximum entropy PDF design using feature density constraints: Applications in signal processing," *IEEE Trans. Signal Process.*, vol. 63, no. 11, pp. 2815–2825, 2015.

[21] P. M. Baggenstoss and S. Kay, "Nonlinear dimension reduction by PDF estimation," *IEEE Trans. Signal Process.*, vol. 70, pp. 1493–1505, 2022.

[22] E. T. Jaynes, "Information theory and statistical mechanics i," *Physical Review*, p. 171–190, 1957.

[23] V. Vapnik, *The Nature of Statistical Learning*. Springer, 1999.

[24] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning*. Cambridge, MA: MIT press, 2016.

[25] K. J. Piczak, "ESC: Dataset for environmental sound classification," in *Proceedings of the 23rd Annual ACM Conference on Multimedia Conference, Brisbane, Australia*. ACM, 2015, pp. 1015–1018.

[26] P. M. Baggenstoss, "Uniform manifold sampling (UMS): sampling the maximum entropy PDF," *IEEE Trans. Signal Process.*, vol. 65, no. 9, pp. 2455–2470, 2017.

[27] P. Baggenstoss, "PBN Toolkit," {*http://class-specific.com/pbntk*}, accessed: 2022-02-28.

[28] P. M. Baggenstoss, "New restricted boltzmann machines and deep belief networks for audio classification," in *14th ITG Conference on Speech Communication (ITG Speech)*. VDE, 2021, pp. 1–5.

[29] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *30th International Conference on Machine Learning (ICML)*, 2013.

[30] K. Wilkinghoff, "Design choices for learning embeddings from auxiliary tasks for domain generalization in anomalous sound detection," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023.

[31] K. Wilkinghoff and F. Fritz, "On using pre-trained embeddings for detecting anomalous sounds with limited training data," in *31st European Signal Processing Conference (EUSIPCO)*. IEEE, 2023.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Conference on Computer Vision and Pattern Recognition, CVPR*. IEEE, 2016, pp. 770–778.

[33] K. Wilkinghoff, "Sub-cluster AdaCos: Learning representations for anomalous sound detection," in *International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021.

[34] X. Zhang, R. Zhao, Y. Qiao, X. Wang, and H. Li, "AdaCos: Adaptively scaling cosine logits for effectively learning deep face representations," in *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019, pp. 10 823–10 832.

[35] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *CoRR*, vol. abs/1207.0580, 2012.

[36] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," in *6th International Conference on Learning Representations (ICLR)*, 2018.

[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations (ICLR)*, 2015.

[38] M. Abadi *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2016, pp. 265–283.