Deep Network for Speech Emotion Recognition

—A Study of Deep Learning—

Zhuowei Han

Institut für Signalverarbeitung
und Systemtheorie

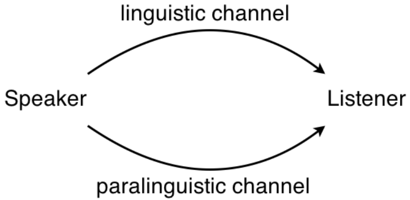
Universität Stuttgart

16/04/2015

Motivation

Speech Emotion Recognition

- Most current work focuses on speech processing based on linguistic information, e.g.: Skype Translator
- More natural human-machine interaction requires paralinguistic information such as age, gender, emotion.
- Speech Recognition / Speeker Identification / Emotion Recognition



Deep Learning

- Deep architecture for extracting complex structure and building internal representations from input
- New research area of machine learning (from shallow to deep structure)
- Widely applied in vision/audition processing, e.g. handwriting recognition (Graves, Alex, et al. 2009), traffic sign classification (Schmidhuber, et al. 2011), text translation (Google, 2014)

Table of Contents

Table of Contents

Foundations

Mel Frequency Cepstral Features

Emotion Recognition Approaches

Conditional Restricted Boltzmann Machine

Restricted Boltzmann Machine

Multilayer Neural Network

Function and Training

Problems and Solutions

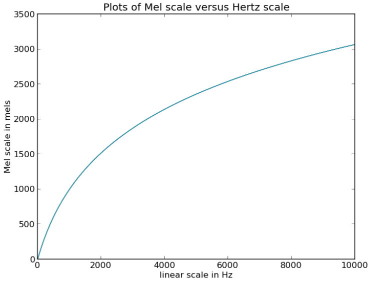
Long Short Term Memory

Recurrent Neural Network

Conclusion and Outlook

Mel Frequency Cepstral Features

- short-term power spectrum
- mel-scale approximate human perception
- widely-used in speech recognition tasks
- Transformation between Mel and Hertz scale



$$f_{mel} = 1125\ \ln\left(1 + f_{Hz}/700\right)$$
$$f_{Hz} = 700\left(\exp(f_{mel}/1125) - 1\right)$$

Emotion Recognition Approaches

Traditional Approaches

- pre-selected features
- supervised training
- low-level features not appropriate for claasification
- shallow structure of classifiers

Deep Learning Approaches

- learning representations from high-dim data
- extracting appropriate features without hand-crafting
- low-level features are used to build high-level features as network gets deeper
- frame-based classification

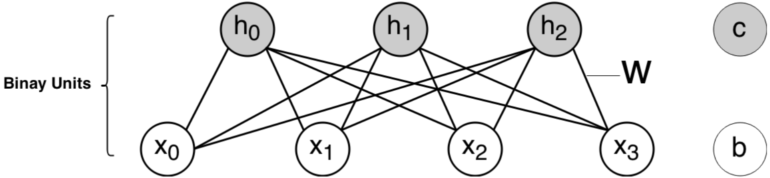
Table of Contents

Concepts

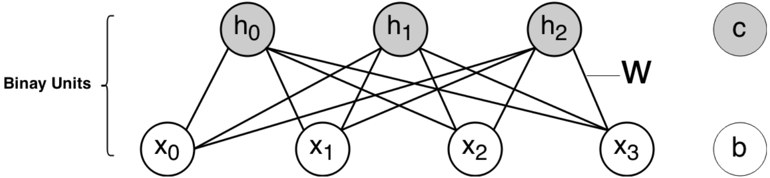
- Generative graphical model, capture data distrbution $P(\mathbf{x}|\boldsymbol{\theta})$
- Trained in unsupervised way, only use unlabeled input sequence$\mathbf{x}$ for learning.
  - automatically extract useful features from data
  - Find hidden structure (distribution).
  - Learned features used for prediction or classification
- Successfully applied in motion capture (Graham W. Taylor, Geoffrey E. Hinton, 2006)
- Potential to be extend to capture temporal information

Restricted Boltzmann Machine

Structure



Structure



Energy Function: $E_{\boldsymbol{\theta}} = -\mathbf{x}^{\mathbf{T}}\mathbf{W}\mathbf{h} - \mathbf{b}^{\mathbf{T}}\mathbf{x} - \mathbf{c}^{\mathbf{T}}\mathbf{h}$

Joint Distribution: $P^{RBM}(\mathbf{x}, \mathbf{h}) = \frac{1}{Z} e^{-E_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{h})}$

Partition Function: $Z = \sum_{\mathbf{x},\mathbf{h}} e^{-E_{\boldsymbol{\theta}}(\mathbf{x},\mathbf{h})}$

Free Energy: $\mathcal{F}(\mathbf{x}) = -\log \sum_{h} e^{-E(\mathbf{x},\mathbf{h})}$

Inference

$$P(\mathbf{x}) = \sum_{\mathbf{h}} P(\mathbf{x}, \mathbf{h})$$

$$P(\mathbf{h}) = \sum_{\mathbf{x}} P(\mathbf{x}, \mathbf{h})$$

Inference

$$P(\mathbf{x}) = \sum_{\mathbf{h}} P(\mathbf{x}, \mathbf{h})$$

$$P(\mathbf{h}) = \sum_{\mathbf{x}} P(\mathbf{x}, \mathbf{h})$$

$$P(\mathbf{h}|\mathbf{x}) = \frac{P(\mathbf{x}, \mathbf{h})}{P(\mathbf{x})}$$

$$P(\mathbf{x}|\mathbf{h}) = \frac{P(\mathbf{x}, \mathbf{h})}{P(\mathbf{h})}$$

Inference

$$P(\mathbf{x}) = \sum_{\mathbf{h}} P(\mathbf{x}, \mathbf{h})$$

$$P(\mathbf{h}) = \sum_{\mathbf{x}} P(\mathbf{x}, \mathbf{h})$$

$$P(\mathbf{h}|\mathbf{x}) = \frac{P(\mathbf{x}, \mathbf{h})}{P(\mathbf{x})}$$

$$P(\mathbf{x}|\mathbf{h}) = \frac{P(\mathbf{x}, \mathbf{h})}{P(\mathbf{h})}$$

$$P(h_j = 1 \mid \mathbf{x}) = sigmoid(\sum_i x_i W_{ij} + c_j)$$

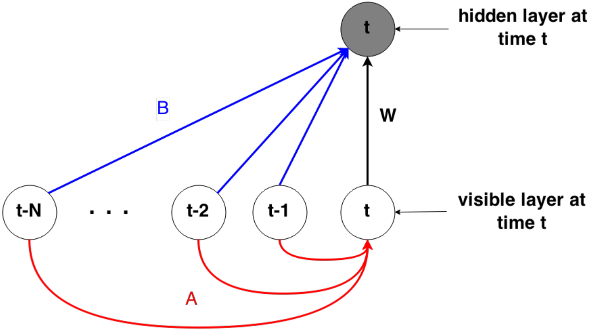
$$P(x_i = 1 \mid \mathbf{h}) = sigmoid(\sum_j W_{ij} h_j + b_i)$$

Conditional RBM

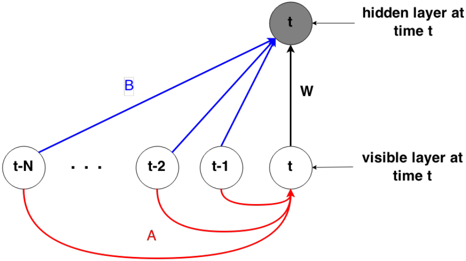
- Linear input units with independent Gaussian noise
- Real-valued data, e.g. spectral features

Conditional RBM

- Linear input units with independent Gaussian noise
- Real-valued data, e.g. spectral features



Conditional RBM



Energy Function: $E_{\boldsymbol{\theta}}^{CRBM}(\mathbf{x},\mathbf{h}) = \left\| \frac{\mathbf{x}-\tilde{\mathbf{b}}}{2} \right\|^2 - \tilde{\mathbf{c}}^T\mathbf{h} - \mathbf{x}^T\mathbf{W}\mathbf{h}$

Free Energy: $\mathcal{F}(\mathbf{x}) = \left\| \mathbf{x}-\tilde{\mathbf{b}} \right\|^2 - \log(1+e^{\tilde{\mathbf{c}}+\mathbf{x}\cdot\mathbf{W}})$

$$\tilde{\mathbf{b}} = \mathbf{b} + \mathbf{A}\cdot\mathbf{x}_{<t}$$

$$\tilde{\mathbf{c}} = \mathbf{c} + \mathbf{B}\cdot\mathbf{x}_{<t}$$

$$\boldsymbol{\theta} = \{\mathbf{W},\mathbf{A},\mathbf{B},\mathbf{b},\mathbf{c}\}$$

Conditional RBM

Energy Function: $E_{\boldsymbol{\theta}}^{CRBM}(\mathbf{x}, \mathbf{h}) = \left\| \frac{\mathbf{x} - \tilde{\mathbf{b}}}{2} \right\|^2 - \tilde{\mathbf{c}}^T \mathbf{h} - \mathbf{x}^T \mathbf{W} \mathbf{h}$

Free Energy: $\mathcal{F}(\mathbf{x}) = \left\| \mathbf{x} - \tilde{\mathbf{b}} \right\|^2 - \log(1 + e^{\tilde{\mathbf{c}} + \mathbf{x} \cdot \mathbf{W}})$

$$\tilde{\mathbf{b}} = \mathbf{b} + \mathbf{A} \cdot \mathbf{x}_{<t}$$

$$\tilde{\mathbf{c}} = \mathbf{c} + \mathbf{B} \cdot \mathbf{x}_{<t}$$

$$\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{A}, \mathbf{B}, \mathbf{b}, \mathbf{c}\}$$

Training of Energy-based Model

Optimization Method: Maximum Likelihood

$$P(\mathbf{x}) = \frac{e^{-\mathcal{F}(\mathbf{x})}}{Z}$$

$$-\frac{\partial \log P(\mathbf{x})}{\partial \boldsymbol{\theta}} = \frac{\partial \mathcal{F}(\mathbf{x})}{\partial \boldsymbol{\theta}} - \sum_{\tilde{\mathbf{x}}} P(\tilde{\mathbf{x}}) \frac{\partial \mathcal{F}(\tilde{\mathbf{x}})}{\partial \boldsymbol{\theta}}$$

$$-\frac{\partial \log P(\mathbf{x})}{\partial \boldsymbol{\theta}} = \frac{\partial \mathcal{F}(\mathbf{x})}{\partial \boldsymbol{\theta}} - \frac{1}{|\mathcal{N}|} \sum_{\tilde{\mathbf{x}} \in \mathcal{N}} P(\tilde{\mathbf{x}}) \frac{\partial \mathcal{F}(\tilde{\mathbf{x}})}{\partial \boldsymbol{\theta}}$$

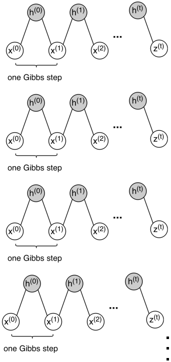
Training of Energy-based Model

Optimization Method: Maximum Likelihood

$$P(\mathbf{x}) = \frac{e^{-\mathcal{F}(\mathbf{x})}}{Z}$$

$$-\frac{\partial \log P(\mathbf{x})}{\partial \boldsymbol{\theta}} = \frac{\partial \mathcal{F}(\mathbf{x})}{\partial \boldsymbol{\theta}} - \sum_{\tilde{\mathbf{x}}} P(\tilde{\mathbf{x}}) \frac{\partial \mathcal{F}(\tilde{\mathbf{x}})}{\partial \boldsymbol{\theta}}$$

$$-\frac{\partial \log P(\mathbf{x})}{\partial \boldsymbol{\theta}} = \frac{\partial \mathcal{F}(\mathbf{x})}{\partial \boldsymbol{\theta}} - \frac{1}{|\mathcal{N}|} \sum_{\tilde{\mathbf{x}} \in \mathcal{N}} P(\tilde{\mathbf{x}}) \frac{\partial \mathcal{F}(\tilde{\mathbf{x}})}{\partial \boldsymbol{\theta}}$$



$$
\begin{aligned}
\mathbf{x_1} &\sim \hat{P}(\mathbf{x}) \\
\mathbf{h_1} &\sim \hat{P}(\mathbf{h}|\mathbf{x}_1) \\
\mathbf{x_2} &\sim \hat{P}(\mathbf{x}|\mathbf{h}_1) \\
\mathbf{h_2} &\sim \hat{P}(\mathbf{h}|\mathbf{x}_2) \\
&\vdots \\
\mathbf{x_{t+1}} &\sim \hat{P}(\mathbf{x}|\mathbf{h}_t)
\end{aligned}
\tag{1}
$$

$t = 1$, Gibbs step $\rightarrow$ Constrastive Divergence

Table of Contents

N-hidden layers neural network

- Hidden layer pre-activation:
  $\mathbf{a}(\mathbf{x}) = \mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}$
  $a_j(\mathbf{x}) = \sum_i w_{ji}^{(1)} x_i + b_j^{(1)}$
- Hidden layer activation:
  $\mathbf{h} = f(\mathbf{a})$
- Output layer activation:
  $\hat{y}(\mathbf{x}) =$
  $o(\mathbf{W}^{(N+1)}\mathbf{h}^{(N)} + \mathbf{b}^{(N+1)})$

Empirical Risk Minimization

- learning algorithms

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{M} \sum_{m} l(\hat{y}(\mathbf{x}^{(m)}; \boldsymbol{\theta}), y^{(m)}) + \lambda \Omega(\boldsymbol{\theta})$$

- loss function $l(\hat{y}(\mathbf{x}^{(m)}; \boldsymbol{\theta}), y^{(m)})$
  for sigmoid activation $l(\boldsymbol{\theta}) = \sum_m \frac{1}{2} \left\| y^{(m)} - \hat{y}^{(m)} \right\|^2$
- regularizer $\lambda \Omega(\boldsymbol{\theta})$

Optimization

- Gradient calculation with Backpropagation
- Stochastic/Mini-batch gradient descent

Vanishing Gradient

- Training time increases as network gets deeper
- Gradient shrink exponentially and training end up local minima
- Caused by random initialization of network parameters

Pre-training

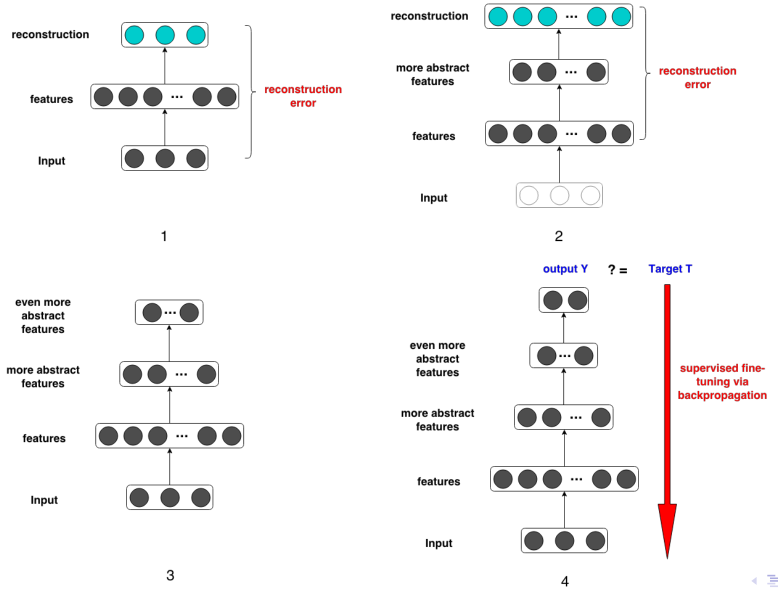
Vanishing Gradient

- Training time increases as network gets deeper
- Gradient shrink exponentially and training end up local minima
- Caused by random initialization of network parameters

Unsupervised layerwise pre-training

- Pretrain the deep network layer by layer to build a stacked auto-encoder
- Each layer is trained as a single hidden layer auto-encoder by minimizing average reconstruction error:
  $\min\ l_{AE} = \sum_m \frac{1}{2} \left\| \mathbf{x}^{(m)} - \hat{\mathbf{x}}^{(m)} \right\|^2$
- Fine-tuning the entire deep network with supervised training

Pre-training



Overfitting

- Huge amount of parameters in deep network
- Not enough data for training
- Poor generalization

Overfitting

- Huge amount of parameters in deep network
- Not enough data for training
- Poor generalization

Regularization

- Add weight penalization $\lambda \|\mathbf{w}\|_p$ to loss function

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{M} \sum_m l(\hat{y}(\mathbf{x}^{(m)}; \boldsymbol{\theta}), y^{(m)}) + \lambda \|\mathbf{w}\|_p$$

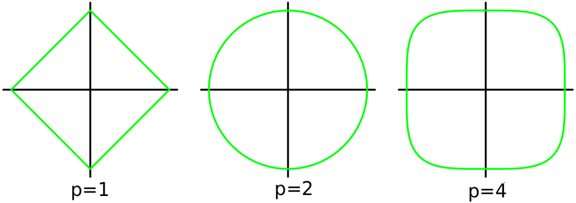
- In convex optimization:

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{M} \sum_m l(\hat{y}(\mathbf{x}^{(m)}; \boldsymbol{\theta}), y^{(m)}), s.t. \|\mathbf{w}\|_p \leq C$$

P-Norm

$$\|\mathbf{w}\|_p := \left(\sum_{n=1}^{n} |w_i|^p\right)^{1/p} = \sqrt[p]{|w_1|^p +, ..., +|w_n|^p}$$

Widely used: L1- and L2-regularization ($p = 1$ and $p = 2$)



P-Norm

$$\|\mathbf{w}\|_p := \left(\sum_{n=1}^{n} |w_i|^p\right)^{1/p} = \sqrt[p]{|w_1|^p +, \ldots, +|w_n|^p}$$

Widely used: L1- and L2-regularization ($p = 1$ and $p = 2$)

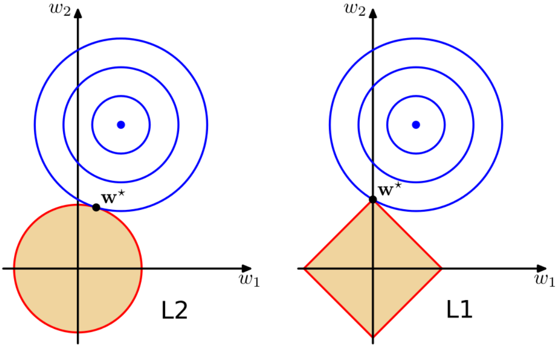


Table of Contents

Concepts of RNN

- modelling sequential data, emotion in speech .
- Same Structure as MLP but differs from feed-forward network, enabling nonlinear mapping
- Feedback connection between previous hidden units and current hidden units, enabling memory past hidden state.
- Potentially to model arbitary dynamic system.
- Trained with **b**ack**p**ropagation **t**hrough **t**ime (BPTT)

Concepts of RNN

- modelling sequential data, emotion in speech .
- Same Structure as MLP but differs from feed-forward network, enabling nonlinear mapping

$$h_t = \mathcal{H}(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$
$$y_t = W_{hy}h_t + b_y$$

- Feedback connection between previous hidden units and current hidden units, enabling memory past hidden state.
- Potentially to model arbitary dynamic system.
- Trained with **b**ack**p**ropagation **t**hrough **t**ime (BPTT)

Concepts of RNN

- modelling sequential data, emotion in speech .
- Same Structure as MLP but differs from feed-forward network, enabling nonlinear mapping
- Feedback connection between previous hidden units and current hidden units, enabling memory past hidden state.
- Potentially to model arbitary dynamic system.
- Trained with **b**ack**p**ropagation **t**hrough **t**ime (BPTT)

Concepts of RNN

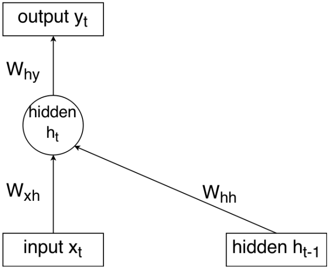
- modelling sequential data, emotion in speech .
- Same Structure as MLP but differs from feed-forward network, enabling nonlinear mapping
- Feedback connection between previous hidden units and current hidden units, enabling memory past hidden state.
- Potentially to model arbitary dynamic system.
- Trained with **b**ack**p**ropagation **t**hrough **t**ime (BPTT)

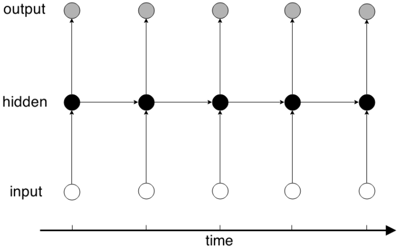
Concepts of RNN

- modelling sequential data, emotion in speech .
- Same Structure as MLP but differs from feed-forward network, enabling nonlinear mapping
- Feedback connection between previous hidden units and current hidden units, enabling memory past hidden state.
- Potentially to model arbitary dynamic system.
- Trained with **b**ack**p**ropagation **t**hrough **t**ime (BPTT)

Recurrent Neural Network

Concepts of RNN





From RNN to LSTM

Problems with RNN

- gradient vanishing during backpropagation as time steps increases ($>$100)
- difficult to capture long-time dependency (which is required in emotion recognition)

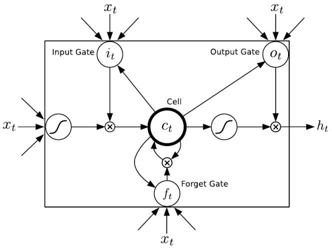
Solutions

-

Long short term memory

S. Hochreiter and J. Schmidhuber, Lovol. 9, pp. 1735-1780, 1997.

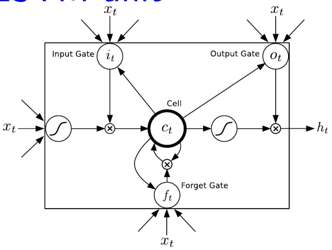
LSTM unit



$$
\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \\
c_t &= f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \\
h_t &= o_t \tanh(c_t)
\end{aligned}
$$

Long short term memory

LSTM unit



$$
\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \\
c_t &= f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \\
h_t &= o_t \tanh(c_t)
\end{aligned}
$$

Features in LSTM

- gates are trained to learn when it shoud be open/closed.
- Constant Error Carousel
- preserve long-time dependency by maintaining gradient over time.

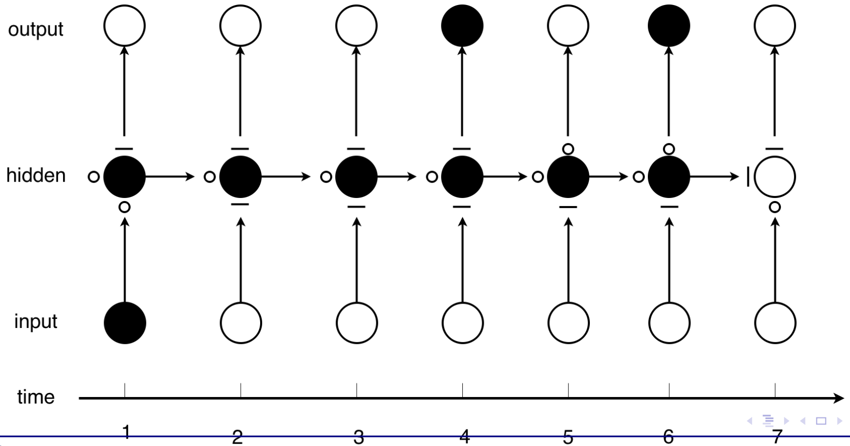


Table of Contents

Conclusion

- Model with long-term dependencies shall be used for speech emotion
- CRBM is appropriate for short-term modelling, but not for long-term variation
- LSTM is good at modelling long time dependency
- Frame-based classification can also reach good result
  - CRBM-LSTM 71.98%
  - LSTM 81.59%
  - LSTM with rectifier layers 83.43%

Outlook

- Stacking CRBM to form deeper structure
- Traing CRBM with more/larger data base
- Second order optimization to speed up learning process
- Bi-directional LSTM, capturing future dependencies

End

Thank You!