A Review of Unsupervised Feature Learning and Deep Learning for Time-Series Modeling

Article *in* Pattern Recognition Letters · June 2014

DOI: 10.1016/j.patrec.2014.01.008

3 authors:


Martin Johan Längkvist
Örebro University


Lars Karlsson
Örebro University


Amy Loutfi
Örebro University

A Review of Unsupervised Feature Learning and Deep Learning for Time-Series Modeling

Martin Längkvist[a,*], Lars Karlsson[a], Amy Loutfi[a]

[a] *Applied Autonomous Sensor Systems, School of Science and Technology, Örebro University, SE-701 82, Örebro, Sweden*

Abstract

This paper gives a review of the recent developments in deep learning and unsupervised feature learning for time-series problems. While these techniques have shown promise for modeling static data, such as computer vision, applying them to time-series data is gaining increasing attention. This paper overviews the particular challenges present in time-series data and provides a review of the works that have either applied time-series data to unsupervised feature learning algorithms or alternatively have contributed to modifications of feature learning algorithms to take into account the challenges present in time-series data.

Keywords: time-series, unsupervised feature learning, deep learning

---

1. Introduction and Background

Time is a natural element that is always present when the human brain is learning tasks like language, vision and motion. Most real-world data has a temporal component, whether it is measurements of natural processes

*Corresponding author

Email addresses: martin.langkvist@oru.se (Martin Längkvist), lars.karlsson@oru.se (Lars Karlsson), amy.loutfi@oru.se (Amy Loutfi)

Preprint submitted to Pattern Recognition Letters *February 3, 2014*

(weather, sound waves) or man-made (stock market, robotics). Analysis of time-series data has been the subject of active research for decades (Keogh and Kasetty, 2002; Dietterich, 2002) and is considered by Yang and Wu (2006) as one of the top 10 challenging problems in data mining due to its unique properties. Traditional approaches for modeling sequential data include the estimation of parameters from an assumed time-series model, such as autoregressive models (Lütkepohl, 2005) and Linear Dynamical Systems (LDS) (Luenberger, 1979), and the popular Hidden Markov Model (HMM) (Rabiner and Juang, 1986). The estimated parameters can then be used as features in a classifier to perform classification. However, more complex, high-dimensional, and noisy real-world time-series data cannot be described with analytical equations with parameters to solve since the dynamics are either too complex or unknown (Taylor, 2009) and traditional shallow methods, which contain only a small number of non-linear operations, do not have the capacity to accurately model such complex data.

In order to better model complex real-world data, one approach is to develop robust features that capture the relevant information. However, developing domain-specific features for each task is expensive, time-consuming, and requires expertise of the data. The alternative is to use unsupervised feature learning (Bengio and LeCun, 2007; Bengio et al., 2012; Erhan et al., 2010) in order to learn a layer of feature representations from unlabeled data. This has the advantage that the unlabeled data, which is plentiful and easy to obtain, is utilized and that the features are learned from the data instead of being hand-crafted. Another benefit is that these layers of feature representations can be stacked to create deep networks, which are more capable of modeling complex structures in the data. Deep networks have been used to achieve state-of-the-art results on a number of benchmark data sets and for solving difficult AI tasks. However, much focus in the feature learning community has been on developing models for static data and not so much on time-series data.

In this paper we review the variety of feature learning algorithms that has been developed to explicitly capture temporal relationships as well as the various time-series problems that they have been used on. The properties of time-series data will be discussed in Section 2 followed by an introduction to unsupervised feature learning and deep learning in Section 3. An overview of some common time-series problems and previous work using deep learning is given in Section 4. Finally, conclusions are given in Section 5.

2. Properties of time-series data

Time-series data consists of sampled data points taken from a continuous, real-valued process over time. There are a number of characteristics of time-series data that make it different from other types of data.

Firstly, the sampled time-series data often contain much noise and have high dimensionality. To deal with this, signal processing techniques such as dimensionality reduction techniques, wavelet analysis or filtering can be applied to remove some of the noise and reduce the dimensionality. The use of feature extraction has a number of advantages (Nanopoulos et al., 2001). However, valuable information could be lost and the choice of features and signal processing techniques may require expertise of the data.

The second characteristics of time-series data is that it is not certain that there are enough information available to understand the process. For example, in electronic nose data, where an array of sensors with various selectivity for a number of gases are combined to identify a particular smell, there is no guarantee that the selection of sensors actually are able to identify the target odour. In financial data when observing a single stock, which only measures a small aspect of a complex system, there is most likely not enough information in order to predict the future (Fama, 1965).

Further, time-series have an explicit dependency on the time variable. Given an input $x(t)$ at time t, the model predicts $y(t)$, but an identical input at a later time could be associated with a different prediction. To solve this problem, the model either has to include more data input from the past or must have a memory of past inputs. For long-term dependencies the first approach could make the input size too large for the model to handle. Another challenge is that the length of the time-dependencies could be unknown.

Many time-series are also non-stationary, meaning that the characteristics of the data, such as mean, variance, and frequency, changes over time. For some time-series data, the change in frequency is so relevant to the task that it is more beneficial to work in the frequency-domain than in the time-domain.

Finally, there is a difference between time-series data and other types of data when it comes to invariance. In other domains, for example computer vision, it is important to have features that are invariant to translations, rotations, and scale. Most features used for time-series need to be invariant to translations in time.

In conclusion, time-series data is high-dimensional and complex with unique properties that make them challenging to analyze and model. There is a large interest in representing the time-series data in order to reduce the dimensionality and extract relevant information. The key for any successful application lies in choosing the right representation. Various time-series problems contain different degrees of the properties discussed in this section and prior knowledge or assumptions about these properties is often infused in the chosen model or feature representation. There is an increasing interest in learning the representation from unlabeled data instead of using hand-designed features. Unsupervised feature learning have shown to be successful at learning layers of feature representations for static data sets and can be combined with deep networks to create more powerful learning models. However, the feature learning for time-series data have to be modified in order to adjust for the characteristics of time-series data in order to capture the temporal information as well.

3. Unsupervised feature learning and deep learning

This section presents both models that are used for unsupervised feature learning and models and techniques that are used for modeling temporal relations. The advantage of learning features from unlabeled data is that the plentiful unlabeled data can be utilized and that potentially better features than hand-crafted features can be learned. Both these advantages reduce the need for expertise of the data.

3.1. Restricted Boltzmann Machine

The Restricted Boltzmann Machines (RBM) (Hinton et al., 2006; Hinton and Salakhutdinov, 2006; Lee et al., 2008) is a generative probabilistic model between input units (visible), $\mathbf{x}$, and latent units (hidden), $\mathbf{h}$, see Figure 1.

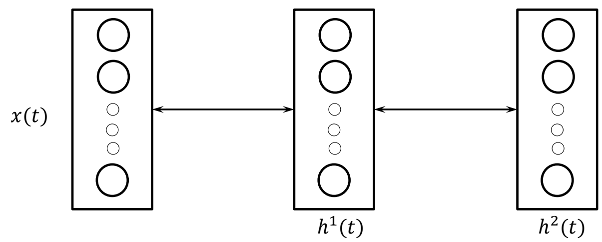

Figure 1: A 2-layer RBM for static data. The visible units x are fully connected to the first hidden layer h^1.

The visible and hidden units are connected with a weight matrix, $\mathbf{W}$ and have bias vectors $\mathbf{c}$ and $\mathbf{b}$, respectively. There are no connections among the visible and hidden units. The RBM can be used to model static data. The energy function and the joint distribution for a given visible and hidden vector is defined as:

$$E(\mathbf{x}, \mathbf{h}) = \mathbf{h}^T \mathbf{W} \mathbf{x} + \mathbf{b}^T \mathbf{h} + \mathbf{c}^T \mathbf{v} \tag{1}$$

$$P(\mathbf{x}, \mathbf{h}) = \frac{1}{Z} \exp^{E(\mathbf{x}, \mathbf{h})} \tag{2}$$

where Z is the partition function that ensures that the distribution is normalized. For binary visible and hidden units, the probability that hidden unit h_j is activated given visible vector x and the probability that visible unit x_i is activated given hidden vector h are given by:

$$P(h_j|\mathbf{x}) = \sigma(b_j + \sum_i W_{ij} x_i) \tag{3}$$

$$P(x_i|\mathbf{h}) = \sigma(c_i + \sum_j W_{ij} h_j) \tag{4}$$

where $\sigma(\cdot)$ is the activation function. The logistic function, $\sigma(x) = \frac{1}{1+e^{-x}}$, is a common choice for the activation function. The parameters W, b, and v, are trained to minimize the reconstruction error using contrastive divergence (Hinton, 2002). The learning rule for the RBM is:

$$\frac{\partial \log P(\mathbf{x})}{\partial W_{ij}} \approx \langle x_i h_j \rangle_{data} - \langle x_i h_j \rangle_{recon} \tag{5}$$

where $\langle \cdot \rangle$ is the average value over all training samples. Several RBMs can be stacked to produce a deep belief network (DBN). In a deep network, the activation of the hidden units in the first layer is the input to the second layer.

3.2. Conditional RBM

An extension of RBM that models multivariate time-series data is the conditional RBM (cRBM), see Figure 2. A similar model is the Temporal RBM (Sutskever and Hinton, 2006). The cRBM consists of auto-regressive weights that model short-term temporal structures, and connections between

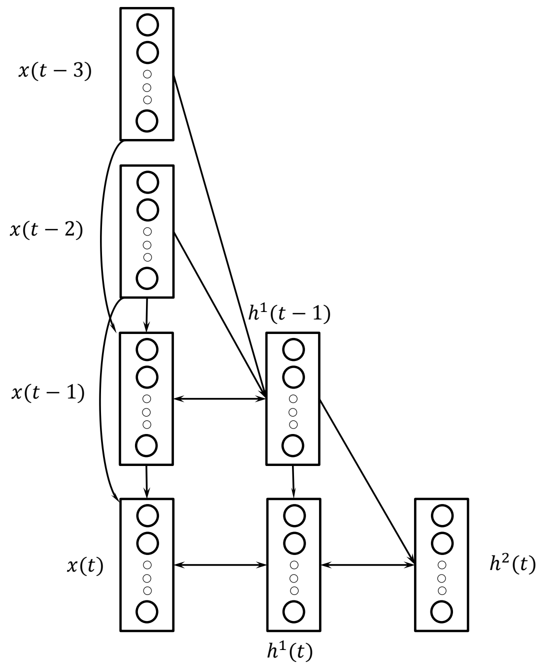

Figure 2: A 2-layer conditional RBM for time-series data. The model order for the first and second layer is 3 and 2, respectively.

past visible units to the current hidden units. The bias vectors in a cRBM depend on previous visible units and are defined as:

$$b_j^* = b_j + \sum_{i=1}^{n} B_i x(t-i) \tag{6}$$

$$c_i^* = c_j + \sum_{i=1}^{n} A_i x(t-i) \tag{7}$$

where A_i is the auto-regressive connections between visible units at time $t-i$ and current visible units, B_i is the weight matrix connecting visible layer at time $t-i$ to the current hidden units. The model order is defined by the constant n. The probabilities for going up or down a layer are:

$$P(h_j|\mathbf{x}) = \sigma\left(b_j + \sum_i W_{ij}x_i + \sum_k \sum_i B_{ijk}x_i(t-k)\right) \tag{8}$$

$$P(x_i|\mathbf{h}) = \sigma\left(c_i + \sum_j W_{ij}h_j + \sum_k \sum_i A_{ijk}x_i(t-k)\right) \tag{9}$$

The parameters $\theta = \{W, b, c, A, B\}$, are trained using contrastive divergence. Just like a RBM, the cRBM can also be used as a module to create deep networks.

3.3. Gated RBM

The Gated Restricted Boltzmann Machine (GRBM) (Memisevic and Hinton, 2007) is another extension of the RBM that models the transition between two input vectors. The GRBM models a weight tensor, W_{ijk}, between the input, $\mathbf{x}$, the output, $\mathbf{y}$, and latent variables, $\mathbf{z}$. The energy function is defined as:

$$E(\mathbf{y}, \mathbf{z}; \mathbf{x}) = -\sum_{ijk} W_{ijk}x_i y_j z_k - \sum_k b_k z_k - \sum_j c_j y_j \tag{10}$$

where $\mathbf{b}$ and $\mathbf{c}$ are the bias vectors for $\mathbf{x}$ and $\mathbf{y}$, respectively. The conditional probability of the transformation and the output image given the input image is:

$$p(\mathbf{y}, \mathbf{z}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})}\exp(-E(\mathbf{y}, \mathbf{z}; \mathbf{x})) \tag{11}$$

where $Z(\mathbf{x})$ is the partition function. Luckily, this quantity does not need to be computed to perform inference or learning. The probability that hidden unit z_i is activated given $\mathbf{x}$ and $\mathbf{y}$ is given by:

$$P(z_k = 1|\mathbf{x}, \mathbf{y}) = \sigma(\sum_{ij} W_{ijk} x_i y_j + b_k) \tag{12}$$

Learning the parameters is performed with an approximation method of the gradient called contrastive divergence (Hinton, 2002). Each latent variable z_k learns a simple transformation that together are combined the represent the full transformation. By fixating a learned transformation $\mathbf{z}$ and given an input image $\mathbf{x}$, the output image $\mathbf{y}$ is the selected transformation applied to the input image. Similarly, for a fixed input image $\mathbf{x}$, a given image $\mathbf{y}$ creates a RBM that learns the transformation $\mathbf{z}$ by reconstructing $\mathbf{y}$. These properties could not be achieved with a regular RBM with input units simply being the concatenated images $\mathbf{x}$ and $\mathbf{y}$ since the latent variables would only learn the spatial information for that particular image pair and not the general transformation. The large number of parameters due to the weight tensor makes it impractical for large image sizes. A factored form of the three-way tensor has been proposed to reduce the number of parameters to learn (Memisevic and Hinton, 2010).

3.4. Auto-encoder

A model that does not have a partition function is the auto-encoder (Ranzato et al., 2006; Bengio et al., 2007; Bengio, 2007), see Figure 3. The auto-encoder was first introduced as a dimensionality reduction algorithm. In fact, a basic linear auto-encoder learns essentially the same representation as

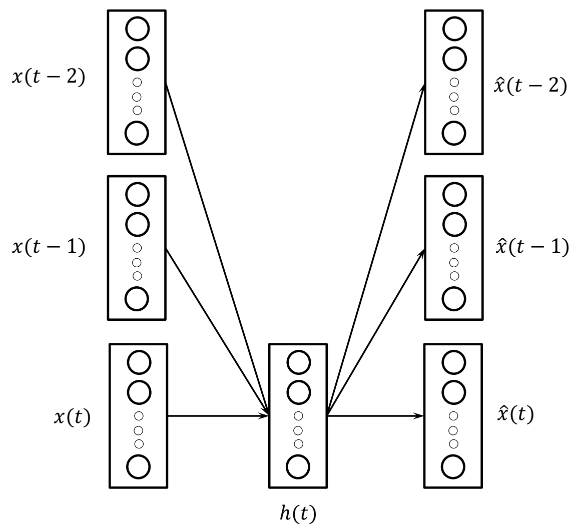

Figure 3: A 1-layer auto-encoder for static time-series input. The input is the concatenation of current and past frames of visible data x. The reconstruction of x is denoted $\hat{x}$.

a Principal Component Analysis (PCA). The layers of visible units, $\mathbf{x}$, hidden units, $\mathbf{h}$, and the reconstruction of the visible units, $\hat{\mathbf{x}}$, are connected via weight matrices $\mathbf{W}^1$ and $\mathbf{W}^2$ and the hidden layer and reconstruction layer have bias vectors $\mathbf{b}^1$ and $\mathbf{b}^2$, respectively. It is common in auto-encoders to have tied weights, that is, $\mathbf{W}^2 = (\mathbf{W}^1)^T$. This works as a regularizer as it constrains the allowed parameter space and reduces the number of parameters to learn (Bengio et al., 2012). The feed-forward activations are calculated as:

$$h_j = \sigma(\sum_i W_{ji}^1 x_i + b_j^1) \tag{13}$$

$$\hat{x}_i = \sigma(\sum_j W_{ij}^2 h_j + b_i^2) \tag{14}$$

where $\sigma(\cdot)$ is the activation function. As with the RBM, a common choice is the logistic activation function. The cost function to be minimized is expressed as:

$$J(\theta) = \frac{1}{2N}\sum_n^N \sum_i (x_i^{(n)} - \hat{x}_i{}^{(n)})^2 + \frac{\lambda}{2}\sum_l \sum_i \sum_j (W_{ij}^l)^2 + \beta \sum_l \sum_j KL(\rho || p_j^l) \tag{15}$$

where p_j^l is the mean activation for unit j in layer l, ρ is the desired mean activation, and N is the number of training examples. KL is the Kullback-Leibler (KL) divergence which is defined as $KL(\rho||p_j^l) = \rho \log \frac{\rho}{p_j^l} + (1 - \rho)\log \frac{1-\rho}{1-p_j^l}$. The first term is the square root error term that will minimize the reconstruction error. The second term is the L2 weight decay term that will keep the weight matrices close to zero. Finally, the third term is the sparsity penalty term and encourages each unit to only be partially activated as specified by the hyperparameter ρ. The inclusion of these regularization terms prevents the trivial learning of a 1-to-1 mapping of the input to the hidden units. A difference between auto-encoders and RBMs is that RBMs do not require such regularization because the use of stochastic binary hidden units acts as a very strong regularizer (Hinton, 2012). However, it is not uncommon to introduce an extra sparsity constraint for RBMs (Lee et al., 2008).

3.5. Recurrent Neural Network

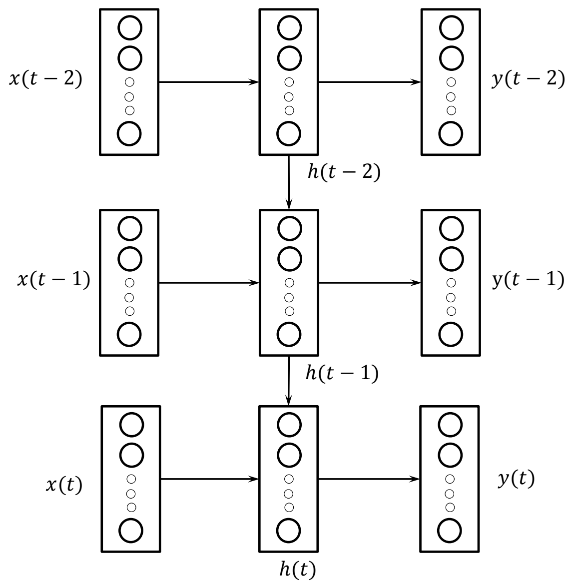

Figure 4: A Recurrent Neural Network (RNN). The input x is transformed to the output representation y via the hidden units h. The hidden units have connections from the input values of the current time frame and the hidden units from the previous time frame.

A model that have been used for modeling sequential data is the Recurrent Neural Network (RNN) (Hüsken and Stagge, 2003). Generally, an RNN is obtained from the feedforward network by connecting the neurons' output to their inputs, see Figure 4. The short-term time-dependency is modeled by the hidden-to-hidden connections without using any time delay-taps. They are usually trained iteratively via a procedure known as backpropagation-through-time (BPTT). RNNs can be seen as very deep networks with shared parameters at each layer when unfolded in time. This results in the problem of vanishing gradients (Pascanu et al., 2012) and has motivated the exploration of second-order methods for deep architectures (Martens and Sutskever, 2012) and unsupervised pre-training. An overview of strategies for training RNNs is provided by Sutskever (2012). A popular extension is the use of the purpose-built Long-short term memory cell (Hochreiter and Schmidhuber, 1997) that better finds long-term dependencies.

3.6. Deep learning

The models presented in this section use a non-linear activation function on the hidden units. This non-linearity enables a more expressive model that can learn more abstract representations when multiple modules are stacked on top of each other to form a deep network (if linear features would be stacked the result would still be a linear operation). The goal of a deep network is to build features at the lower layers that will disentangle the factors of variations in the input data and then combine these representations at the higher layers. It has been proposed that a deep network will generalize better because it has a more compact representation (Le Roux and Bengio, 2008). However, the difficulty with training multiple layers of hidden units lies in the problem of vanishing gradients when the error signal is backpropagated (Bengio et al., 1994). This can be solved by doing unsupervised greedy layer-wise pre-training of each layer. This acts as an unusual form of regularization (Erhan et al., 2010) that avoids poor local minima and gives a better initialization than a random initialization (Bengio et al., 2012). However, the importance of parameter initialization is not as crucial as other factors such as input connections and architecture (Saxe et al., 2011).

3.7. Convolution and pooling

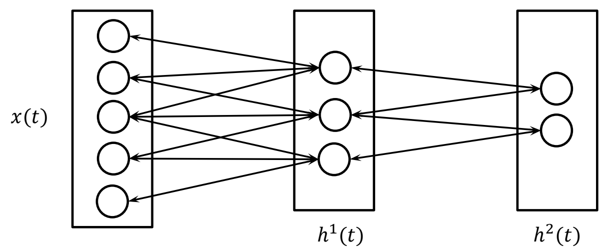

Figure 5: A 2-layer convolutional neural network.

A technique that is particularly interesting for high-dimensional data, such as images and time-series data, is convolution. In a convolutional setting, the hidden units are not fully connected to the input but instead divided into locally connected segments, see Figure 5. Convolution has been applied to both RBMs and auto-encoders to create convolutional RBMs (convRBM) (Lee et al., 2009b,a) and convolutional auto-encoders (convAE) (Masci et al., 2011). A Time-Delay Neural Network (TDNN) is a specialization of Artificial Neural Networks (ANN) that exploits the time structure of the input by performing convolutions on overlapping windows.

A common operator used together with convolution is pooling, which combines nearby values in input or feature space through a max, average or histogram operator. The purpose of pooling is to achieve invariance to small local distortions and reduce the dimensionality of the feature space. The work by Lee et al. (2009a) introduces probabilistic max-pooling in the context of convolutional RBMs. The Space-Time DBN (ST-DBN) (Bo Chen and de Freitas, 2010) uses convolutional RBMs together with a spatial pooling layer and a temporal pooling layer to build invariant features from spatio-temporal data.

3.8. Temporal coherence

There are a number of other ways besides the architectural structure that can be used to capture temporal coherence in data. One way is to introduce a smoothness penalty on the hidden variables in the regularization. This is done by minimizing the changes in the hidden unit activations from one frame to the next by $\min |h(t) - h(t-1)|$. The motivation behind this is that for sequential data the hidden unit activations should not change much if the time-dependent data is fed to the model in a chronological order. Other strategies include penalizing the squared difference, slow feature analysis (Wiskott and Sejnowski, 2002), or as a function of other factors, for example the change in the input data in order to adapt to both slow and rapid changing input data.

Temporal coherence is related to invariant feature representations since both methods want to achieve small changes in the feature representation for small changes in the input data. It is suggested in Hinton et al. (2011) that the pose parameters and affine transformations should be modeled instead of using invariant feature representations. In that case, temporal coherence should be over a group of numbers, such as the position and pose of the object rather than a single scalar. This could for example be achieved using a structured sparsity penalty (Kavukcuoglu et al., 2009).

3.9. Hidden Markov Model

The Hidden Markov Model (HMM) (Rabiner and Juang, 1986) is a popular model for modeling sequential data and is defined by two probability distributions. The first one is the transition distribution $P(y_t|y_{t-1})$, which defines the probability of going from one hidden state y to the next hidden state. The second one is the observation distribution $P(x_t|y_t)$, which defines the relation between observed x values and hidden y states. One assumption is that these distributions are stationary. However, the main problem with HMMs are that they require a discrete state space, often have unrealistic independence assumptions, and have a limited representational capacity of their hidden states (Mohamed and Hinton, 2010). HMMs require 2^N hidden states in order to model N bits of information about the past history.

3.10. Summary

Table 1 gives a summary of the briefly presented models in this section. The first column indicates whether the model is capable of capturing temporal relations. A model that captures temporal relations does so by having a memory of past inputs. The memory of a model, indicated in the second column, means how many steps back in time an input have on the current frame. Without the temporal order, any permutation of the feature sequence would yield the same distribution (Humphrey et al., 2013). The implementation of a memory is performed differently between the models. In a cRBM, delay taps are used to create a short-term dependency on past visible units. The long-term dependency comes from modeling subsequent layers. This means that the length of the memory for a cRBM is increased for each added layer. The model order for a cRBM in one layer is typically below 5 for input sizes around 50. A decrease in the input size would allow a higher model order. In an RNN, hidden units in the current time frame are affected by the state of the hidden units in the previous time frame. This can create a ripple effect with a duration of potentially infinite time frames. On the other hand, this ripple effect can be prevented by using a forget gate (Gers et al., 2000). The use of Long-short term memory (Hochreiter and Schmidhuber, 1997) or hessian-free optimizer (Martens and Sutskever, 2012) can produce recurrent networks that has a memory of over 100 time steps. The Gated RBM and the convolutional GRBM models transitions between pairs of input vectors so the memory for these models is 2. The Space-Time DBN (Bo Chen and de Freitas, 2010) models 6 sequences of outputs from the spatial pooling layer, which is a longer memory than GRBM, but using a lower input size.

The last column in Table 1 indicates if the model is generative (as opposed to discriminative). A generative model can generate observable data given a hidden representation and this ability is mostly used for generating synthetic data of future time steps. Even though the auto-encoder is not generative, a probabilistic interpretation can be made using auto-encoder scoring (Kamyshanska and Memisevic, 2013; Bengio et al., 2013).

For selecting a model for a particular problem, a number of questions

Table 1: A summary of commonly used models for feature learning.

Method	Temporal relation	Memory	Typical input size	Generative
RBM	-	-	10-1000	✓
AE	-	-	10-1000	-
RNN	✓	1-100	50-1000	✓
cRBM	✓	2-5	50	✓
TDNN	✓	2-5	5-50	-
ANN	-	-	10-1000	-
GRBM	✓	2	<64x64	✓
ConvGRBM	✓	2	>64x64	✓
ConvRBM	-	-	>64x64	✓
ConvAE	-	-	>64x64	-
ST-DBN	✓	2-6	10x10	✓

should be taken into consideration: (1) Use a generative or discriminative model? (2) What are the properties of the data? and (3) How large is the input size? A generative model is preferred if the trained model should be used for synthesizing new data or prediction tasks where partial input data (data at $t + 1$) need to be reconstructed. If the task is to do classification, a discriminative model is sufficient. A discriminative model will attempt to model the training data even if that data is noisy while a generative model will simply assign a low probability for outliers. This makes a generative model more robust for noisy inputs and a better outlier detector. There is also the factor of training time. Generative models use Gibbs sampling to approximate the derivatives for each parameter update while a discriminative model calculates the exact gradients in one iteration. However, if the simulation time is an issue, it is a good idea to look for hardware solutions or the choice of optimization method before considering which method is the fastest. When the combination of input size, model parameters, and number of training examples in one training batch is large, the training time could be decreased by performing the parameter updates on a GPU instead of the CPU. For large-scale problems, i.e., the number of training examples is large, it is recommended to use stochastic gradient descent instead of L-BFGS or conjugate gradient descent as optimization method (Bottou, 2010). Furthermore, if the data has a temporal structure it is not recommended to treat the input data as a feature vector since this will discard the temporal information. Instead, a model that inherently models temporal relations or incorporates temporal coherence (by regularization or temporal pooling) in a static model is a better approach. For high-dimensional problems, like images which have a pictorial structure, it may be appropriate to use convolution. The use of pooling further decreases the number of dimensions and introduces invariance for small translations of the input data.

4. Classical time-series problems

In this section we will highlight some common time-series problems and the models that have been used to address them in the literature. We will focus on complex problems that require the use of models with hidden variables for feature representation and where the representations are fully or partially learned from unlabeled data. A summary of the classical time-series problems that will be presented in this section is given in Table 2.

Figure 6: Four images from the KTH action recognition data set of a person running at frame 100, 105, 110, and 115. The KTH data set also contains videos of walking, jogging, boxing, hand waving, and handclapping.

4.1. Videos

Video data are series of images over time (spatio-temporal data) and can therefore be viewed as high-dimensional time-series data. Figure 6 shows a sequence of images from the KTH activity recognition data set[1]. The traditional approach to modeling video streams is to treat each individual static image and detecting interesting points using common feature detectors such as SIFT (Lowe, 1999) or HOG (Dalal and Triggs, 2005). These features are domain-specific for static images and are not easily extended to other domains such as video (Le et al., 2011).

The approach taken by Stavens and Thrun (2010) learns its own domain-optimized features instead of using pre-defined features, but still from static images. A better approach to modeling videos is to learn image transitions instead of working with static images. A Gated Restricted Boltzmann Machine (GRBM) (Memisevic and Hinton, 2007) has been used for this purpose where the input, x, of the GRBM is the full image in one time frame and the output y is the full image in the subsequent time frame. However, since the network is fully connected to the image the method does not scale well

[1] `http://www.nada.kth.se/cvap/actions/`

to larger images and local transformations at multiple locations must be re-learned.

A convolutional version of the GRBM using probabilistic max-pooling is presented by Taylor et al. (2010). The use of convolution reduces the number of parameters to learn, allows for larger input sizes, and better handles the local affine transformations that can appear anywhere in the image. The model was validated on synthetic data and a number of benchmark data sets, including the KTH activity recognition data set.

The work by Le et al. (2011) presents an unsupervised spatio-temporal feature learning method using an extension of Independent Subspace Analysis (ISA) (Hyvèarinen et al., 2009). The extensions include hierarchical (stacked) convolutional ISA modules together with pooling. A disadvantage of ISA is that it does not scale well to large input sizes. The inclusion of convolution and stacking solves this problem by learning on smaller patches of input data. The method is validated on a number of benchmark sets, including KTH. One advantage of the method is that the use of ISA reduces the need for tweaking many of the hyperparameters seen in RBM-based methods, such as learning rate, weight decay, convergence parameters, etc.

Modeling temporal relations in video have also been done using temporal pooling. The work by Bo Chen and de Freitas (2010) uses convolutional RBMs as building blocks for spatial pooling and then performs temporal pooling on the spatial pooling units. The method is called Space-Time Deep Belief Network (ST-DBN). The ST-DBN allows for invariance and statistical dependencies in both space and time. The method achieved superior performance on applications such as action recognition and video denoising when compared to a standard convolutional DBN.

The use of temporal coherence for modeling videos is done by Zou et al. (2011), where an auto-encoder with a L1-cost on the temporal difference on the pooling units is used to learn features that improve object recognition on still images. The work by Hyvärinen et al. (2003) also uses temporal information as a criterion for learning representations.

The use of deep learning, feature learning, and convolution with pooling has propelled the advances in video processing. Modeling streams of video is a natural continuation for deep learning algorithms since they have already been shown to be successful at building useful features from static images. By focusing on learning temporal features in videos, the performance on static images can be improved, which motivates the need for continuing developing deep learning algorithms that capture temporal relations. The early attempts at extending deep learning algorithms to video data was done by modeling the transition between two frames. The use of temporal pooling extends the time-dependencies a model can learn beyond a single frame transition. However, the time-dependency that has been modeled is still just a few frames. A possible future direction for video processing is to look at models that can learn longer time-dependencies.

4.2. Stock market prediction

Stock market data are highly complex and difficult to predict, even for human experts, due to a number of external factors, e.g., politics, global economy, and trader expectation. The trends in stock market data tend to be nonlinear, uncertain, and non-stationary. Figure 7 shows the Dow Jones Industrial Average (DJOI) over a decade. According to the Efficient Market

Figure 7: Dow Jones Industrial Average (DJOI) over a period of 10 years.

Hypothesis (EMH) (Fama, 1965), stock market prices follow a random walk pattern, meaning that a stock has the same probability to go up as it has to go down, resulting in that predictions can not have more than 50% accuracy (Tsai and Hsiao, 2010). The EMH state that stock prices are largely driven by "news" rather than present and past prices. However, it has also been argued that stock market prices do not follow a random walk and that they can be predicted (Malkiel, 2003). The landscape for acquiring both news and stock information looks very different today than it did decades ago. As an example, it has been shown that predicted stock prices can be improved if further information is extracted from online social media, such as Twitter feeds (Bollen et al., 2011) and online chat activity (Gruhl et al., 2005).

One model that has emerged and shown to be suitable for stock market prediction is the artificial neural network (ANN) (Atsalakis and Valavanis, 2009). This is due to its ability to handle non-linear complex systems. A survey of ANNs applied to stock market prediction is given in Li and Ma (2010). However, most approaches of ANN applied to stock prediction have given unsatisfactory results (Agrawal et al., 2013). Neural networks with feedback have also been tried, such as recurrent versions of TDNN (Kim, 1998), wavelet transformed features with an RNN (Hsieh et al., 2011), and echo state networks (Lin et al., 2009). Many of these methods are applied directly on the raw data, while other papers focus more on the feature selection step (Tsai and Hsiao, 2010).

In summary, it can be concluded that there is still room to improve existing techniques for making safe and accurate stock prediction systems. If additional information from sources that affect the stock market can be measured and obtained, such as general public opinions from social media (Bollen et al., 2011), trading volume (Zhu et al., 2008), market specific domain knowledge, and political and economical factors, it can be combined together with the stock price data to achieve higher stock price predictions (Agrawal et al., 2013). The limited success of applying small, one layer neural networks for stock market prediction and the realization that there is a need to add more information to make better predictions indicate that a future direction for stock market prediction is to apply the combined data to more powerful models that are able to handle such complex, high-dimensional data. Deep learning methods for multivariate time-series fit this description and provide new interesting approach for the financial field and a new challenging application for the deep learning community, which to the authors knowledge has not yet been tried.

4.3. Speech recognition

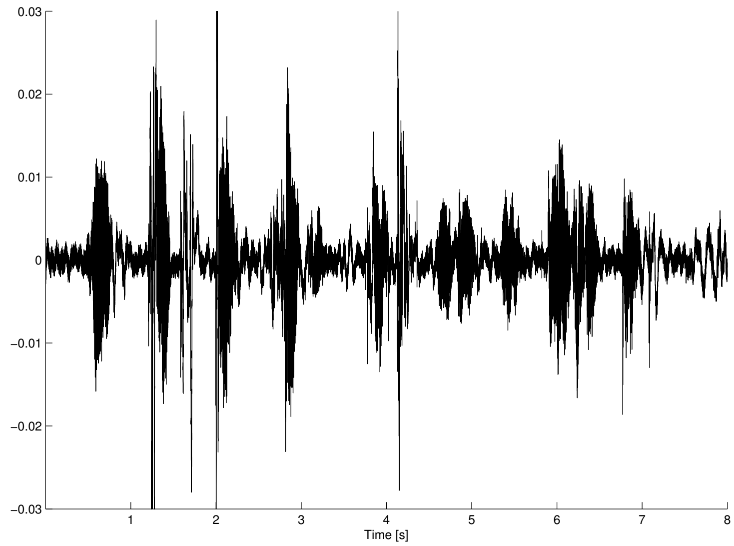

Figure 8: Raw acoustic signal of the utterance of the sentence "The quick brown fox jumps over the lazy dog".

Speech recognition is one area where deep learning has made significant progress (Hinton et al., 2012). The problem of speech recognition can be divided into a variety of sub-problems, such as speaker identification (Lee et al., 2009a), gender identification (Lee et al., 2009b; Parris and Carey, 1996), speech-to-text (Furui et al., 2004) and acoustic modeling. The raw input data is single channel and highly time and frequency dependent, see Figure 8. A common approach is to use pre-set features that are designed for speech processing such as Mel-frequency cepstral coefficients (MFCC).

For decades, Hidden Markov Models (HMMs) (Rabiner and Juang, 1986) have been the state-of-the-art technique for speech recognition. A common method for discretization of the input data for speech that is required by the HMM is to use Gaussian mixture models (GMM). More recently however, the Restricted Boltzmann Machines (RBM) have shown to be an adequate alternative for replacing the GMM in the discretization step. A classification error of 20.7% on the TIMIT speech recognition data set[2] was achieved by (Mohamed et al., 2012) by training a RBM on MFCC features. A similar setup has been used for large vocabulary speech recognition by Dahl et al. (2012). A convolutional deep belief networks was applied by Lee et al. (2009b) to audio data and evaluated on various audio classification tasks.

A number of variations on the RBM have also been tried on speech data. The mean-covariance RBM (mcRBM) (Ranzato and Hinton, 2010; Ranzato et al., 2010) achieved a classification error of 20.5% on the TIMIT data set by Dahl et al. (2010). A conditional RBM (cRBM) was modified by Mohamed and Hinton (2010) by including connections from future instead of only having connections from the past, which presumably gave better classification because the near future is more relevant than the more distant past.

Earlier, a Time-Delay Neural Network (TDNN) has been used for speech recognition (Waibel et al., 1989) and a review of TDNN architectures for speech recognition is given by Sugiyama et al. (1991). However, it has been suggested that convolution over the frequency instead of the time is better since the HMM on top models the temporal information.

The recent work by Graves et al. (2013) uses a deep Long Short-term Memory Recurrent Neural Network (RNN) (Hochreiter and Schmidhuber,

[2] `http://www.ldc.upenn.edu/Catalog/`

1997) to achieve a classification error of 17.7% on the TIMIT data set, which is the best result to date. One difference between the approaches of RBM-HMM and RNN is that the RNN can be used as an 'end-to-end' model because it replaces a combination of different techniques that are currently used in sequence modeling, such as the HMM. However, both these approaches still rely on pre-defined features as input.

While using features such as MFCCs that collapse high dimensional speech sound waves into low dimensional encodings have been successful in speech recognition systems, such low dimensional encodings may lose some relevant information. On the other hand, there are approaches that build their own features instead of using pre-defined features. The work by Jaitly and Hinton (2011) used raw speech as input to a RBM and achieved a classification error of 21.8% on the TIMIT data set. Another approach that uses raw data is learning the auditory codes using spiking population code (Smith and Lewicki, 2005). In this model, each spike encodes the precise time position and magnitude of a localized, time varying kernel function. The learned representations (basis vectors) show a striking resemblance to the cochlear filters in the auditory cortex.

Similarly sparse coding for audio classification is used by Grosse et al. (2007). The authors used features as input and a shift-invariant sparse coding model that reconstructs a time-series input using all the basis functions in all possible shifts. The model was evaluated on speaker identification and music genre classification.

A multimodal framework was explored by Ngiam et al. (2011) where video data of spoken digits and letters where combined with the audio data to improve the classification.

In conclusion, there have been a lot of recent improvements to the previous dominance of the features-GMM-HMM structure that has been used in speech recognition. First, there is a trend towards replacing GMM with a feature learning model such as deep belief networks or sparse coding. Second, there is a trend towards replacing HMM with other alternatives. One of them is the conditional random field (CRF) (Lafferty et al., 2001) that have been shown to outperform HMM, see for example the work by van Kasteren et al. (2008) and Bengio and Frasconi (1996). However, to date, the best reported result is replacing both parts of GMM-HMM with RNN (Graves et al., 2013). A next possible step for speech processing would be to replace the pre-made features with algorithms that build even better features from raw data.

4.4. Music recognition

Music recognition is similar to speech recognition with the exception that the data can be multivariate and either presented as raw acoustic signals or by discrete chords. In music recognition, a number of sub-problems are considered, such as music annotation (genre, chord, instrument, mood classification), music retrieval (text-based content search, content-based similarity retrieval, organization), and tempo identification. For music recognition, a commonly used set of features are MFCCs, chroma, constant-Q spectrograms (CQT) (Schoerkhuber and Klapuri, 2010), local contrast normalization (LCN) (LeCun et al., 2010), or Compressive Sampling (CS) (Chang et al., 2010). However, there is an increasing interest in learning the features from the data instead of using highly engineered features based on acoustic knowledge. A widely used data set for music genre recognition is GTZAN[3]. Even though it is possible to solve many tasks on text-based meta-data, such as user data (playlists, song history, social structure), there is still a need for content-based analysis. The reasons for this is that manual labeling is inefficient due to the large amount of music content and some tasks require the well-trained ear of an expert, e.g., chord recognition.

The work by Humphrey et al. (2013) gives a review and future directions for music recognition. In this work, three deficiencies are identified: hand-crafted features are sub-optimal and unsustainable to develop for each task, shallow architectures are fundamentally limited, and short-time analysis cannot encode a musically meaningful structure. To handle these deficiencies it is proposed to learn features automatically, apply deep architectures, and model longer time-dependencies than the current use of data in milliseconds.

The work by Nam et al. (2012) addresses the first deficiency by presenting a processing pipeline for automatically learning features for music recognition. The model follows the structure of a high-dimensional single layer network with max-pooling separately after learning the features (Coates et al., 2010). The input data is taken from multiple audio frames and fed into three different feature learning algorithms, namely K-means clustering, sparse coding, and RBM. The learned features gave better performance compared to MFCC, regardless of the feature learning algorithm.

Sparse coding have been used by Grosse et al. (2007) for learning features for music genre recognition. The work by Henaff et al. (2011) used Predictive Sparse Decomposition (PSD), which is similar to sparse coding, and achieved

[3] `http://marsyas.info/download/data_sets`

an accuracy of 83.4% on the GTZAN data. In this work, the features are automatically learned from CTQ spectograms in an unsupervised manner. The learned features capture information about which chords are being played in a particular frame and produce comparable results to hand-crafted features for the task of genre recognition. A limitation, however, is that it ignores temporal dependencies between frames.

Convolutional DBNs were used by Lee et al. (2009b) to learn features from speech and music spectrograms and from engineered features by Dieleman et al. (2011). The work by (Hamel and Eck, 2010) also uses convolutional DBN to achieve an accuracy of 84.3% on the GTZAN dataset.

Self-taught learning have also been used for music genre classification. The self-taught learning framework attempts to use unlabeled data that does not share the labels of the classification task to improve classification performance (Raina et al., 2007; Jialin Pan and Yang, 2010). Self-taught learning and sparse coding are used by Markov and Matsui (2012) where unlabeled data from other music genres other than in the classification task was used to train the model.

In conclusion, there are many works that use unsupervised feature learning methods for music recognition. The motivation for using deep networks is that music itself is structured hierarchically by a combination of chords, melodies and rhythms that creates motives, phrases, sections and finally entire pieces (Humphrey et al., 2013). Just like in speech recognition, the input data is often in some form of spectrograms. Many works leave the natural step of learning features from raw data as future work (Nam, 2012). Still, as proposed by (Humphrey et al., 2013), even though convolutional networks have given good results on time-frequency representations of audio, there is room for discovering new and better models.

4.5. Motion capture data

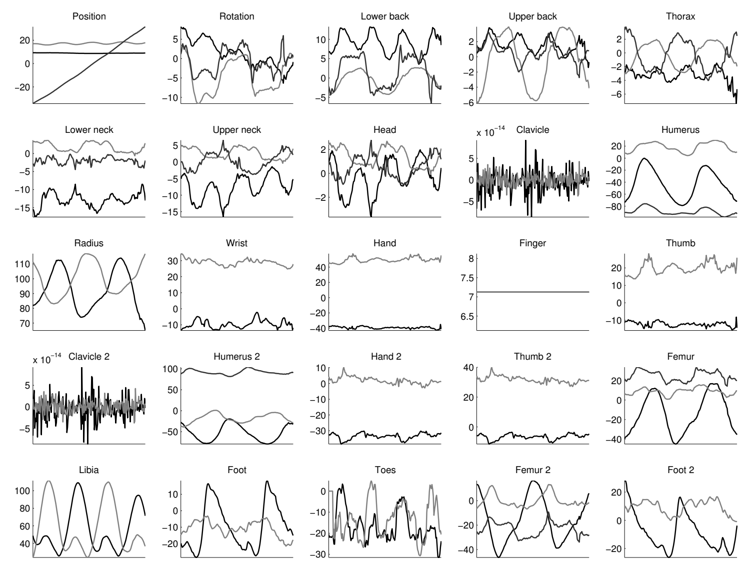

Figure 9: A sequence of human motion from the CMU motion capture data set.

Modeling human motion has several applications such as tracking, activity recognition, style and content separation, person identification, computer animation, and synthesis of new motion data. Motion capture data is collected from recordings of movements from several points on the body of a human actor. These points can be captured by cameras that either track the position of strategically placed markers (usually at joint centers) or uses vision-based algorithms for tracking points of interest (Gleicher, 2000). The points are represented as 3D Cartesian coordinates over time and are used to form a skeletal structure with constant limb lengths by translating the points to relative joint angles. The joint angles can be expressed in Euler angles, 4D quaternions, or exponential map parameterization (Grassia, 1998) and can have 1-3 degrees of freedom (DOF) each. The full data set consists of the orientation and translation of the root and all relative joint angles for each time frame as well as the constant skeleton model. The data is noisy, high-dimensional, and multivariate with complex nonlinear relationships. It has a lower frequency compared to speech and music data and some of the signals may be task-redundant.

Some of the traditional approaches include the work by Brand and Hertzmann (2000), which models both the style and content of human motion using Hidden Markov Models (HMMs). The different styles were learned from unlabeled data and the trained model was used to synthesize motion data. A linear dynamical systems was used by Chiappa et al. (2009) to model three different motions of a human performing the task of holding a cup that has a ball attached to it with a string and then try to catch the ball into the cup (game of Balero). A Bayesian mixture of linear Gaussian state-space models (LGSSM) was trained with data from a human learner and used to generate new motions that was clustered and simulated on a robotic manipulator.

Both HMMs and linear dynamical systems are limited by their ability to model complex full-body motions. The work by Wang et al. (2007) uses Gaussian Processes to model three styles of locomotive motion (walk, run, stride) from the CMU motion capture data set[4], see Figure 9. The CMU data set have also been used to generate motion capture from just a few initialization frames with a Temporal RBM (TRBM) (Sutskever and Hinton, 2006) and a conditional RBM (cRBM) Taylor et al. (2007). Better

[4] `http://mocap.cs.cmu.edu/`

modeling and smoother transition between different styles of motions was achieved by adding a second hidden layer to the cRBM, using the Recurrent TRBM (Sutskever et al., 2008), and using the factored conditional RBM (fcRBM) (Taylor and Hinton, 2009). The work by Längkvist and Loutfi (2012) restructures an auto-encoder to resemble a cRBM but is used to perform classification on the CMU motion capture data instead of generating new sequences. The drawbacks with general-purpose models such as Gaussian Processes and cRBM are that prior information about motion is not utilized and they have a costly approximation sampling procedure.

An unsupervised hierarchical model that is specifically designed for modeling locomotion styles was developed by Pan and Torresani (2009) and builds on the Hierarchical Bayesian Continuous Profile Model (HB-CPM). A Dynamic Factor Graph (DFG), which is an extension of factor graphs, was introduced by Mirowski and LeCun (2009) and used on motion capture data to fill in missing data. The advantage of DFG is that it has a constant partition function which avoids the costly approximation sampling procedure that is used in a cRBM.

In summary, analyzing and synthesizing motion capture data is a challenging task and it encourages researchers to further improve learning algorithms for dealing with complex, multivariate time-series data. A motivation for using deep learning algorithms for motion capture data is that it has been suggested that human motion is composed of elementary building blocks (motion templates) and any complex motion is constructed from a library of these previously learned motion templates (Flash and Hochner, 2005). Deep networks can, in an unsupervised manner, learn these motion templates from raw data and use them to form complex human motions. Motion capture data also provides an interesting platform for feature learning from raw data since there is no commonly used feature set for motion capture data. Therefore, the success of applying deep learning algorithms to motion data can inspire learning features from raw data in other time-series problems as well.

4.6. Electronic nose data

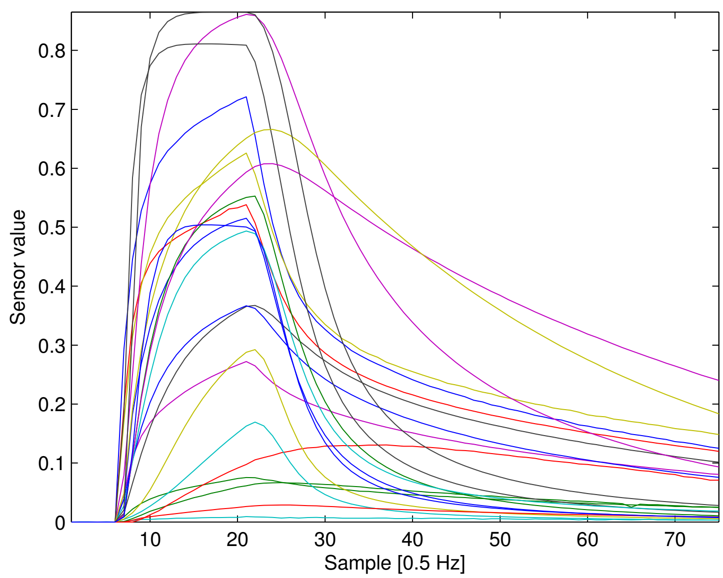

Figure 10: Normalized data from an array of electronic nose sensors.

Machine olfaction (Osuna et al., 2003; Gardner and Bartlett, 1999) is a field that seeks to quantify and analyze odours using an electronic nose (e-nose). An e-nose is composed of an array of selective gas sensors together with pattern recognition techniques. Figure 10 shows the readings from an e-nose sensor array. The number of sensors in the array typically ranges from 4-30 sensors and are therefore, just like motion capture data, multivariate and may contain redundant signals. The data is also unintuitive and there is a lack of expert knowledge that can guide the design of features. E-noses are mostly used in practice for industrial applications such as measuring food, beverage (Gardner et al., 2000b), and air quality (Zampolli et al., 2004), gas identification, and gas source localization Bennetts et al. (2011), but also has medical applications such as bacteria identification (Dutta et al., 2002) and diagnosis (Gardner et al., 2000a).

The traditional approach of analyzing e-nose data involves extracting information in the static and dynamic phases of the signals (Gutierrez-Osuna, 2002) for the use of static pattern analysis techniques (PCA, discriminant function analysis, cluster analysis and neural networks). Some commonly used features are the static sensor response, transient derivatives (Trincavelli et al., 2010), area under the curve (Carmona et al., 2006), model parameter identification (Vembu et al., 2012), and dynamic analysis (Hines et al., 1999).

A popular approach for modeling e-nose data is the Time-Delay Neural Networks (TDNN) (Waibel et al., 1989). It has been used for identifying the smell of spices (Zhang et al., 2003), ternary mixtures (Vito et al., 2007), optimum fermentation time for black tea (Bhattacharya et al., 2008), and vintages of wine (Yamazaki et al., 2001). An RNN have been used for odour localization with a mobile robot (Duckett et al., 2001).

The work by Vembu et al. (2012) compares the gas discrimination and localization between three approaches: SVM on raw data, SVM on features extracted from auto-regressive and linear dynamical systems, and finally a SVMs with kernels specialized for structured data (Gärtner, 2003). The SVM with built-in time-aware kernels performed better than techniques that used feature extraction, even though the features captured temporal information.

More recently, an auto-encoder, RBM, and cRBM have been used for bacteria identification (Längkvist and Loutfi, 2011) and fast classification of meat spoilage markers (Längkvist et al., 2013).

E-nose data introduces the challenge of improving models that can deal with redundant signals. It is not feasible to produce tailor-made sensors for each possible individual gas and combinations of gases of interest. Therefore the common approach is to use an array of sensors with different properties and leave the discrimination to the pattern analysis software. It is also not desirable to construct new feature sets for each e-nose application so a data-driven feature learning method is useful. The early works on e-nose data create feature vectors of simple features for each signal such as the static response or the slope of dynamic response and then feed it to a classifier. Recently, the use of dynamic models such as neural networks with tapped delays and SVMs with kernels for structured data have shown to improve the performance over static approaches. The next step is to continue this trend of using dynamical models that constructs robust features that can deal with noisy inputs in order to quantify and classify odors in more challenging open environments with many different simultaneous gas sources.

4.7. Physiological data

With physiological data we consider recordings such as electroencephalography (EEG), magnetoencephalography (MEG), electrocardiography (ECG), and wearable sensors for health monitoring. Figure 11 shows an example of how physiological data look like. The data can exist both as singular or multiple channels. The use of a feature learning algorithm is particularly

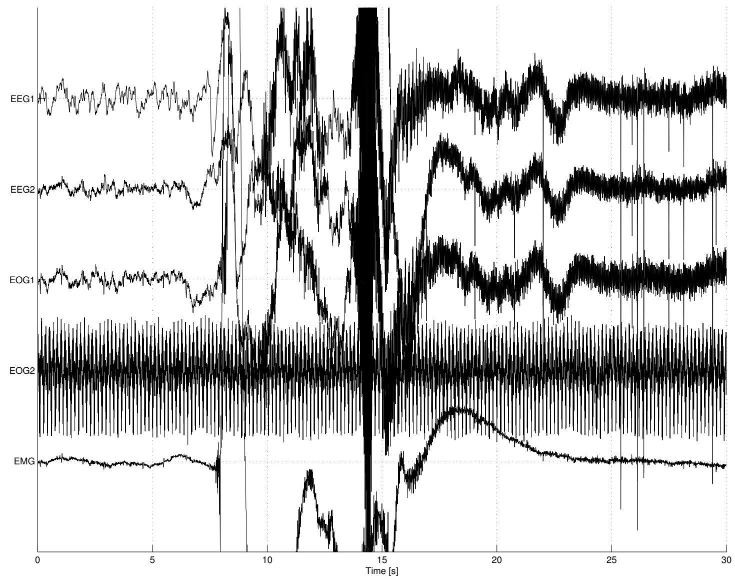

Figure 11: Data from EEG (top two signals), EOG (third and fourth signal), and EMG (bottom signal), recorded with a polysomnograph during sleep.

beneficial in medical applications because acquiring a labeled medical data set is expensive since the data sets are often very large and require the labeling of an expert in the field.

The work by Mirowski et al. (2008) compares convolutional networks with logistic regression and SVMs for epileptic seizure prediction from intracranial EEG signals. The features that are used are hand-engineered bivariate features between channels that encode relationship between pairs of EEG channels. The result was that convolutional networks achieved only 1 false-alarm prediction from 21 patients while the SVM had 10 false-alarms. TDNN and ICA has also been used for EEG-based prediction of epileptic seizures (Mirowski et al., 2007). The application of self-organizing maps (SOM) to analyze EMG data is presented by Tucker (1999).

A RBM-based method that builds features from raw data for sleep stage classification from 4-channel polysomnography data has been proposed by Längkvist et al. (2012). A similar setup was used by Wulsin et al. (2011) for modeling single channel EEG waveforms used for anomaly detection. A DBN is used by (Wang and Shang, 2013) to automatically extract features from raw unlabelled physiological data and achieves better classification than a feature-based approach. These recent works show that DBNs can be applied to raw physiological data to effectively learn relevant features.

A source separation method tailor-made to EEG and MEG signals is proposed by Hyvärinen et al. (2010). The data is preprocessed by short-time Fourier transforms and then fed to an ICA. The work shows that temporal correlations are adequately taken into account. Independent Component Analysis (ICA) has provided to be a new tool to analyze time series and is a unifying framework that combines sparseness, temporal coherence, topography and complex cell pooling in a single model (Hyvärinen et al., 2003). A method for how to order the independent components for time-series is explored by Cheung and Xu (2001).

Self-taught learning has been used with time-series data from wearable hand-motion sensors (Amft, 2011).

The field of physiological data is large and many different methods have been used. The characteristics of physiological data could be particularly interesting for the deep learning community because it can be used to explore the feasibility of learning features from raw data, which hopefully can inspire similar approaches in other time-series domains.

Table 2: A summary of commonly used time-series problems.

Problem	Multi-variate	Raw data	Frequency rich	Common features	Common method	Benchmark set
Stock prediction	-	✓	-	-	ANN	DJIA
Video	✓	✓	-	SIFT, HOG	ConvRBM	KTH
Speech Recognition	-	(✓)	✓	MFCC	RBM, RNN	TIMIT
Music recognition	✓	-	✓	Chroma, MFCC	ConvRBM	GTZAN
Motion capture	✓	✓	-	-	cRBM	CMU
E-nose	✓	✓	-	Many	TDNN	-
Physiological data	✓	(✓)	✓	Many, spectogram	RBM, AE	PhysioNET

4.8. Summary

Table 2 gives a summary of the time-series problems that have been presented in this section. The first column indicates if the data is multivariate (or only contains one signal, univariate). Stock prediction is often viewed as a single channel problem, which explains the difficulties to produce accurate prediction systems, since stocks depend on a myriad of other factors, and arguably not at all on past values of the stock itself. For speech recognition, the use of multimodal sources can improve performance (Ngiam et al., 2011).

The second column shows which problems have attempted to create features purely from raw data. Only a few works have attempted this with speech recognition (Jaitly and Hinton, 2011; Smith and Lewicki, 2005) and physiological data (Wulsin et al., 2011; Längkvist et al., 2012; Wang and Shang, 2013). To the authors knowledge, learning features from raw data has not been attempted in music recognition. The process of constructing features from raw data has been well demonstrated for vision-tasks but is cautiously used for time-series problems. Models such as TDNN, cRBM and convolutional RBMs are well suited for being applied to raw data (or slightly pre-processed data).

The third column indicates which time-series problems have valuable information in the frequency-domain. For frequency-rich problems, it is uncommon to attempt to learn features from raw data. A reason for this is that current feature learning algorithms are yet not well-suited for learning features in the frequency-domain.

The fourth column displays some common features that have been used in the literature. SIFT and HOG have been applied to videos even though those features are developed for static images. Chroma and MFCC have been applied to music recognition, even though they are develop for speech recognition. The e-nose community have tried a plethora of features. E-nose data is a relatively new field where a hand-crafted feature set have not been developed since this kind of data is complex and unintuitive. For physiological data, the used features are often a combination of application-specific features from previous works or hand-crafted features.

The fifth column reports the most commonly used method(s), or current state-of-the-art, for each time-series problem. For stock prediction, the progress has stopped at classical neural networks. The current state-of-the-art augments additional information beside the stock data. For high-dimensional temporal data such as video and music recognition, the convolutional version of RBM have been successful. In recent years, the RBM have been used for speech recognition but the current state-of-the-art is achieved with an RNN. The cRBM introduced motion capture data to the deep learning community and it is an interesting problem to explore with other methods. Single layer neural networks with temporal capabilities have been used to model e-nose data and the use of deep networks is an interesting future direction for modeling e-nose data.

And finally, the last column indicates a typical benchmark set for each problem. There is currently no well-known publicly available benchmark data set for e-nose data. For deep learning to enter the field of e-nose data it requires a large, well-organized data set that would benefit both communities. A data base of physiological data is available from PhysioNET (Goldberger et al., 2000 (June 13).

5. Conclusion

Unsupervised feature learning and deep learning techniques have been successfully applied to a variety of domains. While much focus in deep learning and unsupervised feature learning have been in the computer vision domain, this paper has reviewed some of the successful applications of deep learning methods to the time-series domain. Some of these approaches have treated the input as static data but the most successful ones are those that have modified the deep learning models to better handle time-series data.

The problem with processing time-series data as static input is that the importance of time is not captured. Modeling time-series faces many of the same challenges as modeling static data, such as coping with high-dimensional observations and nonlinear relationships between variables, however, by simply ignoring time and applying models of static data to time series one disregards much of the rich structure present in the data. When taking this approach, the context of the current input frame is lost and the only time-dependencies that are captured is within the input size. In order to capture long-term dependencies, the input size has to be increased, which can be impractical for multivariate signals or if the data has very long-term dependencies. The solution is to use a model that incorporates temporal coherence, performs temporal pooling, or models sequences of hidden unit activations.

The choice of model and how the data should be presented to the model is highly dependent on the type of data. Within a chosen model there are additional design choices in terms of connectivity, architecture, and hyperparameters. For these reasons, even though many unsupervised feature learning models offer to relieve the user of having to come up with useful features for the current domain, there are still many challenges for applying them to time-series data. It is also worth noting that many works that construct useful features from the input data actually still use input data from pre-processed features.

Deep learning methods offer better representation and classification on a multitude of time-series problems compared to shallow approaches when configured and trained properly. There is still room for improving the learning algorithms specifically for time-series data, e.g., performing signal selection that deals with redundant signals in multivariate input data. Another possible future direction is to develop models that change their internal architecture during learning or use model averaging in order to capture both short and long-term time dependencies. Further research in this area is needed to develop algorithms for time-series modeling that learn even better features and are easier and faster to train. Therefore, there is a need to focus less on the pre-processing pipeline for a specific time-series problem and focus more on learning better feature representations for a general-purpose algorithm for structured data, regardless of the application.

References

Agrawal, J.G., Chourasia, V.S., Mittra, A.K., 2013. State-of-the-art in stock prediction techniques. International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering 2, 1360–1366.

Amft, O., 2011. Self-taught learning for activity spotting in on-body motion sensor data, in: ISWC 2011: Proceedings of the IEEE International Symposium on Wearable Computing, IEEE. pp. 83–86.

Atsalakis, G.S., Valavanis, K.P., 2009. Surveying stock market forecasting techniques Íc part ii: Soft computing methods. Expert Systems with Applications 36, 5932 – 5941.

Bengio, Y., 2007. Learning deep architectures for AI. Technical Report 1312. Dept. IRO, Universite de Montreal.

Bengio, Y., Courville, A., Vincent, P., 2012. Unsupervised Feature Learning and Deep Learning: A Review and New Perspectives. Technical Report arXiv:1206.5538. U. Montreal. URL: `http://arxiv.org/abs/1206.5538`.

Bengio, Y., Frasconi, P., 1996. Input-output HMM's for sequence processing. IEEE Transactions on Neural Networks 7(5), 1231–1249.

Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H., 2007. Greedy layer-wise training of deep networks. Advances in neural information processing systems 19, 153.

Bengio, Y., LeCun, Y., 2007. Scaling learning algorithms towards AI, in: Bottou, L., Chapelle, O., DeCoste, D., Weston, J. (Eds.), Large-Scale Kernel Machines, MIT Press.

Bengio, Y., Simard, P., Frasconi, P., 1994. Learning longterm dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks 5(2), 157–166.

Bengio, Y., Yao, L., Alain, G., Vincent, P., 2013. Generalized denoising auto-encoders as generative models. CoRR abs/1305.6663.

Bennetts, V.H., Lilienthal, A.J., Neumann, P.P., Trincavelli, M., 2011. Mobile robots for localizing gas emission sources on landfill sites: is bio-inspiration the way to go? Frontiers in neuroengineering 4.

Bhattacharya, N., Tudu, B., Jana, A., Ghosh, D., Bandhopadhyaya, R., Bhuyan, M., 2008. Preemptive identification of optimum fermentation time for black tea using electronic nose. Sensors and Actuators B: Chemical 131, 110–116.

Bo Chen, Jo-Anne Ting, B.M., de Freitas, N., 2010. Deep learning of invariant spatio-temporal features from video, in: NIPS 2010 Deep Learning and Unsupervised Feature Learning Workshop.

Bollen, J., Mao, H., Zeng, X., 2011. Twitter mood predicts the stock market. Journal of Computational Science 2, 1 – 8.

Bottou, L., 2010. Large-scale machine learning with stochastic gradient descent, in: Lechevallier, Y., Saporta, G. (Eds.), Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT'2010), Springer, Paris, France. pp. 177–187. URL: http://leon.bottou.org/papers/bottou-2010.

Brand, M., Hertzmann, A., 2000. Style machines, in: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA. pp. 183–192.

Carmona, M., Martinez, J., Zalacain, A., Rodriguez-Mendez, M.L., de Saja, J.A., Alonso, G.L., 2006. Analysis of saffron volatile fraction by td–gc–ms and e-nose. European Food Research and Technology 223, 96–101.

Chang, K., Jang, J., Iliopoulos, C., 2010. Music genre classification via compressive sampling, in: Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR), pp. 387–392.

Cheung, Y., Xu, L., 2001. Independent component ordering in ica time series analysis. Neurocomputing 41, 145–152.

Chiappa, S., Kober, J., Peters, J., 2009. Using bayesian dynamical systems for motion template libraries. In Adv. in Neural Inform. Proc. Systems 21, 297–304.

Coates, A., Lee, H., Ng, A.Y., 2010. An Analysis of Single-Layer Networks in Unsupervised Feature Learning. Engineering , 1–9.

Dahl, G., Yu, D., Deng, L., Acero, A., 2012. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. Audio, Speech, and Language Processing, IEEE Transactions on 20, 30–42. doi:10.1109/TASL.2011.2134090.

Dahl, G.E., Ranzato, M., Mohamed, A., Hinton, G., 2010. Phone recognition with the mean-covariance restricted boltzmann machine. Advances in Neural Information Processing Systems 23, 469–477.

Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection, in: In CVPR.

Dieleman, S., Brakel, P., Schrauwen, B., 2011. Audio-based music classification with a pretrained convolutional network, in: In The International Society for Music Information Retrieval (ISMIR).

Dietterich, T.G., 2002. Machine learning for sequential data: A review, in: Structural, Syntactic, and Statistical Pattern Recognition, Springer-Verlag. pp. 15–30.

Duckett, T., Axelsson, M., Saffiotti, A., 2001. Learning to locate an odour source with a mobile robot, in: Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on, pp. 4017–4022 vol.4. doi:10.1109/ROBOT.2001.933245.

Dutta, R., Hines, E., Gardner, J., Boilot, P., 2002. Bacteria classification using cyranose 320 electronic nose. Biomedical Engineering Online 1, 4.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P., Vincent, P., Bengio, S., 2010. Why does unsupervised pre-training help deep learning? Journal of Machine Learning Research 11, 625–660.

Fama, E.F., 1965. The behavior of stock-market prices. The Journal of Business 1, 34–105.

Flash, T., Hochner, B., 2005. Motor primitives in vertebrates and invertebrates. Current Opinion in Neurobiology 15(6), 660–666.

Furui, S., Kikuchi, T., Shinnaka, Y., Hori, C., 2004. Speech-to-text and speech-to-speech summarization of spontaneous speech. Speech and Audio Processing, IEEE Transactions on 12, 401–408.

Gardner, J., Bartlett, P., 1999. Electronic Noses, Principles and Applications. Oxford University Press, New York, NY, USA.

Gardner, J.W., Shin, H.W., Hines, E.L., 2000a. An electronic nose system to diagnose illness. Sensors and Actuators B: Chemical 70, 19–24.

Gardner, J.W., Shin, H.W., Hines, E.L., Dow, C.S., 2000b. An electronic nose system for monitoring the quality of potable water. Sensors and Actuators B: Chemical 69, 336–341.

Gärtner, T., 2003. A survey of kernels for structured data. SIGKDD Explor. Newsl. 5, 49–58.

Gers, F.A., Schmidhuber, J., Cummins, F., 2000. Learning to Forget: Continual Prediction with LSTM. Neural Computation 12, 2451–2471.

Gleicher, M., 2000. Animation from observation: Motion capture and motion editing. SIGGRAPH Computer Graphics 33, 51–54.

Goldberger, A.L., Amaral, L.A.N., Glass, L., Hausdorff, J.M., Ivanov, P.C., Mark, R.G., Mietus, J.E., Moody, G.B., Peng, C.K., Stanley, H.E., 2000 (June 13). PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. Circulation 101, e215–e220. Circulation Electronic Pages: http://circ.ahajournals.org/cgi/content/full/101/23/e215.

Grassia, F.S., 1998. Practical parameterization of rotations using the exponential map. J. Graph. Tools 3, 29–48.

Graves, A., Mohamed, A., Hinton, G., 2013. Speech recognition with deep recurrent neural networks, in: The 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP).

Grosse, R., Raina, R., Kwong, H., Ng, A.Y., 2007. Shift-invariant sparse coding for audio classification, in: Conference on Uncertainty in Artificial Intelligence (UAI).

Gruhl, D., Guha, R., Kumar, R., Novak, J., Tomkins, A., 2005. The predictive power of online chatter, in: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, pp. 78–87.

Gutierrez-Osuna, R., 2002. Pattern analysis for machine olfaction: A review. IEEE Sensors Journal 2(3), 189–202.

Hamel, P., Eck, D., 2010. Learning features from music audio with deep belief networks, in: 11th International Society for Music Information Retrieval Conference (ISMIR).

Henaff, M., Jarrett, K., Kavukcuoglu, K., LeCun, Y., 2011. Unsupervised learning of sparse features for scalable audio classification, in: Proceedings of International Symposium on Music Information Retrieval (ISMIR'11).

Hines, E., Llobet, E., Gardner, J., 1999. Electronic noses: a review of signal processing techniques. Circuits, Devices and Systems, IEE Proceedings - 146, 297–310.

Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A.r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., et al., 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. Signal Processing Magazine, IEEE 29, 82–97.

Hinton, G., Salakhutdinov, R., 2006. Reducing the dimensionality of data with neural networks. Science 313(5786), 504–507.

Hinton, G.E., 2002. Training products of experts by minimizing contrastive divergence. Neural Computation 14, 1771 – 1800.

Hinton, G.E., 2012. A practical guide to training restricted boltzmann machines, in: Montavon, G., Orr, G.B., Müller, K.R. (Eds.), Neural Networks: Tricks of the Trade. Springer Berlin Heidelberg. volume 7700 of *Lecture Notes in Computer Science*, pp. 599–619. URL: `http://dx.doi.org/10.1007/978-3-642-35289-8_32`, doi:`10.1007/978-3-642-35289-8_32`.

Hinton, G.E., Krizhevsky, A., Wang, S.D., 2011. Transforming auto-encoders, in: Proceedings of the 21th international conference on Artificial neural networks - Volume Part I, pp. 44–51.

Hinton, G.E., S., O., Y., T., 2006. A fast learning algorithm for deep belief nets. Neural Computation 18 , 1527–1554.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Computation 9, 1735–1780.

Hsieh, T.J., Hsiao, H.F., Yeh, W.C., 2011. Forecasting stock markets using wavelet transforms and recurrent neural networks: An integrated system based on artificial bee colony algorithm. Applied Soft Computing 11, 2510 – 2525.

Humphrey, E.J., Bello, J.P., LeCun, Y., 2013. Feature learning and deep architectures: new directions for music informatics. Journal of Intelligent Information Systems 41, 461–481.

Hüsken, M., Stagge, P., 2003. Recurrent Neural Networks for Time Series Classification. Neurocomputing 50, 223–235.

Hyvärinen, A., Hurri, J., Väyrynen, J., 2003. Bubbles: a unifying framework for low-level statistical properties of natural image sequences. J. Opt. Soc. Am. A 20, 1237–1252.

Hyvärinen, A., Ramkumar, P., Parkkonen, L., Hari, R., 2010. Independent component analysis of short-time Fourier transforms for spontaneous EEG/MEG analysis. NeuroImage 49(1), 257–271.

Hyvèarinen, A., Hurri, J., Hoyer, P.O., 2009. Natural Image Statistics. volume 39. Springer.

Jaitly, N., Hinton, G., 2011. Learning a better representation of speech soundwaves using restricted boltzmann machines, in: Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on, IEEE. pp. 5884–5887.

Jialin Pan, S., Yang, Q., 2010. A survey on transfer learning. IEEE Transactions On Knowledge and Data Engineering 22.

Kamyshanska, H., Memisevic, R., 2013. On autoencoder scoring, in: Proceedings of the 30th International Conference on Machine Learning (ICML-13), JMLR Workshop and Conference Proceedings. pp. 720–728.

van Kasteren, T., Noulas, A., Kröse, B., 2008. Conditional random fields versus hidden markov models for activity recognition in temporal sensor data, in: In Proceedings of the 14th Annual Conference of the Advanced School for Computing and Imaging (ASCI'08), The Netherlands.

Kavukcuoglu, K., Ranzato, M., Fergus, R., Le-Cun, Y., 2009. Learning invariant features through topographic filter maps, in: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE. pp. 1605–1612.

Keogh, E., Kasetty, S., 2002. On the need for time series data mining benchmarks: A survey and empirical demonstration, in: In proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 102–111.

Kim, S.S., 1998. Time-delay recurrent neural network for temporal correlations and prediction. Neurocomputing 20, 253 – 263.

Lafferty, J.D., McCallum, A., Pereira, F.C.N., 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data, in: Proceedings of the Eighteenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. pp. 282–289.

Längkvist, M., Coradeschi, S., Loutfi, A., Rayappan, J.B.B., 2013. Fast Classification of Meat Spoilage Markers Using Nanostructured ZnO Thin Films and Unsupervised Feature Learning. Sensors 13(2), 1578–1592. Doi:10.3390/s130201578.

Längkvist, M., Karlsson, L., Loutfi, A., 2012. Sleep stage classification using unsupervised feature learning. Advances in Artificial Neural Systems 2012. Doi:10.1155/2012/107046.

Längkvist, M., Loutfi, A., 2011. Unsupervised feature learning for electronic nose data applied to bacteria identification in blood, in: NIPS workshop on Deep Learning and Unsupervised Feature Learning.

Längkvist, M., Loutfi, A., 2012. Not all signals are created equal: Dynamic objective auto-encoder for multivariate data, in: NIPS workshop on Deep Learning and Unsupervised Feature Learning.

Le, Q.V., Zou, W.Y., Yeung, S.Y., Ng, A.Y., 2011. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis, in: Computer Vision and Pattern Recognition (CVPR).

Le Roux, N., Bengio, Y., 2008. Representational power of restricted Boltzmann machines and deep belief networks. Neural Computation 20, 1631–1649.

LeCun, Y., Kavukvuoglu, K., Farabet, C., 2010. Convolutional networks and applications in vision, in: Proc. International Symposium on Circuits and Systems (ISCASąŕ10), IEEE.

Lee, H., Ekanadham, C., Ng, A.Y., 2008. Sparse deep belief net model for visual area V2, in: Advances in Neural Information Processing Systems 20, pp. 873–880.

Lee, H., Grosse, R., Ranganath, R., Ng, A.Y., 2009a. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations, in: Twenty-Sixth International Conference on Machine Learning.

Lee, H., Largman, Y., Pham, P., Ng, A.Y., 2009b. Unsupervised feature learning for audio classification using convolutional deep belief networks, in: Advances in Neural Information Processing Systems 22, pp. 1096–1104.

Li, Y., Ma, W., 2010. Applications of artificial neural networks in financial economics: A survey, in: Proceedings of the 2010 International Symposium on Computational Intelligence and Design - Volume 01, IEEE Computer Society. pp. 211–214.

Lin, X., Yang, Z., Song, Y., 2009. Short-term stock price prediction based on echo state networks. Expert Systems with Applications 36, 7313 – 7317.

Lowe, D., 1999. Object recognition from local scale-invariant features, in: In ICCV.

Luenberger, D., 1979. Introduction to Dynamic Systems: Theory, Models, and Applications. Wiley.

Lütkepohl, H., 2005. New Introduction to Multiple Time Series Analysis. Springer-Verlag.

Malkiel, B., 2003. The efficient market hypothesis and its critics. The Journal of Economic Perspectives 17. Http://dx.doi.org/10.2307/3216840.

Markov, K., Matsui, T., 2012. Music genre classification using self-taught learning via sparse coding, in: Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on, pp. 1929–1932.

Martens, J., Sutskever, I., 2012. Training deep and recurrent neural networks with hessian-free optimization, in: Neural Networks: Tricks of the Trade. Springer Berlin Heidelberg. volume 7700 of *Lecture Notes in Computer Science*.

Masci, J., Meier, U., Cireşan, D., Schmidhuber, J., 2011. Stacked convolutional auto-encoders for hierarchical feature extraction, in: Proceedings of the 21th international conference on Artificial neural networks - Volume Part I, pp. 52–59.

Memisevic, R., Hinton, G., 2007. Unsupervised learning of image transformations, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–8.

Memisevic, R., Hinton, G.E., 2010. Learning to represent spatial transformations with factored higher-order boltzmann machines. Neural Computation 22, 1473–1492.

Mirowski, P., LeCun, Y., 2009. Dynamic factor graphs for time series modeling. Machine Learning and Knowledge Discovery in Databases , 128–143.

Mirowski, P., Madhavan, D., LeCun, Y., 2007. Time-delay neural networks and independent component analysis for eeg-based prediction of epileptic seizures propagation, in: Association for the Advancement of Artificial Intelligence Conference.

Mirowski, P.W., LeCun, Y., Madhavan, D., Kuzniecky, R., 2008. Comparing SVM and convolutional networks for epileptic seizure prediction from intracranial EEG, in: Machine Learning for Signal Processing, 2008. MLSP 2008. IEEE Workshop on, IEEE. pp. 244–249.

Mohamed, A., Dahl, G.E., Hinton, G., 2012. Acoustic modeling using deep belief networks. IEEE Transactions on Audio, Speech, and Language Processing archive 20(1), 14–22.

Mohamed, A., Hinton, G., 2010. Phone recognition using restricted boltzmann machines, in: Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on, pp. 4354–4357. doi:10.1109/ICASSP.2010.5495651.

Nam, J., 2012. Learning Feature Representations for Music Classification. Ph.D. thesis. Stanford University.

Nam, J., Herrera, J., Slaney, M., Smith, J.O., 2012. Learning Sparse Feature Representations for Music Annotation and Retrieval, in: In The International Society for Music Information Retrieval (ISMIR), pp. 565–570.

Nanopoulos, A., Alcock, R., Manolopoulos, Y., 2001. Feature-based classification of time-series data. International Journal of Computer Research 10, 49–61.

Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., Ng, A.Y., 2011. Multimodal deep learning, in: In Proceedings of the Twenty-Eigth International Conference on Machine Learning.

Osuna, G.R., Nagle, T.H., Kermani, B., Schiffman, S.S., 2003. HandBook of Machine Olfaction, electronic nose technology. Wiley-Vch Verlag GmbH & Co. KGaA. chapter Signal Conditioning and Preprocessing. pp. 105–132.

Pan, W., Torresani, L., 2009. Unsupervised hierarchical modeling of locomotion styles, in: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 785–792.

Parris, E., Carey, M., 1996. Language independent gender identification, in: Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on, pp. 685–688 vol. 2.

Pascanu, R., Mikolov, T., Bengio, Y., 2012. Understanding the exploding gradient problem. Computing Research Repository (CoRR) abs/1211.5063.

Rabiner, L., Juang, B., 1986. An introduction to hidden markov models. IEEE ASSP Magazine 3(1), 4–16.

Raina, R., Battle, A., Lee, H., Packer, B., Ng, A.Y., 2007. Self-taught learning: Transfer learning from unlabeled data, in: Proceedings of the Twenty-fourth International Conference on Machine Learning.

Ranzato, M., Hinton, G., 2010. ąřmodeling pixel means and covariances using factorized third-order boltzmann machines, in: in Proc. of Computer Vision and Pattern Recognition Conference (CVPR 2010).

Ranzato, M., Krizhevsky, A., Hinton, G., 2010. Factored 3-way restricted boltzmann machines for modeling natural images, in: in Proceedings of the International Conference on Artificial Intelligence and Statistics.

Ranzato, M., Poultney, C., Chopra, S., LeCun, Y., 2006. Efficient learning of sparse representations with an energy-based model, in: et al., J.P. (Ed.), Advances in Neural Information Processing Systems (NIPS 2006), MIT Press.

Saxe, A., Koh, P., Chen, Z., Bhand, M., Suresh, B., Ng, A.Y., 2011. On random weights and unsupervised feature learning, in: In Proceedings of the Twenty-Eighth International Conference on Machine Learning.

Schoerkhuber, C., Klapuri, A., 2010. Constant-q transform toolbox for music processing, in: 7th Sound and Music Computing Conference.

Smith, E., Lewicki, M.S., 2005. Learning efficient auditory codes using spikes predicts cochlear filters, in: In Advances in Neural Information Processing Systems, MIT Press.

Stavens, D., Thrun, S., 2010. Unsupervised learning of invariant features using video, in: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, pp. 1649–1656.

Sugiyama, M., Sawai, H., Waibel, A., 1991. Review of tdnn (time delay neural network) architectures for speech recognition, in: Circuits and Systems, 1991., IEEE International Sympoisum on, pp. 582–585 vol.1.

Sutskever, I., 2012. Training Recurrent Neural Networks. Ph.D. thesis. University of Toronto.

Sutskever, I., Hinton, G., 2006. Learning multilevel distributed representations for high-dimensional sequences. Technical Report. University of Toronto.

Sutskever, I., Hinton, G.E., Taylor, G.W., 2008. The recurrent temporal restricted boltzmann machine, in: Advances in Neural Information Processing Systems, pp. 1601–1608.

Taylor, G., Fergus, R., LeCun, Y., Bregler, C., 2010. Convolutional learning of spatio-temporal features, in: Proc. European Conference on Computer Vision (ECCV'10).

Taylor, G., Hinton, G., 2009. Factored conditional restricted boltzmann machines for modeling motion style, in: Proc. of the 26th International Conference on Machine Learning (ICML).

Taylor, G., Hinton, G.E., Roweis, S., 2007. Modeling human motion using binary latent variables, in: Advances in Neural Information Processing Systems.

Taylor, G.W., 2009. Composable, distributed-state models for high-dimensional time series. Ph.D. thesis. Departmet of Computer Science University of Toronto.

Trincavelli, M., Coradeschi, S., Loutfi, A., Söderquist, B., Thunberg, P., 2010. Direct identification of bacteria in blood culture samples using an electronic nose. IEEE Trans Biomedical Engineering 57, 2884–2890.

Tsai, C.F., Hsiao, Y.C., 2010. Combining multiple feature selection methods for stock prediction: Union, intersection, and multi-intersection approaches. Decision Support Systems 50, 258 – 269.

Tucker, C., 1999. Self-organizing maps for time series analysis of electromyographic data, in: Neural Networks, 1999. IJCNN '99. International Joint Conference on, pp. 3577–3580.

Vembu, S., Vergara, A., Muezzinoglu, M.K., Huerta, R., 2012. On time series features and kernels for machine olfaction. Sensors and Actuators B: Chemical 174, 535–546.

Vito, S.D., Castaldo, A., Loffredo, F., Massera, E., Polichetti, T., Nasti, I., Vacca, P., Quercia, L., Francia, G.D., 2007. Gas concentration estimation in ternary mixtures with room temperature operating sensor array using tapped delay architectures. Sensors and Actuators B: Chemical 124, 309 – 316.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., Lang, K., 1989. Phoneme recognition using time-delay neural networks. IEEE Trans. Acoust., Speech, Signal Processing 37, 328–339.

Wang, D., Shang, Y., 2013. Modeling physiological data with deep belief networks. International Journal of Information and Education Technology 3.

Wang, J.M., Fleet, D.J., Hertzmann, A., 2007. Multi-factor gaussian process models for style-content separation, in: International Conference of Machine Learning (ICML), pp. 975–ÍC982.

Wiskott, L., Sejnowski, T.J., 2002. Slow feature analysis: Unsupervised learning of invariances. Neural computation 14, 715–770.

Wulsin, D., Gupta, J., Mani, R., Blanco, J., Litt, B., 2011. Modeling electroencephalography waveforms with semi-supervised deep belief nets: faster classification and anomaly measurement. Journal of Neural Engineering 8, 1741 – 2552.

Yamazaki, A., Ludermir, T., De Souto, M.C.P., 2001. Classification of vintages of wine by artificial nose using time delay neural networks. Electronics Letters 37, 1466–1467.

Yang, Q., Wu, X., 2006. 10 challenging problems in data mining research. International Journal of Information Technology & Decision Making 05, 597–604.

Zampolli, S., Elmi, I., Ahmed, F., Passini, M., Cardinali, G., Nicoletti, S., Dori, L., 2004. An electronic nose based on solid state sensor arrays for low-cost indoor air quality monitoring applications. Sensors and Actuators B: Chemical 101, 39–46.

Zhang, H., Balaban, M.O., Principe, J.C., 2003. Improving pattern recognition of electronic nose data with time-delay neural networks. Sensors and Actuators B: Chemical 96, 385–389.

Zhu, X., Wang, H., Xu, L., Li, H., 2008. Predicting stock index increments by neural networks: The role of trading volume under different horizons. Expert Systems with Applications 34, 3043 – 3054.

Zou, W.Y., Ng, A.Y., Yu, K., 2011. Unsupervised learning of visual invariance with temporal coherence, in: In NIPS 2011 Workshop on Deep Learning and Unsupervised Feature Learning.