Deep Learning in MIR:
Demystifying the Dark Art

ISMIR Tutorial Session

Curitiba, Brazil
4 November 2013

Erik M. Schmidt
eschmidt@pandora.com

Philipe Hamel
hamelphi@google.com

Eric J. Humphrey
ejh333@nyu.edu

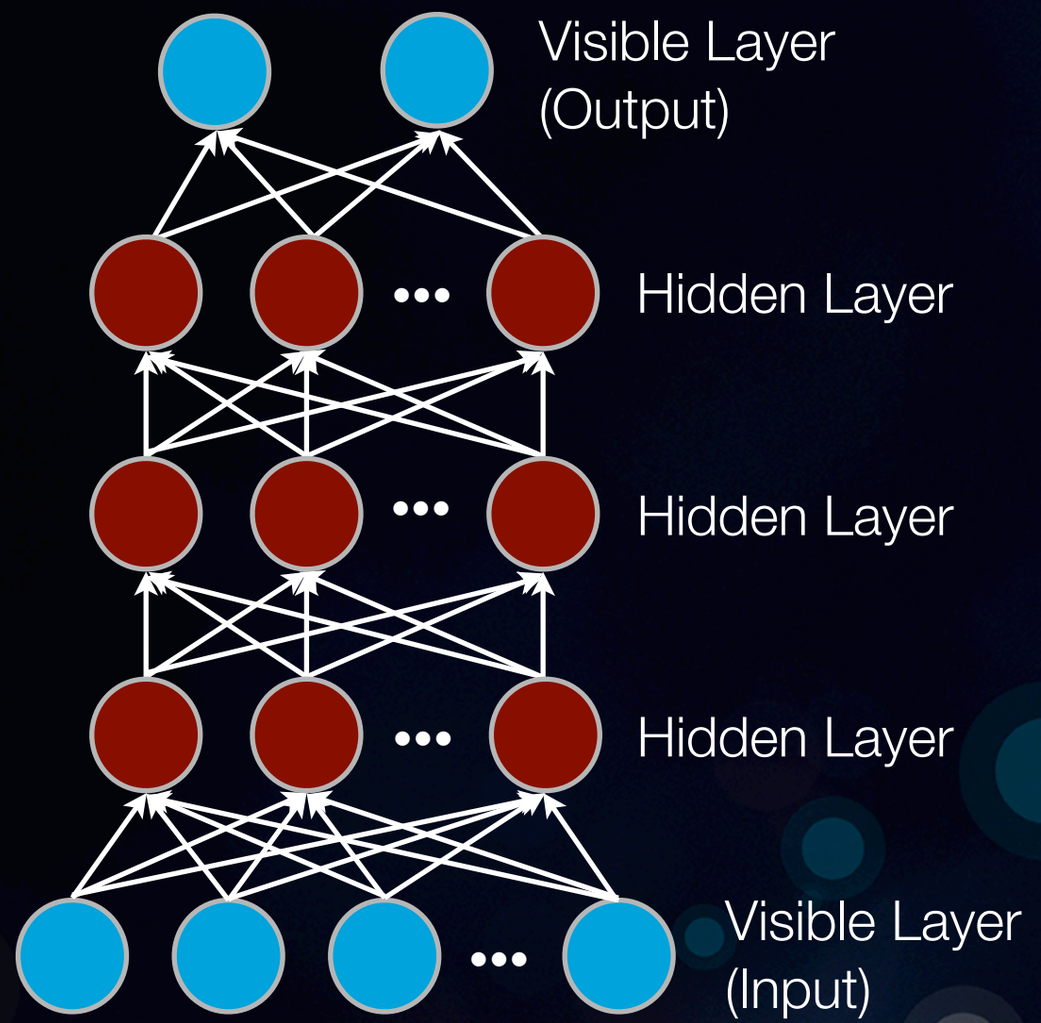
Deep Learning

What exactly does that mean?

- A deep learning model simply is one which contains multiple layers
- A layer can be any simple transformation:

$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b$$

- Generally contain non-linearities
- Many names: Artificial neural nets, multi-layer perceptrons, convolutional neural networks, deep belief networks, etc



Deep Learning in MIR

Tutorial Overview

Erik Schmidt
Pandora Media, Inc

- History of deep learning in AI
  - Perceptrons, neural nets, SVMs, convolutional neural nets, restricted Boltzman machines, greedy-wise pre-training, dropout
- Deep learning in music emotion recognition
  - Learning emotion-based representations from raw magnitude spectra
  - Learning melody and rhythm features with deep belief networks

Deep Learning in MIR

Tutorial Overview

Philippe Hamel
Google, Inc

- The state-of-the-art in deep learning
  - Pre-training vs. initialization
  - Dropout
  - Activation functions
  - Structure (convolution, recurrence)
- Deep learning in MIR

Deep Learning in MIR

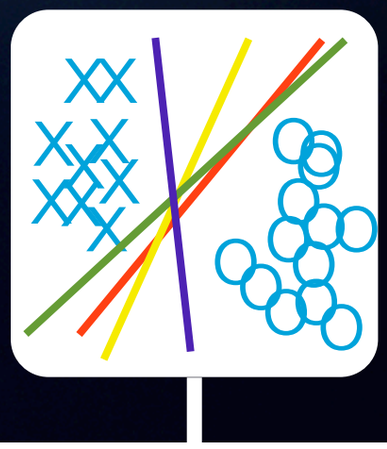
Tutorial Overview

Eric Humphrey
NYU MARL

- Deep learning through a signal processing lens
  - What “is” deep learning
  - How you are already doing it (kinda)
  - How you might do it intentionally
  - Some tips, tricks, insight, and advice
  - A few thoughts for the future

Deep Learning

A long, long time ago...



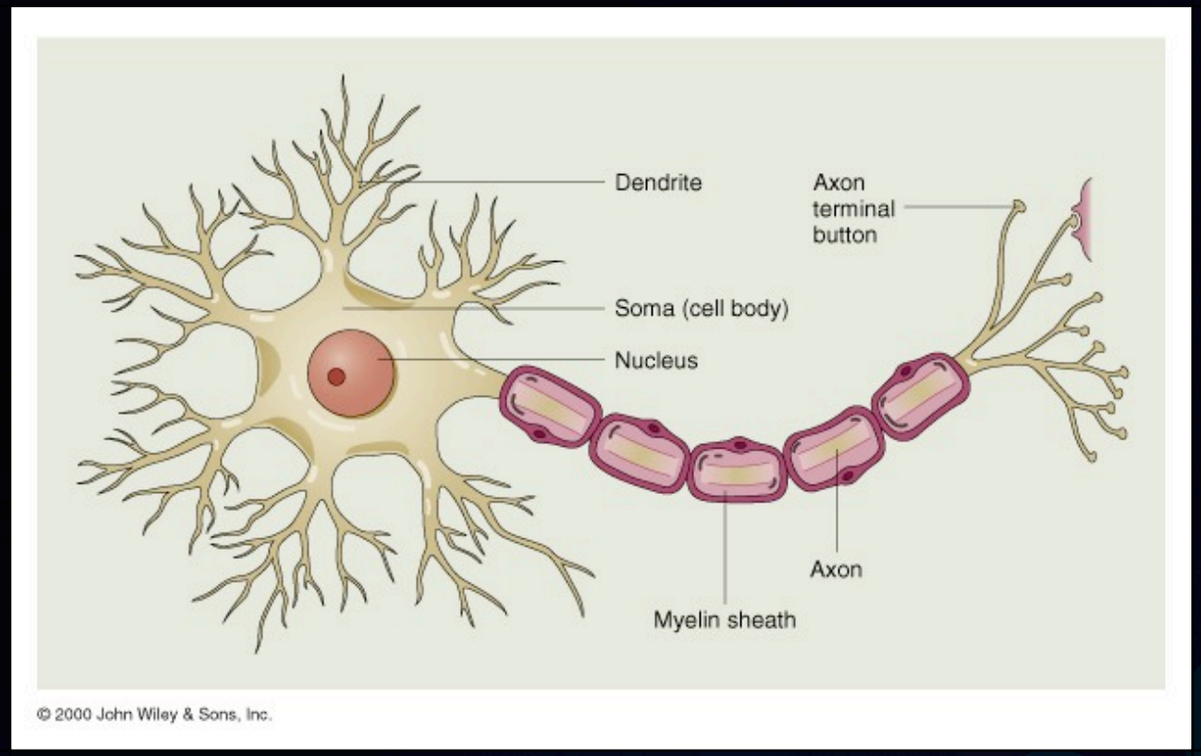
1959

Perceptrons

Perceptrons

Neural inspired models

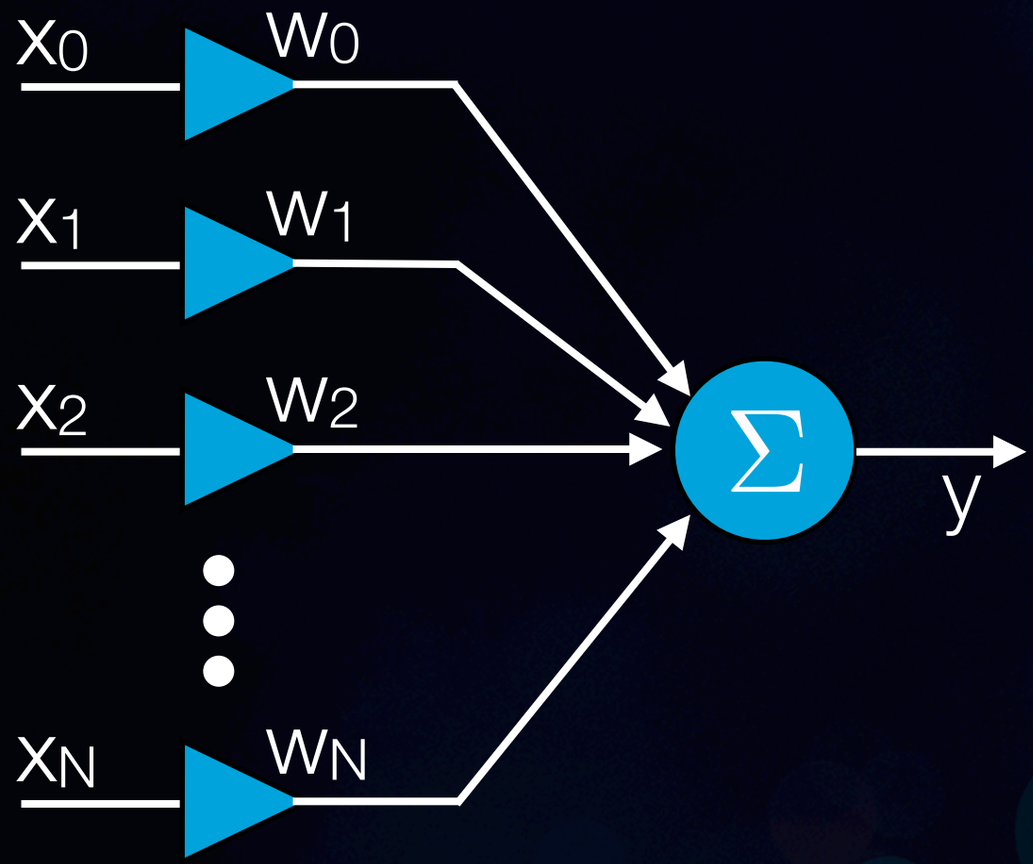
- Dendrite: input terminals
- Cell body: signals combined
- Axon: output cable
- Synapse: interface to other neurons



Perceptrons

Neural inspired models

- The perceptron is a model inspired by the neuron
- One of the earliest machine learning models
- Later we will see these as a building block for deep learning



$$f(x) = x_0 w_0 + ... + x_N w_N$$

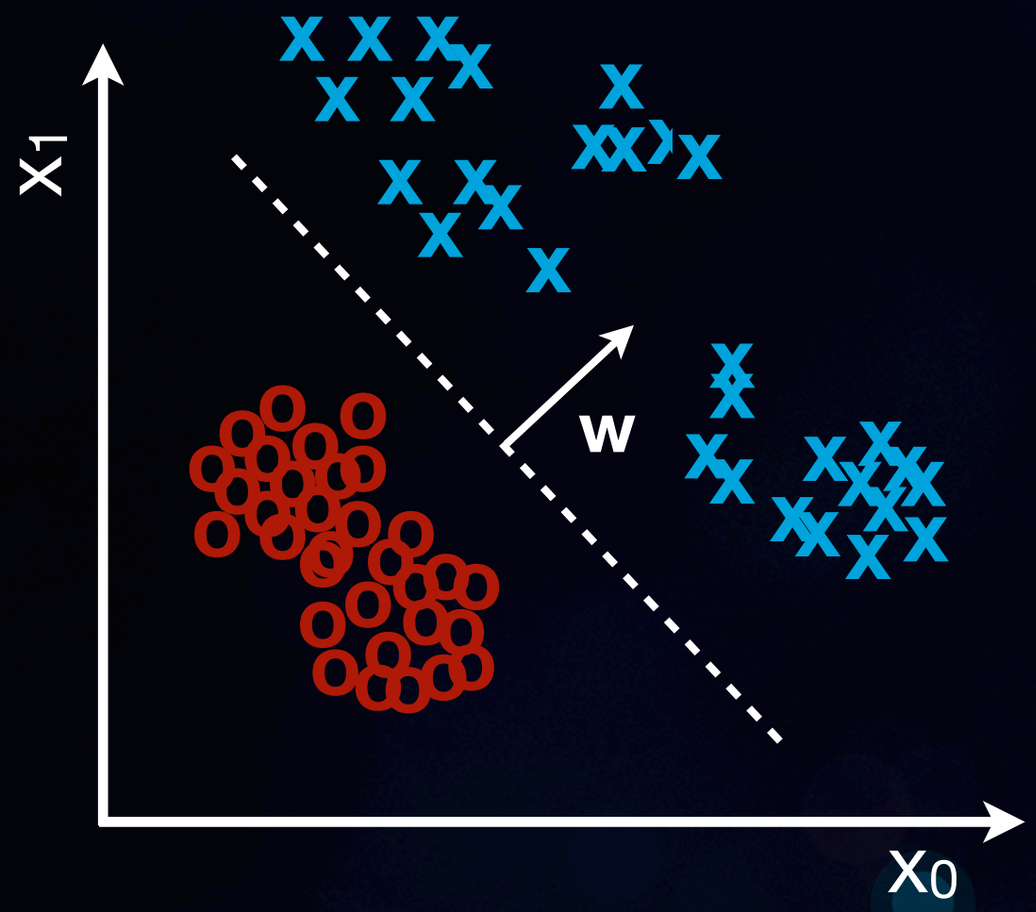
F. Rosenblatt (1957). The Perceptron--a perceiving and recognizing automaton. *Cornell Report 85-460-1*.

Perceptrons

Neural inspired models

- Include intercept term b
- Use inner product notation
- Decision function that's very easy to evaluate on a computer

$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b$$



Perceptron Algorithm

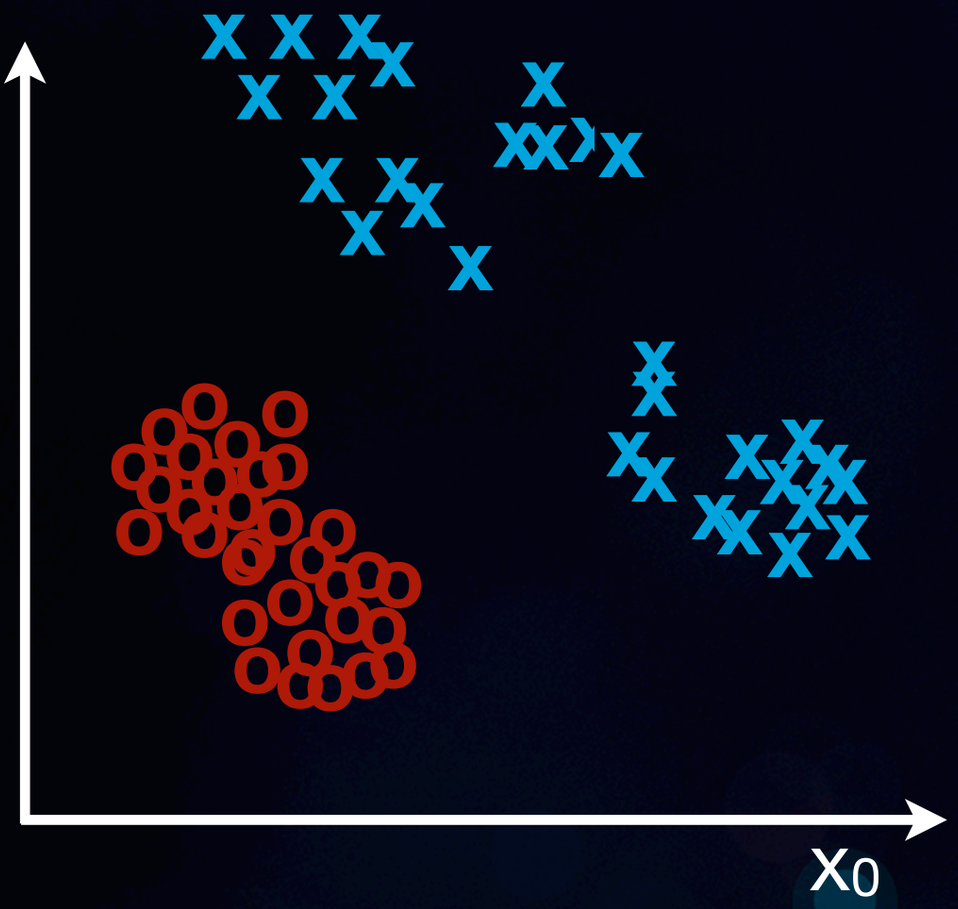
Fitting the model to data

- Input data:
  - Training data **x**
  - Labels y = +/-1
- Learning Objective:

$$\text{if } y_i = 1$$
$$\langle \mathbf{w}, \mathbf{x}_i \rangle + b > 0$$

$$\text{if } y_i = -1$$
$$\langle \mathbf{w}, \mathbf{x}_i \rangle + b < 0$$

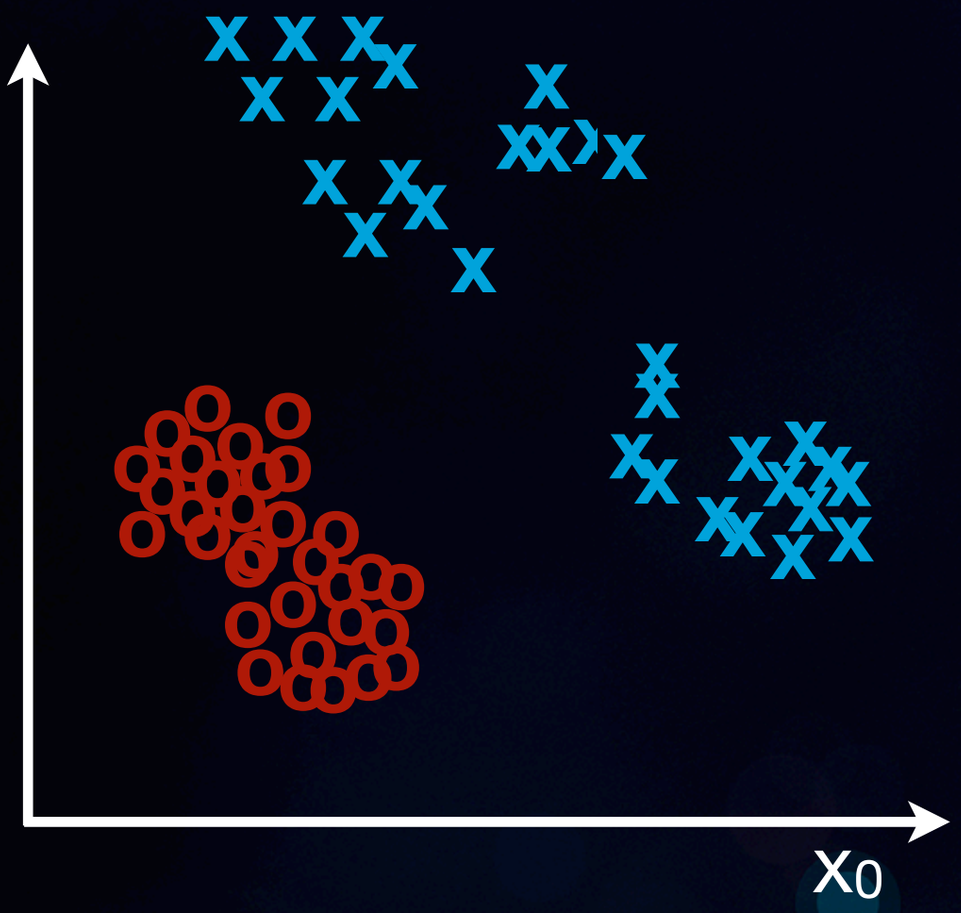
$$\text{for } \forall y$$
$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) > 0$$



Perceptron Algorithm

Fitting the model to data

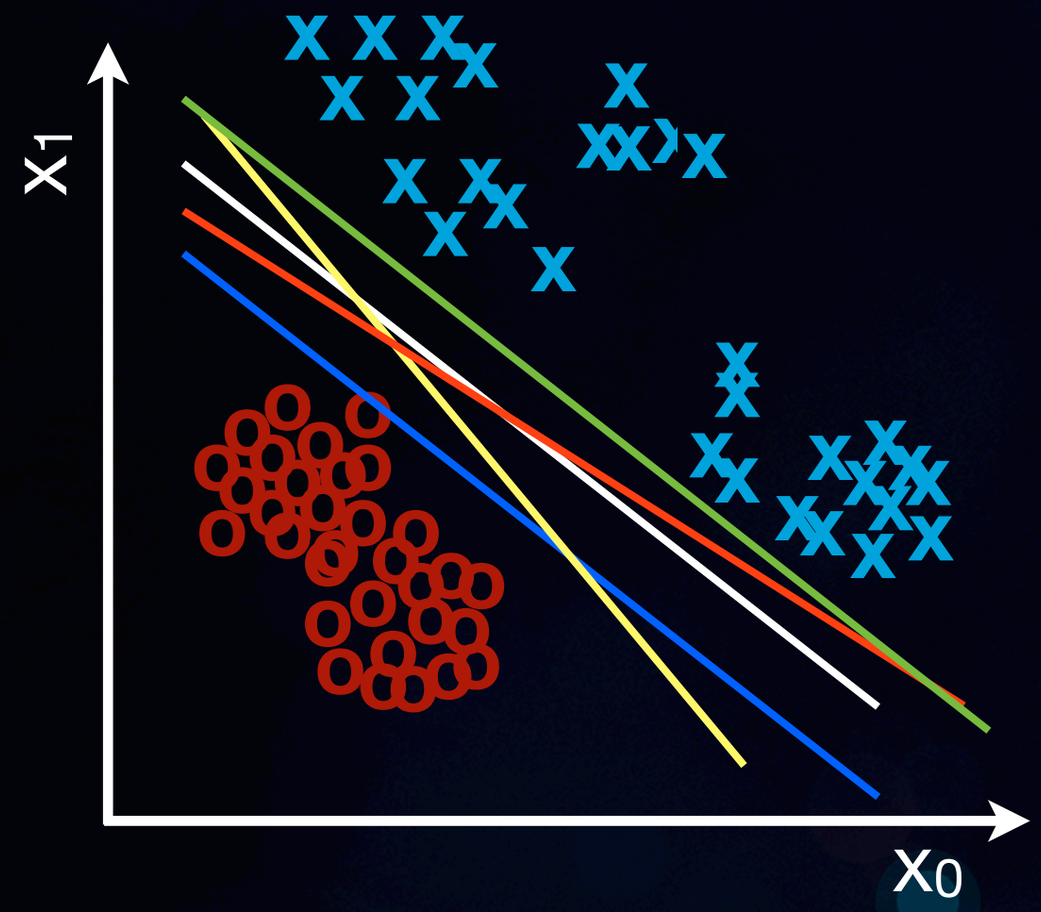
initialize w, b = 0
  repeat
    Pick (x_i, y_i) from data
    if $y_i(\langle w, x_i \rangle + b) \leq 0$ then
      $w' = w + y_i x_i$
      $b' = b + y_i$
until $y_i(\langle w, x_i \rangle + b) > 0$ for $\forall_i$



Perceptron Algorithm

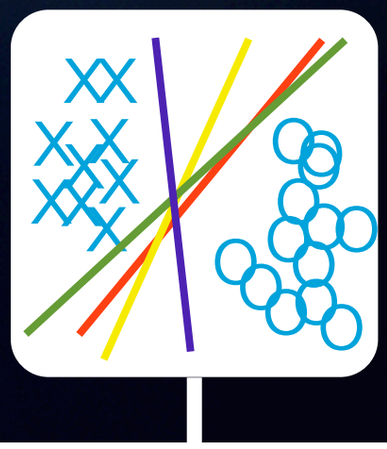
Fitting the model to data

initialize w, b = 0
  repeat
    Pick (x_i, y_i) from data
    if $y_i(\langle w, x_i \rangle + b) \leq 0$ then
      $w' = w + y_i x_i$
      $b' = b + y_i$
  until $y_i(\langle w, x_i \rangle + b) > 0$ for $\forall_i$



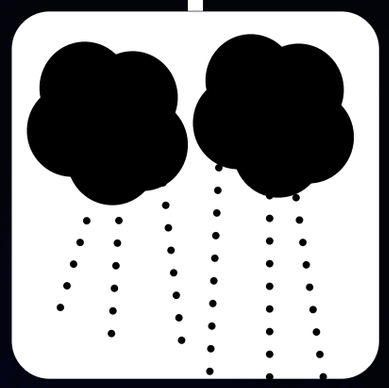
Deep Learning

A long, long time ago...



AI Winter
1969

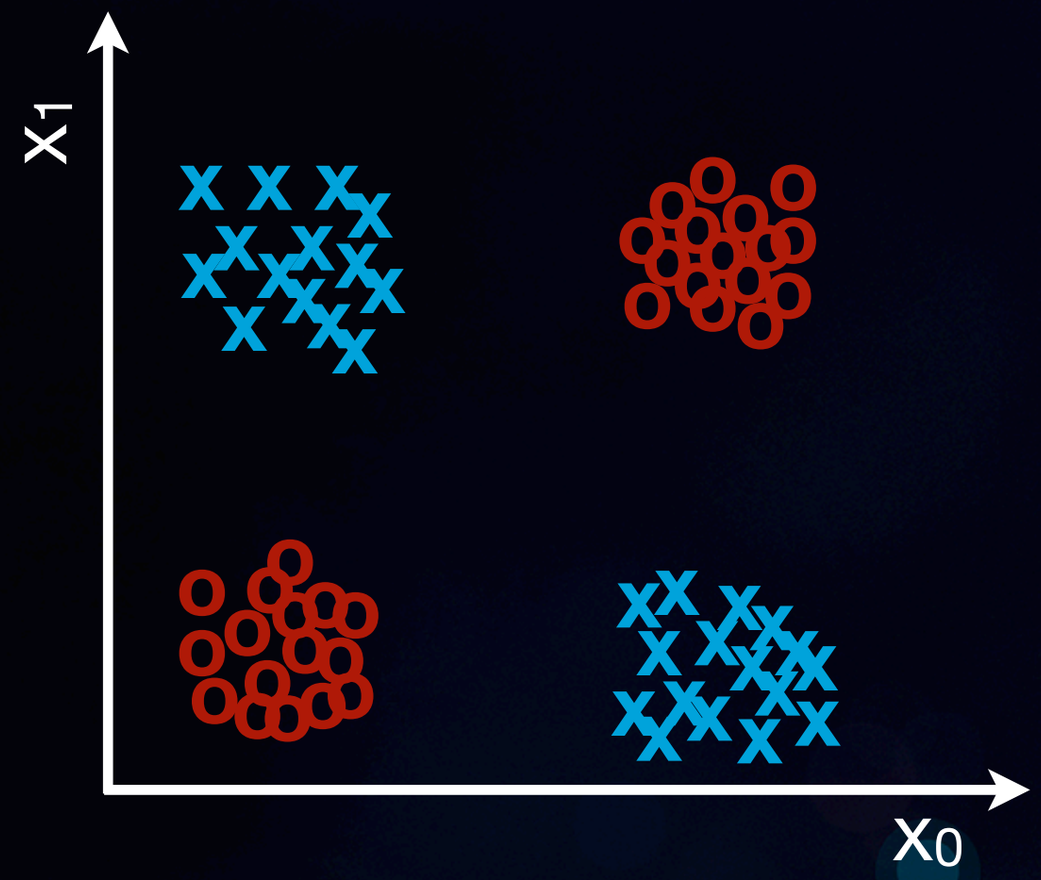
1959
Perceptrons



XOR Problem

A major limitation

- The perceptron algorithm can only learn a linear discriminant
- The inability to learn an XOR function was demonstrated by Marvin Minsky

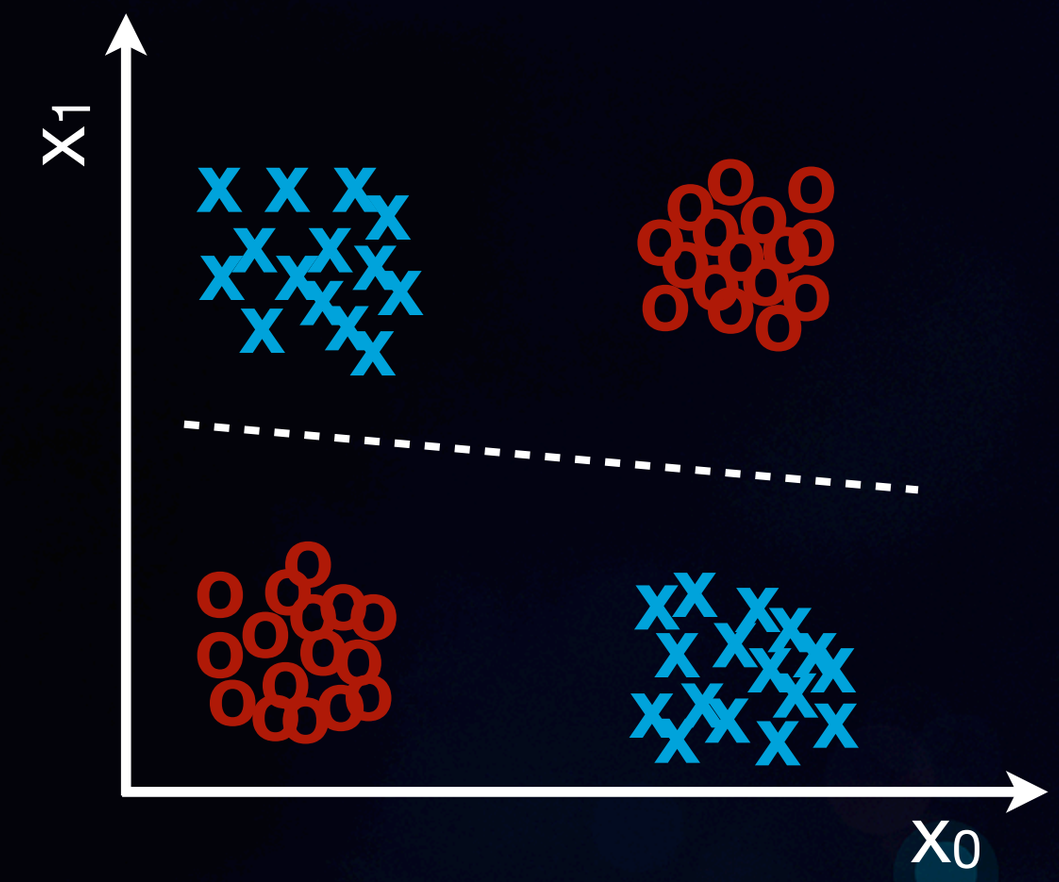


M. L. Minsky and S. A. Papert (1969). *Perceptrons*. Cambridge, MA: MIT Press.

XOR Problem

A major limitation

- The perceptron algorithm can only learn a linear discriminant
- The inability to learn an XOR function was demonstrated by Marvin Minsky

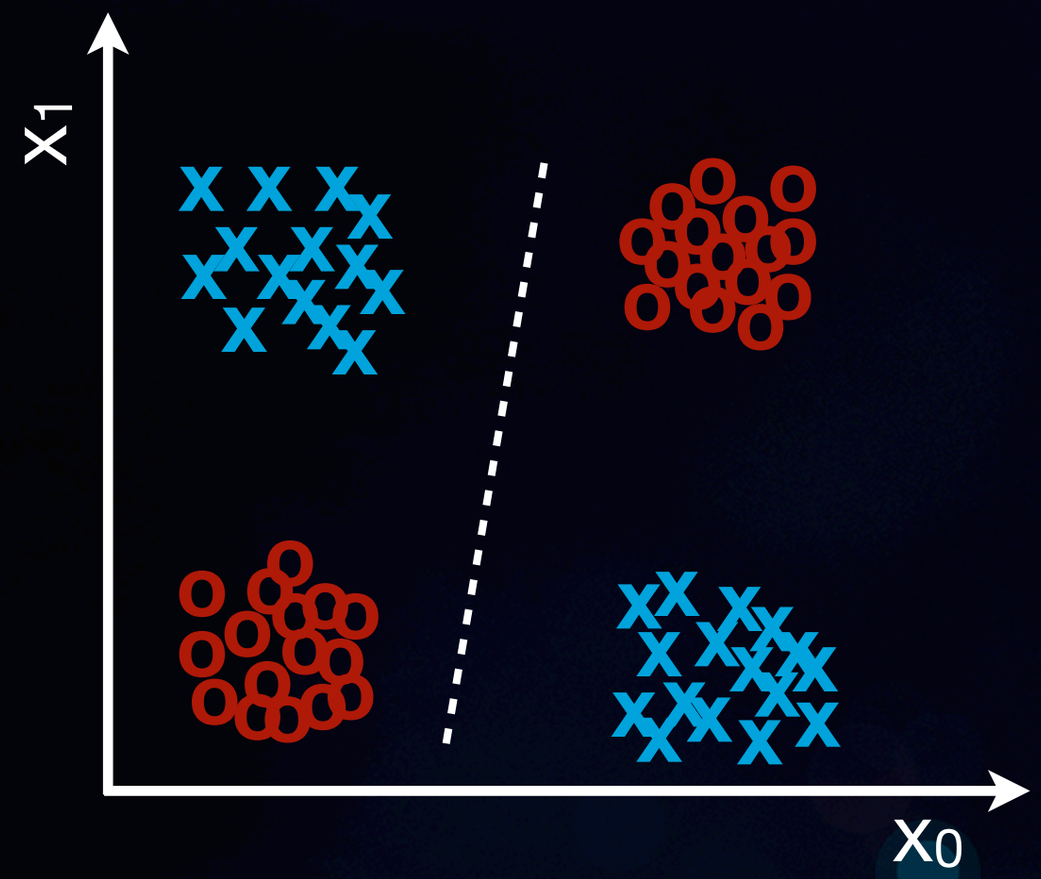


M. L. Minsky and S. A. Papert (1969). *Perceptrons*. Cambridge, MA: MIT Press.

XOR Problem

A major limitation

- The perceptron algorithm can only learn a linear discriminant
- The inability to learn an XOR function was demonstrated by Marvin Minsky

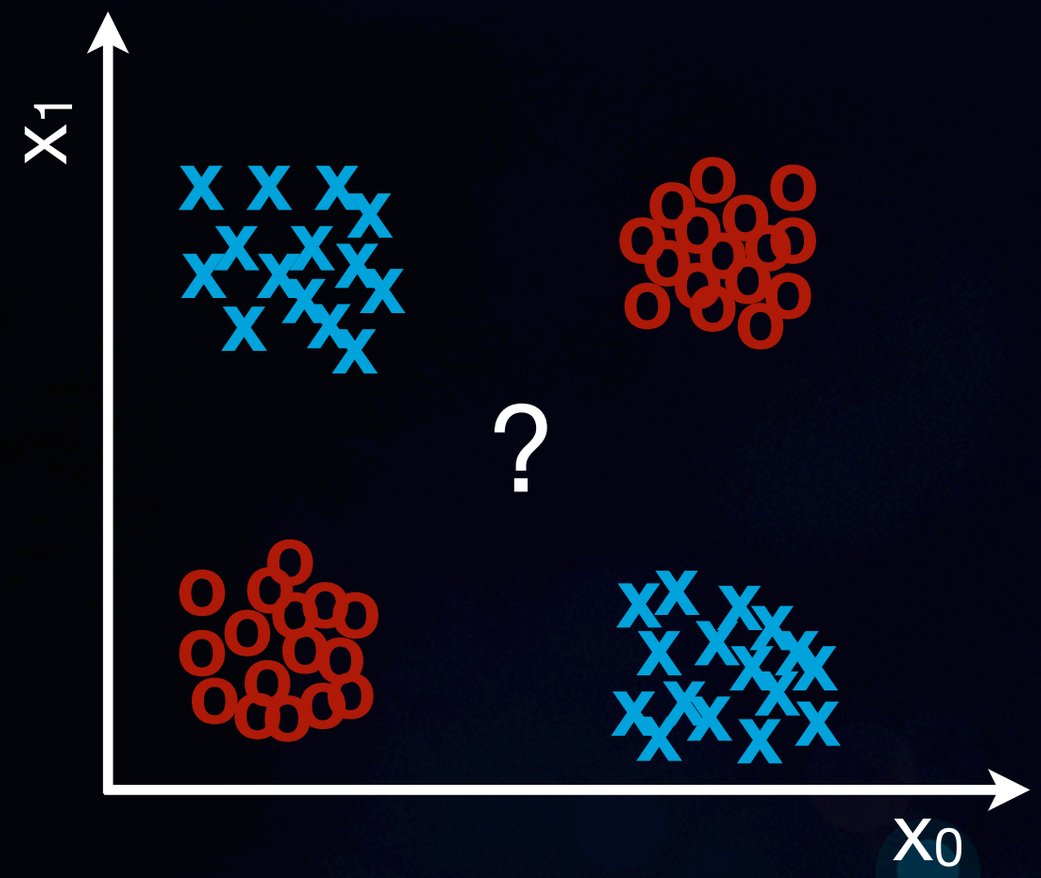


M. L. Minsky and S. A. Papert (1969). *Perceptrons*. Cambridge, MA: MIT Press.

XOR Problem

A major limitation

- The perceptron algorithm can only learn a linear discriminant
- The inability to learn an XOR function was demonstrated by Marvin Minsky

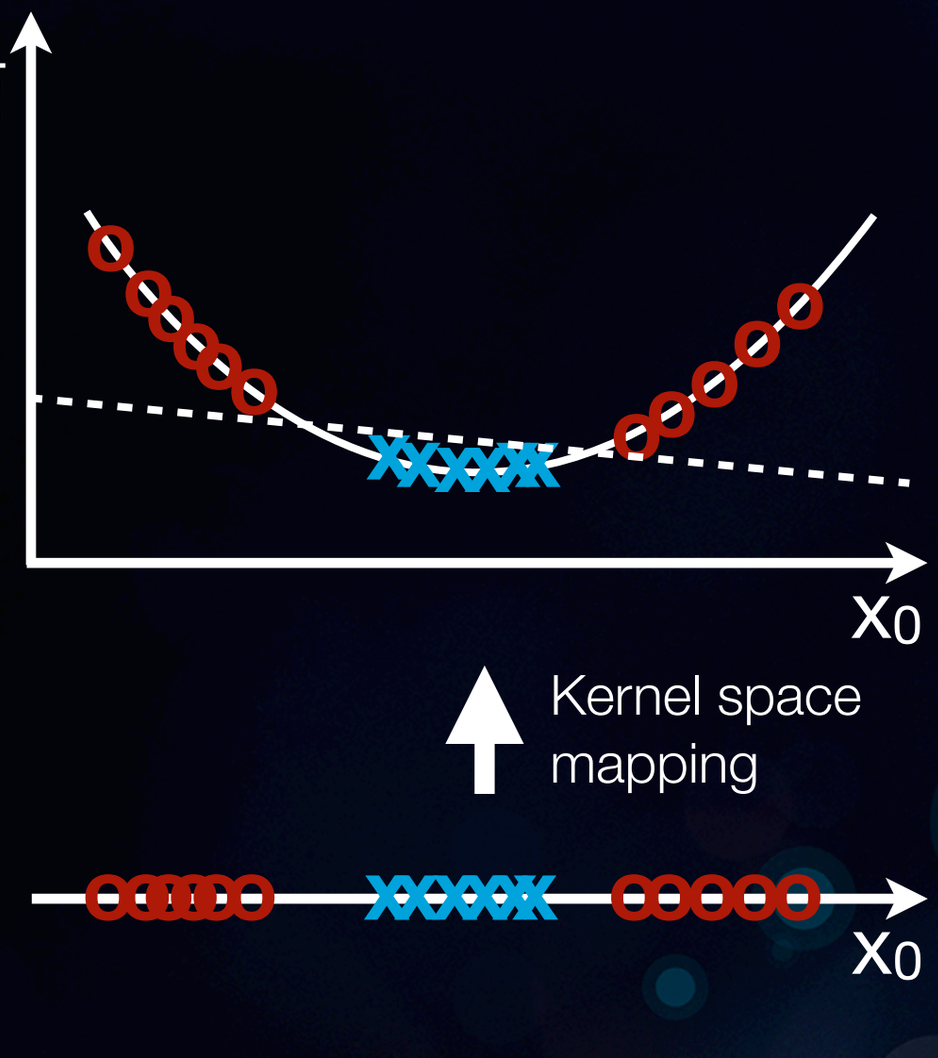


M. L. Minsky and S. A. Papert (1969). *Perceptrons*. Cambridge, MA: MIT Press.

Kernel Trick

Solving non-linear problems

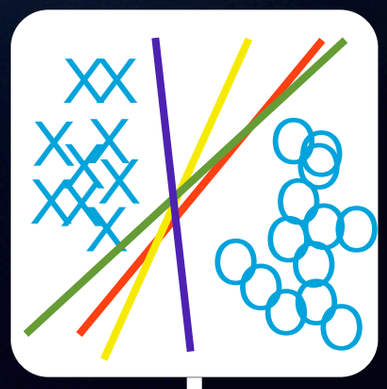
- Project data into higher dimensional space
- Employ non-linear transformation
- Obtain linear separability



M. Aizerman, E. Braverman, and L. Rozonoer (1967). Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*.

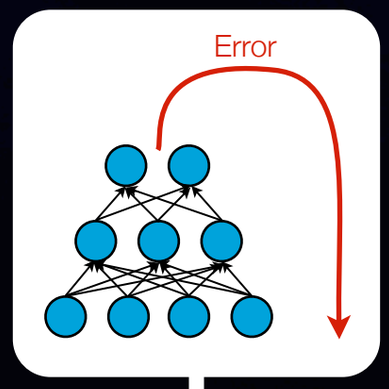
Deep Learning

A long, long time ago...

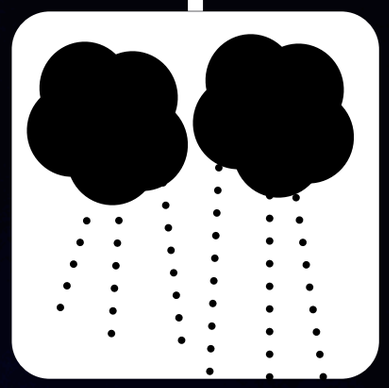


1959
Perceptrons

AI Winter
1969



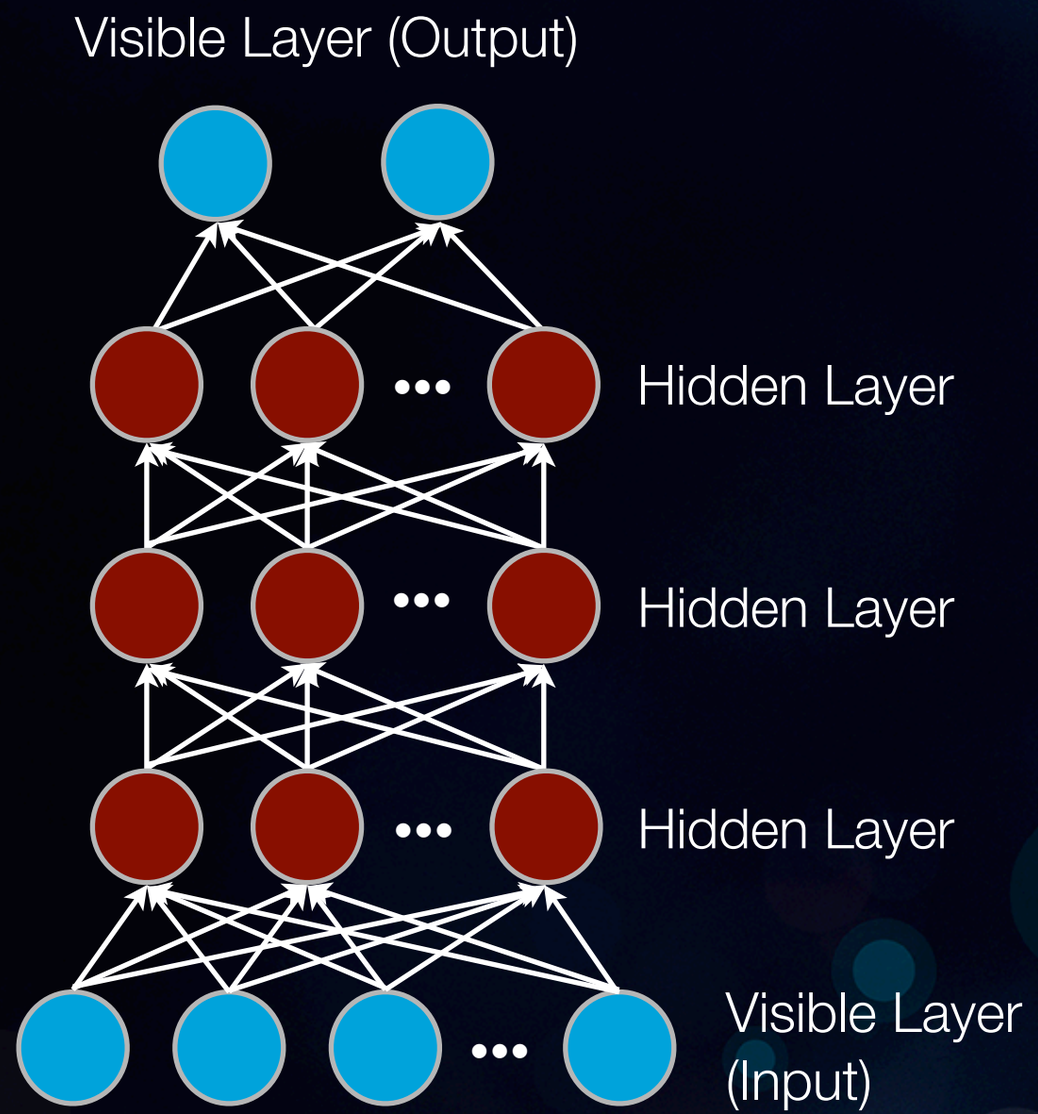
1986
Artificial Neural
Networks



Artificial Neural Networks

Adding in multiple layers

- In the 1980s researchers began making progress on deeper representations
- Build up larger models inspired by neural interaction in human brain

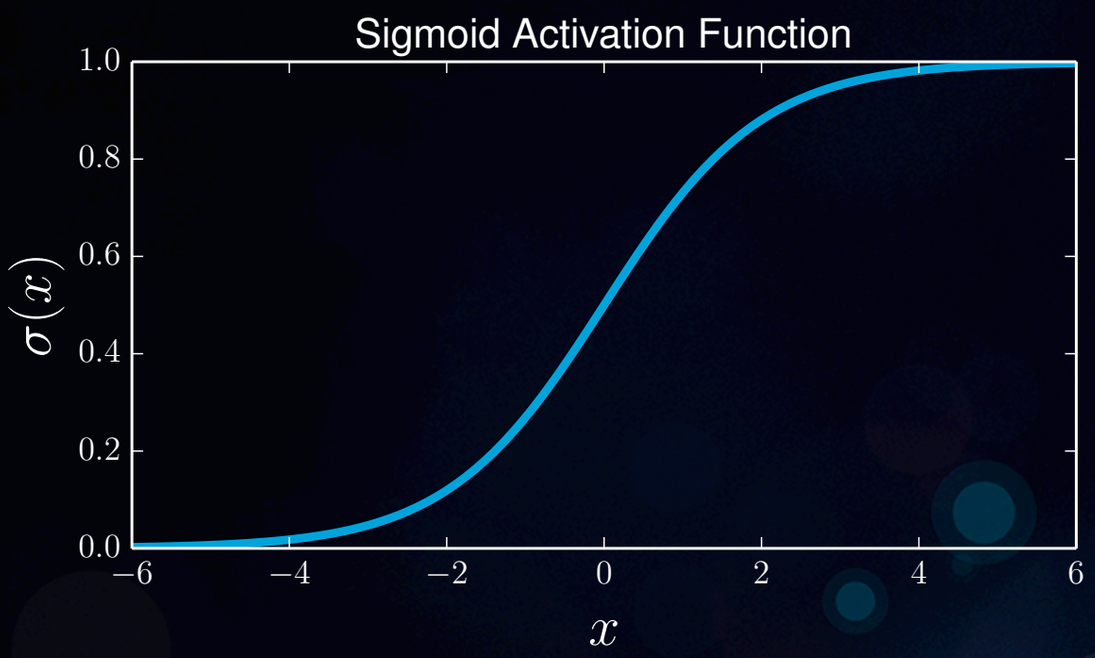


Artificial Neural Networks

Adding in multiple layers

- Each layer output applies a non-linearity or “activation”
- Most common is sigmoid activation
- Allows learning of non-linear relationships
- Without non-linearities deep network could be represented in a single layer

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



How Do We Train?

These models are complicated!

1-layer Model:

$$f(\mathbf{x}) = \sigma\left(\langle \mathbf{w_0}, \mathbf{x} \rangle + b_0\right)$$

2 parameters to fit...

2-layer Model:

$$f(\mathbf{x}) = \sigma\left(\langle \mathbf{w_1}, \sigma\left(\langle \mathbf{w_0}, \mathbf{x} \rangle + b_0\right) \rangle + b_1\right)$$

4 parameters to fit...

3-layer Model:

$$f(\mathbf{x}) = \sigma\left(\langle \mathbf{w_2}, \sigma\left(\langle \mathbf{w_1}, \sigma\left(\langle \mathbf{w_0}, \mathbf{x} \rangle + b_0\right) \rangle + b_1\right) \rangle + b_2\right)$$

6 parameters to fit!!!

Backpropagation

An algorithm for deep neural nets

Simple 1-layer Model:

$$\hat{y} = \sigma\left(\langle \mathbf{w}, \mathbf{x} \rangle + b\right)$$

Simple cost function:

$$\text{cost} = \frac{1}{N} \sum_i \left(y_i - \hat{y}_i\right)^2$$

Compute partials:

$$\partial \mathbf{w} = \frac{\partial}{\partial \mathbf{w}} \text{cost} = \frac{\partial}{\partial \mathbf{w}} \frac{1}{N} \sum_i \left(y_i - \hat{y}_i\right)^2$$

$$\partial b = \frac{\partial}{\partial b} \text{cost} = \frac{\partial}{\partial b} \frac{1}{N} \sum_i \left(y_i - \hat{y}_i\right)^2$$

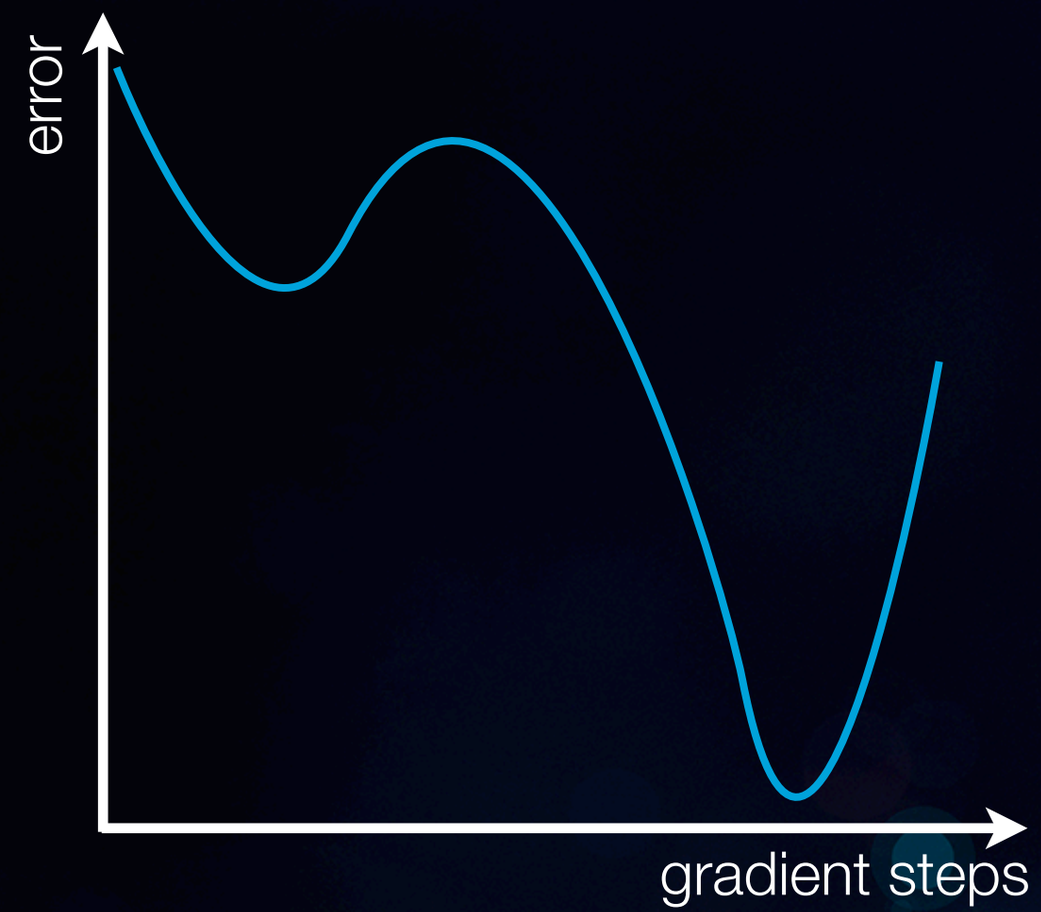
initialize θ (small random values)
  repeat
    Pick (x_i, y_i) from data
    Compute model error
    Compute: $\Delta(\theta^{(t)})$
    Update: $\theta^{(t+1)} = \theta^{(t)} - \varepsilon \, \Delta E(\theta^{(t)})$
until stopping criteria satisfied

D. E. Rumelhart, G. E. Hinton, and R. J. Williams (1986). Learning representations by backpropagating errors. *Nature.*

Backpropagation

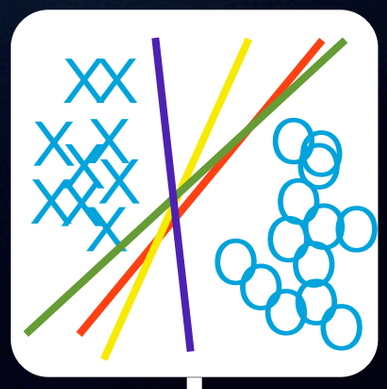
Issues

- Optimization guarantees a local minima, not global
- Small datasets produce noisy gradients
    - Lots of local minima
- Limited computational resources



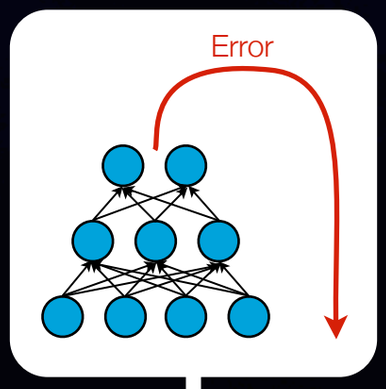
Deep Learning

A long, long time ago...



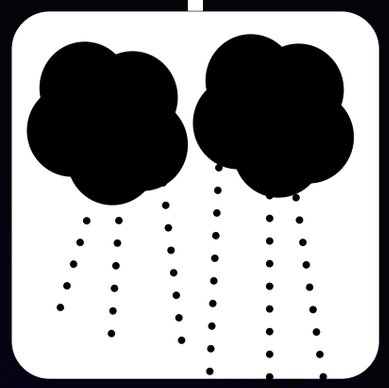
1959
Perceptrons

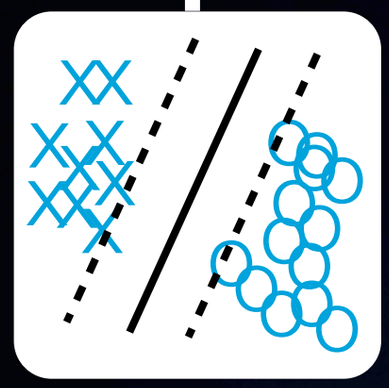
AI Winter
1969



1986
Artificial Neural
Networks

Support Vector
Machines
1995





Support Vector Machines

A free lunch?

New constraints:

if $y_i = 1$

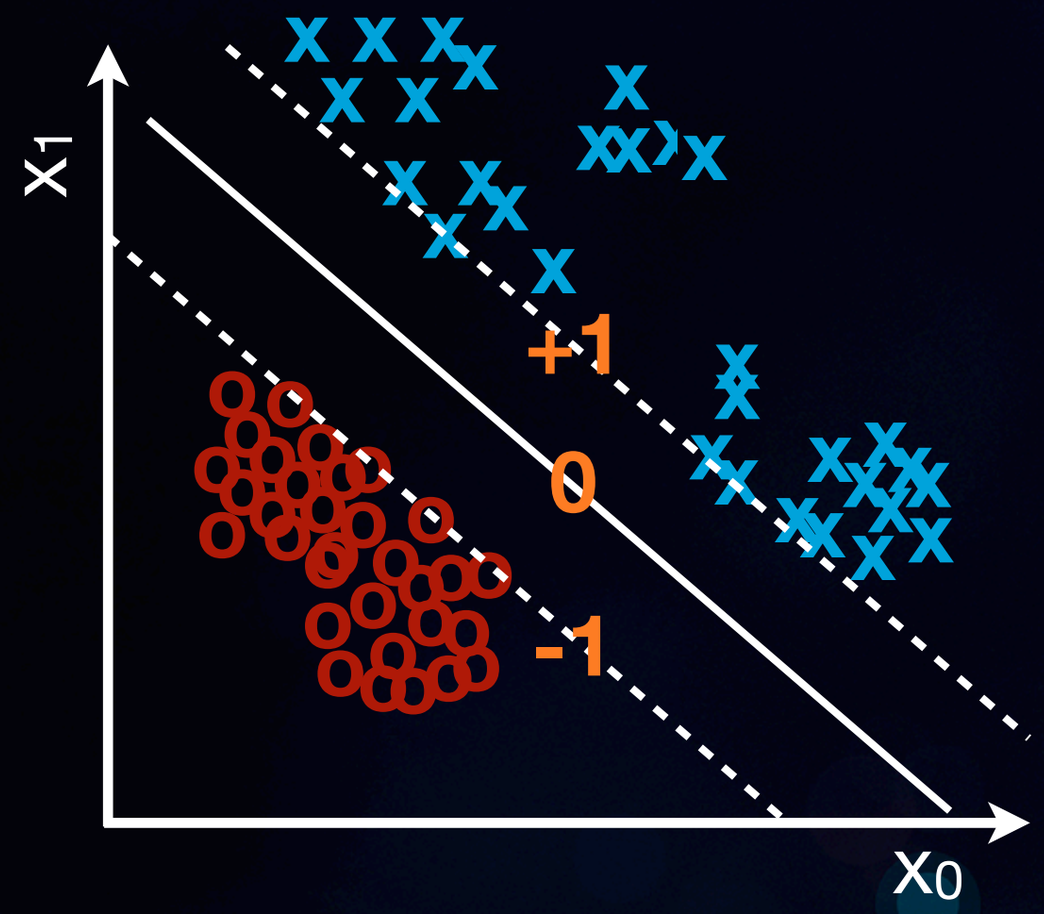
$$\langle \mathbf{w}, \mathbf{x}_i \rangle + b \geq 1$$

if $y_i = -1$

$$\langle \mathbf{w}, \mathbf{x}_i \rangle + b \leq -1$$

for $\forall y$

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$$



Optimization Function:

$$\text{minimize } \frac{1}{2}||\mathbf{w}||^2$$

$$\text{subject to } y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1 \geq 0 \text{ for all } 1 \leq i \leq m$$

C. Cortes and V. Vapnik (1995). Support Vector Networks. *Machine Learning*.

Support Vector Machines

A free lunch?

New constraints:

if $y_i = 1$

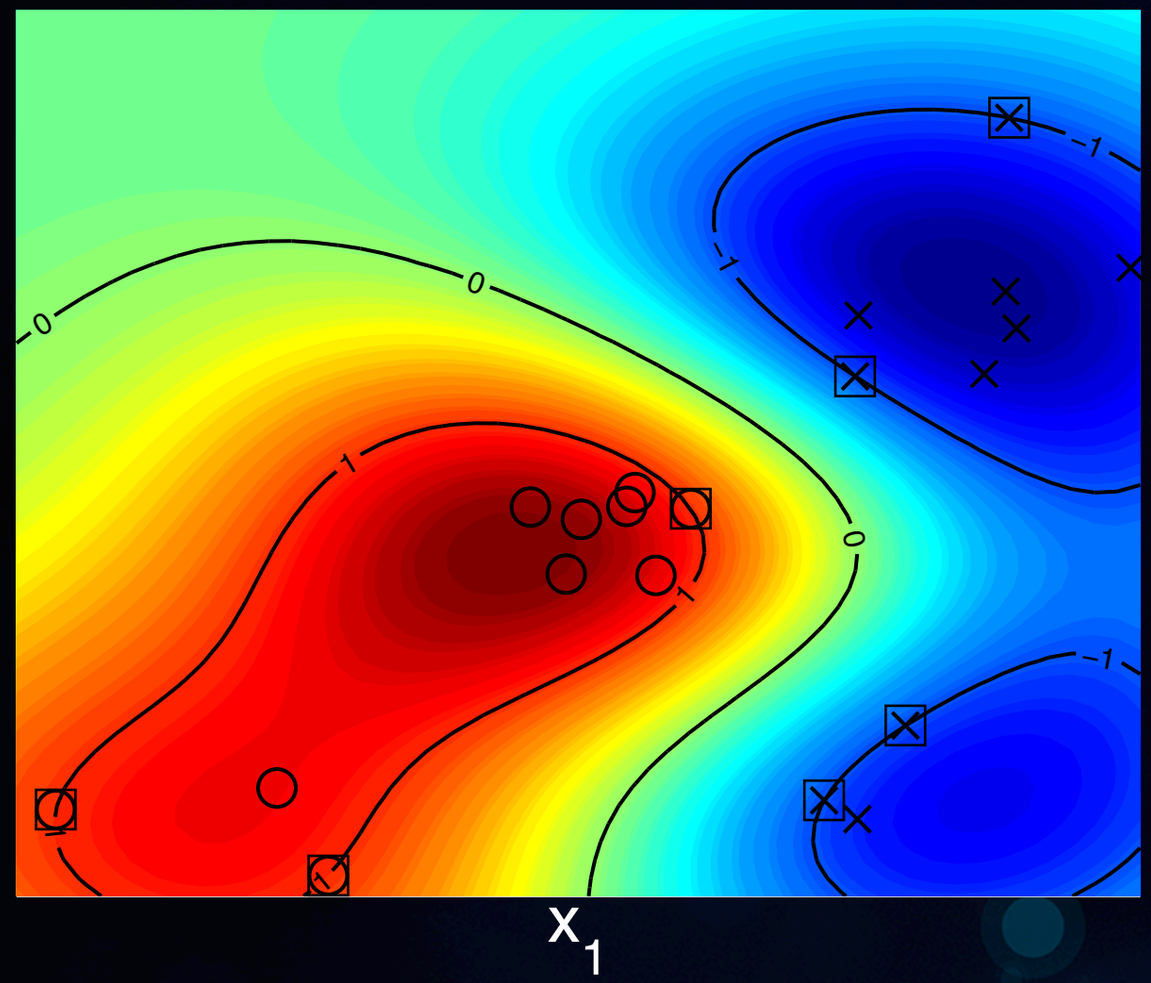
$$\langle \mathbf{w}, \mathbf{x}_i \rangle + b \geq 1$$

if $y_i = -1$

$$\langle \mathbf{w}, \mathbf{x}_i \rangle + b \leq -1$$

for $\forall y$

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$$



Optimization Function:

$$\text{minimize } \frac{1}{2}||\mathbf{w}||^2$$

$$\text{subject to } y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1 \geq 0 \text{ for all } 1 \leq i \leq m$$

C. Cortes and V. Vapnik (1995). Support Vector Networks. *Machine Learning*.

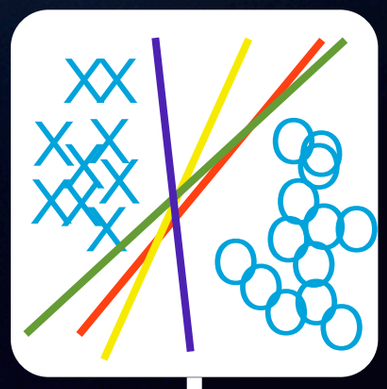
Neural Networks Became Uncool

SVMs

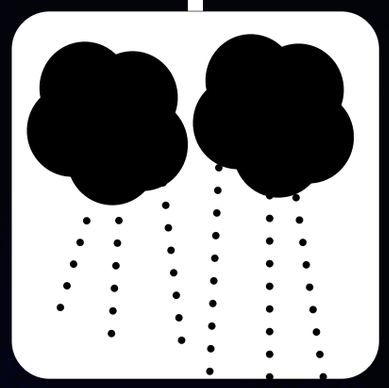
- SVMs killed the credibility of neural network research
- They soon went the way of other 1980's technology
- Neural networks about as uncool as you could possibly get...

Deep Learning

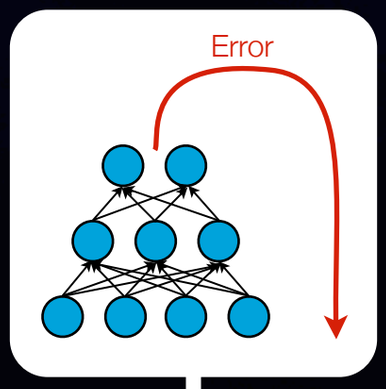
A long, long time ago...



1959
Perceptrons

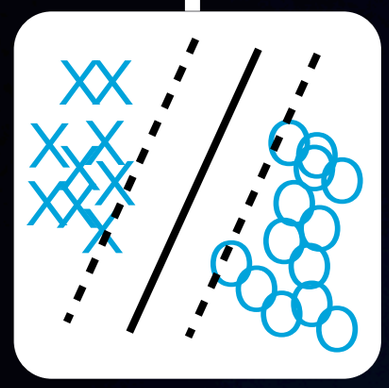


AI Winter
1969



1986
Artificial Neural
Networks

Support Vector
Machines
1995

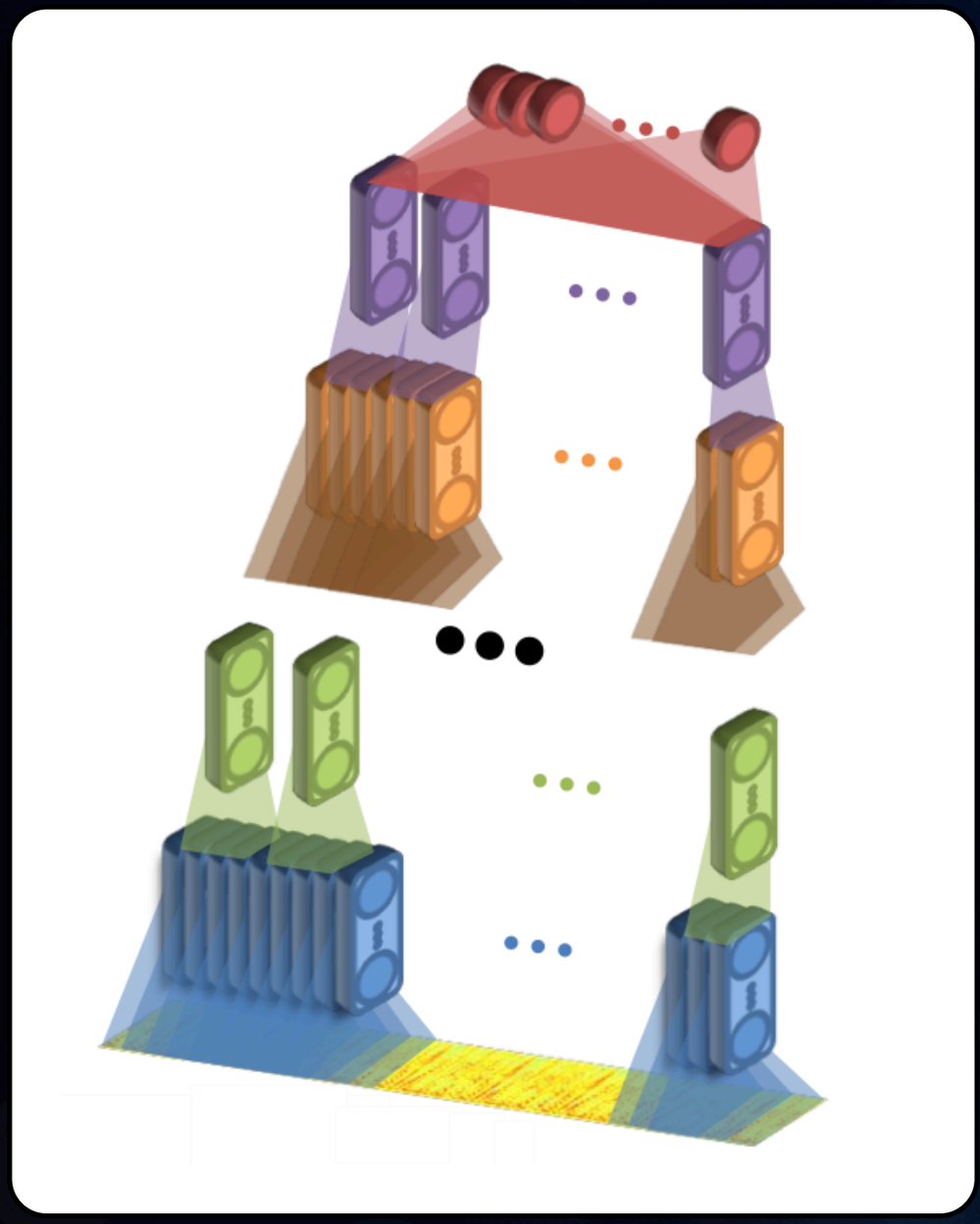


LeNet
1998

Convolutional Neural Networks

Interesting work did continue

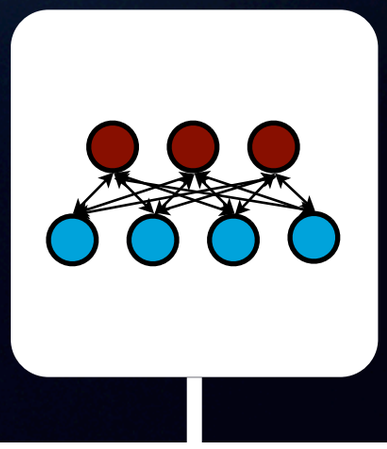
- Initial work focused on image and handwritten digit detection
  - Problems where scale/ rotation invariance are important
- Using convolutional functions allows taking variability of shapes
  - Convolutional layers (linear)
  - Max pooling (non-linear)



Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE.*

Deep Learning

A little more recently...



2002
Restricted
Boltzman
Machines

Restricted Boltzman Machines

Building blocks of “deep belief networks”

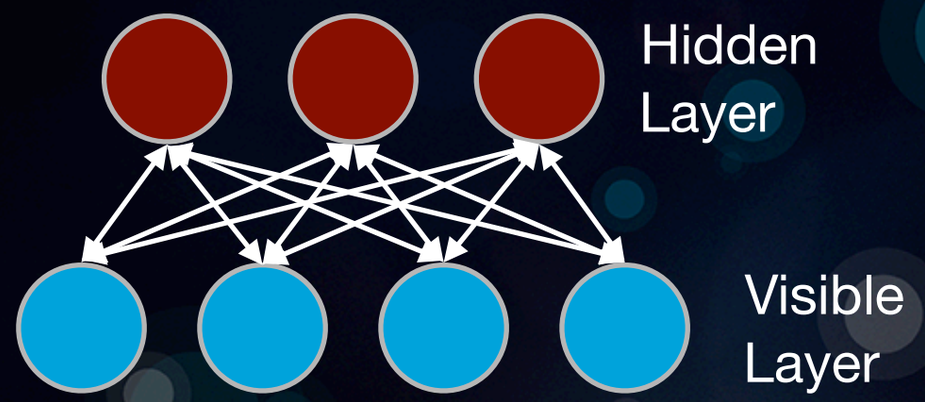
- Restricted Boltzman machines (RBMs) are a type of log-linear Markov random field with hidden units
- Two layer architecture, visible units v and hidden units h

Energy Function:

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{i \in \text{freq}} b_i v_i - \sum_{j \in \text{features}} c_j h_j - \sum_{i,j} v_i h_j w_{ij}$$

Model Distribution:

$$P(\mathbf{v}) = \sum_{\mathbf{h}} P(\mathbf{v}, \mathbf{h}) = \sum_{\mathbf{h}} \frac{e^{-E(\mathbf{v},\mathbf{h})}}{Z}$$



Restricted Boltzman Machines

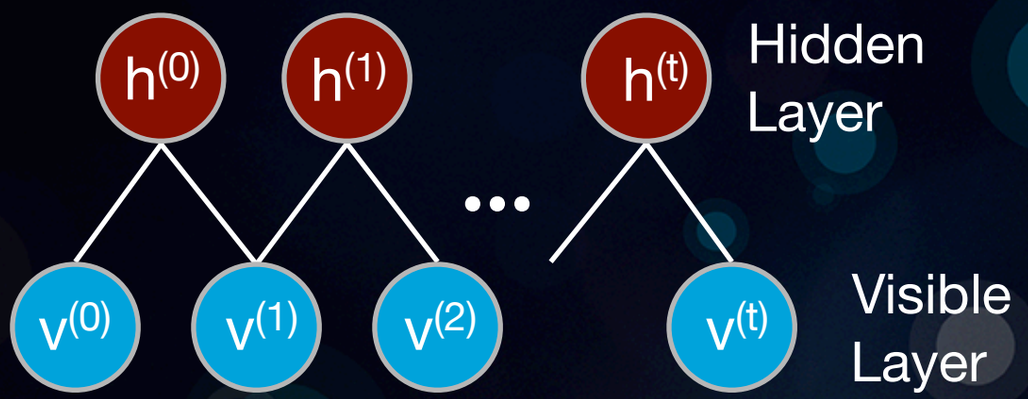
Inference

- In order to optimize this function it is necessary to first determine the values of the hidden variables
- Gibbs sampling is used in this case
- In theory it should be run until the chain converges (i.e., values no longer change)

$$\mathbf{h}^{(n+1)} \sim \sigma(b_j + \sum_i w_{ij} v_i),$$

$$\mathbf{v}^{(n+1)} \sim \sigma(a_i + \sum_j w_{ij} h_j)$$

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$



Restricted Boltzman Machines

Learning

- In practice, the general approach is to run the chain for just 1 step
- Update the parameter estimates for the model
- Weighted by a learning rate ε
- This is done iteratively for some number of epochs

$$\frac{\partial \log P(\mathbf{v})}{\partial W} = \mathbf{v}' E[\mathbf{h}|\mathbf{v}] - E[\mathbf{v}'\mathbf{h}],$$

$$\frac{\partial \log P(\mathbf{v})}{\partial a} = E[\mathbf{h}|\mathbf{v}] - E[\mathbf{h}],$$

$$\frac{\partial \log P(\mathbf{v})}{\partial b} = \mathbf{v} - E[\mathbf{v}].$$

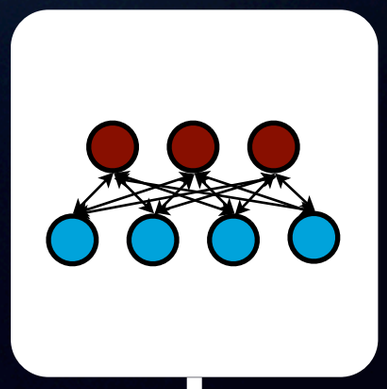
$$W = W + \epsilon(\mathbf{v}^{(0)}\hat{\mathbf{h}}^{(0)\prime} - \mathbf{v}^{(1)}\hat{\mathbf{h}}^{(1)\prime}),$$

$$a = a + \epsilon(\hat{\mathbf{h}}^{(0)} - \hat{\mathbf{h}}^{(2)}),$$

$$b = b + \epsilon(\mathbf{v}^{(0)} - \mathbf{v}^{(1)})$$

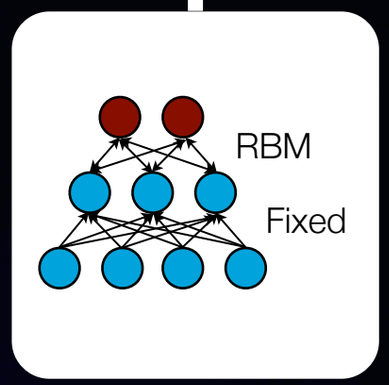
Deep Learning

A little more recently...



Greedy-Wise
Pretraining
2006

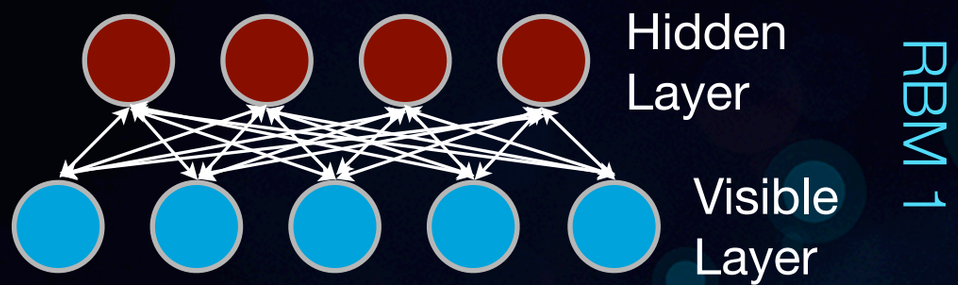
2002
Restricted
Boltzman
Machines



Learning Deep Autoencoders

Stacked RBMs

- Stacked RBMs shown as a powerful way to initialize autoencoders
- First we train RBMs one layer at a time
  - Previous hidden layer becomes current visible

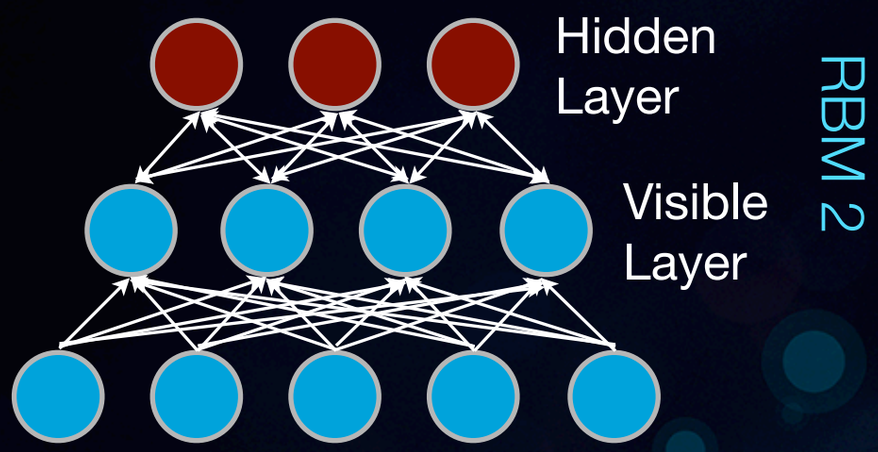


G. E. Hinton and R. R. Salakhutdinov (2006). Reducing the Dimensionality of Data with Neural Networks. *Science*.

Learning Deep Autoencoders

Stacked RBMs

- Stacked RBMs shown as a powerful way to initialize autoencoders
- First we train RBMs one layer at a time
  - Previous hidden layer becomes current visible

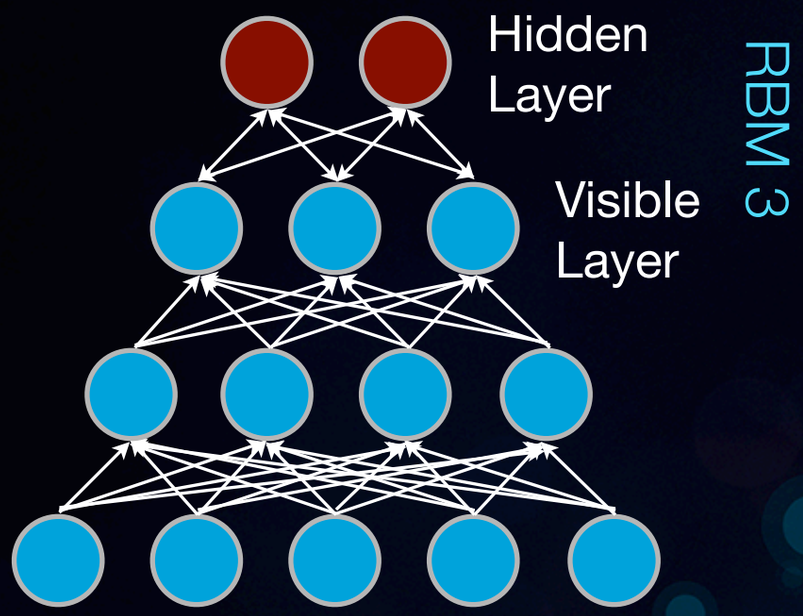


G. E. Hinton and R. R. Salakhutdinov (2006). Reducing the Dimensionality of Data with Neural Networks. *Science*.

Learning Deep Autoencoders

Stacked RBMs

- Stacked RBMs shown as a powerful way to initialize autoencoders
- First we train RBMs one layer at a time
  - Previous hidden layer becomes current visible

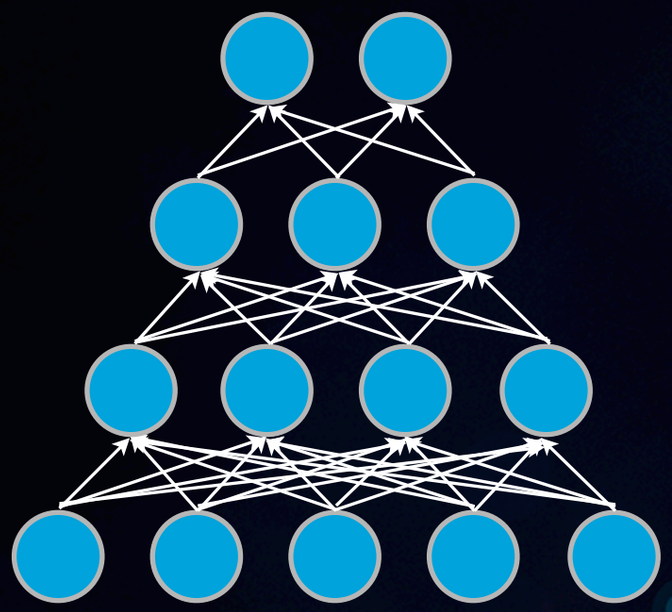


G. E. Hinton and R. R. Salakhutdinov (2006). Reducing the Dimensionality of Data with Neural Networks. *Science*.

Learning Deep Autoencoders

Stacked RBMs

- Stacked RBMs shown as a powerful way to initialize autoencoders
- First we train RBMs one layer at a time
  - Previous hidden layer becomes current visible

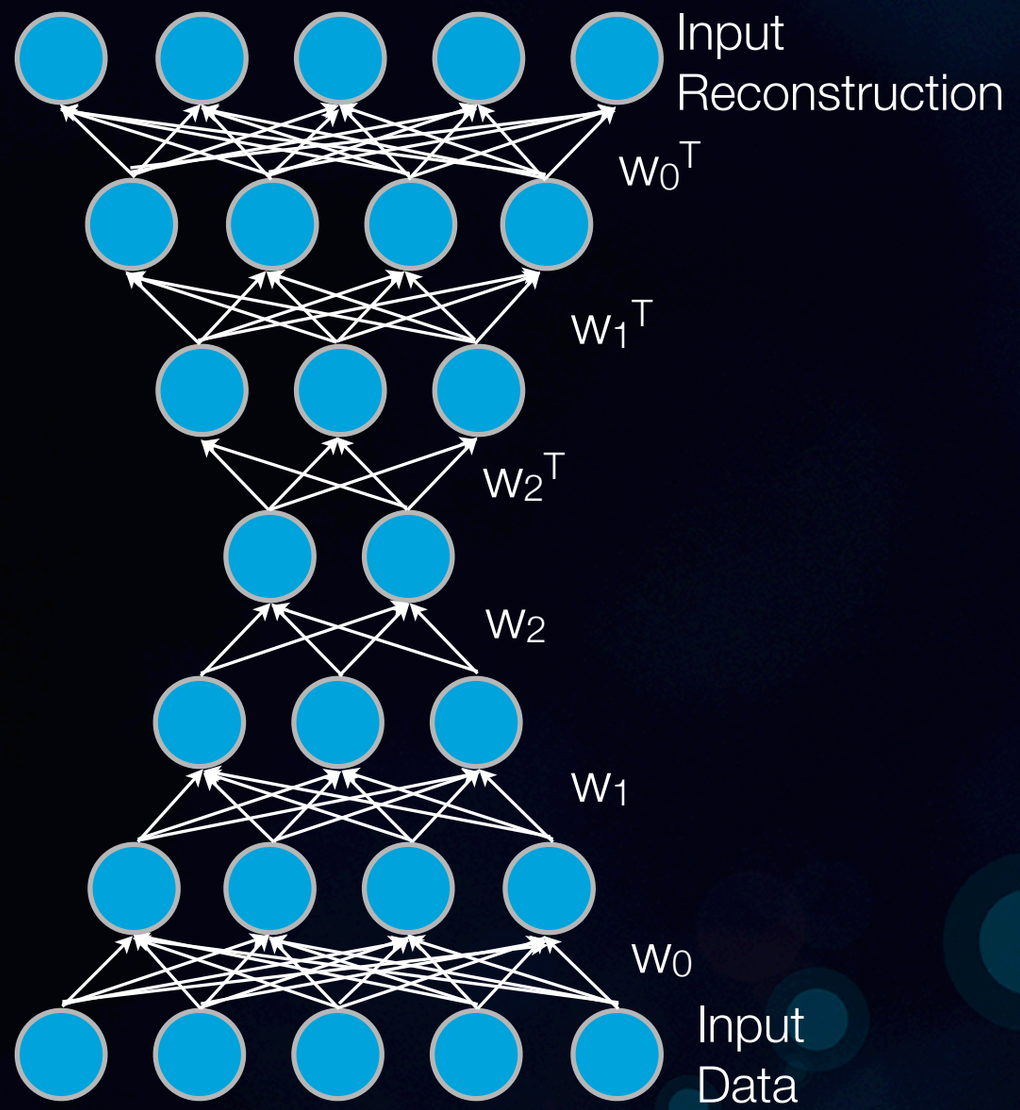


G. E. Hinton and R. R. Salakhutdinov (2006). Reducing the Dimensionality of Data with Neural Networks. *Science*.

Learning Deep Autoencoders

Stacked RBMs

- Stacked RBMs shown as a powerful way to initialize autoencoders
- First we train RBMs one layer at a time
  - Previous hidden layer becomes current visible

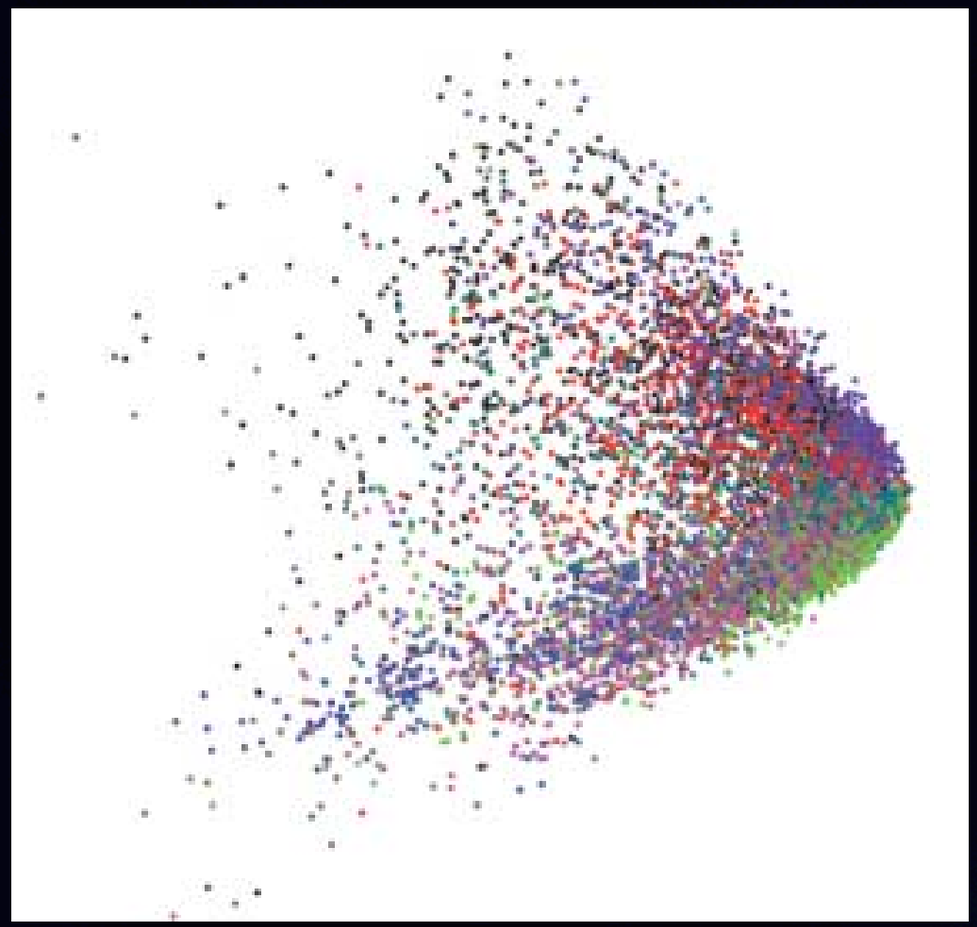


G. E. Hinton and R. R. Salakhutdinov (2006). Reducing the Dimensionality of Data with Neural Networks. *Science*.

Learning Deep Autoencoders

Stacking RBMs

Two dimensional latent semantic analysis (Figure 4, B)

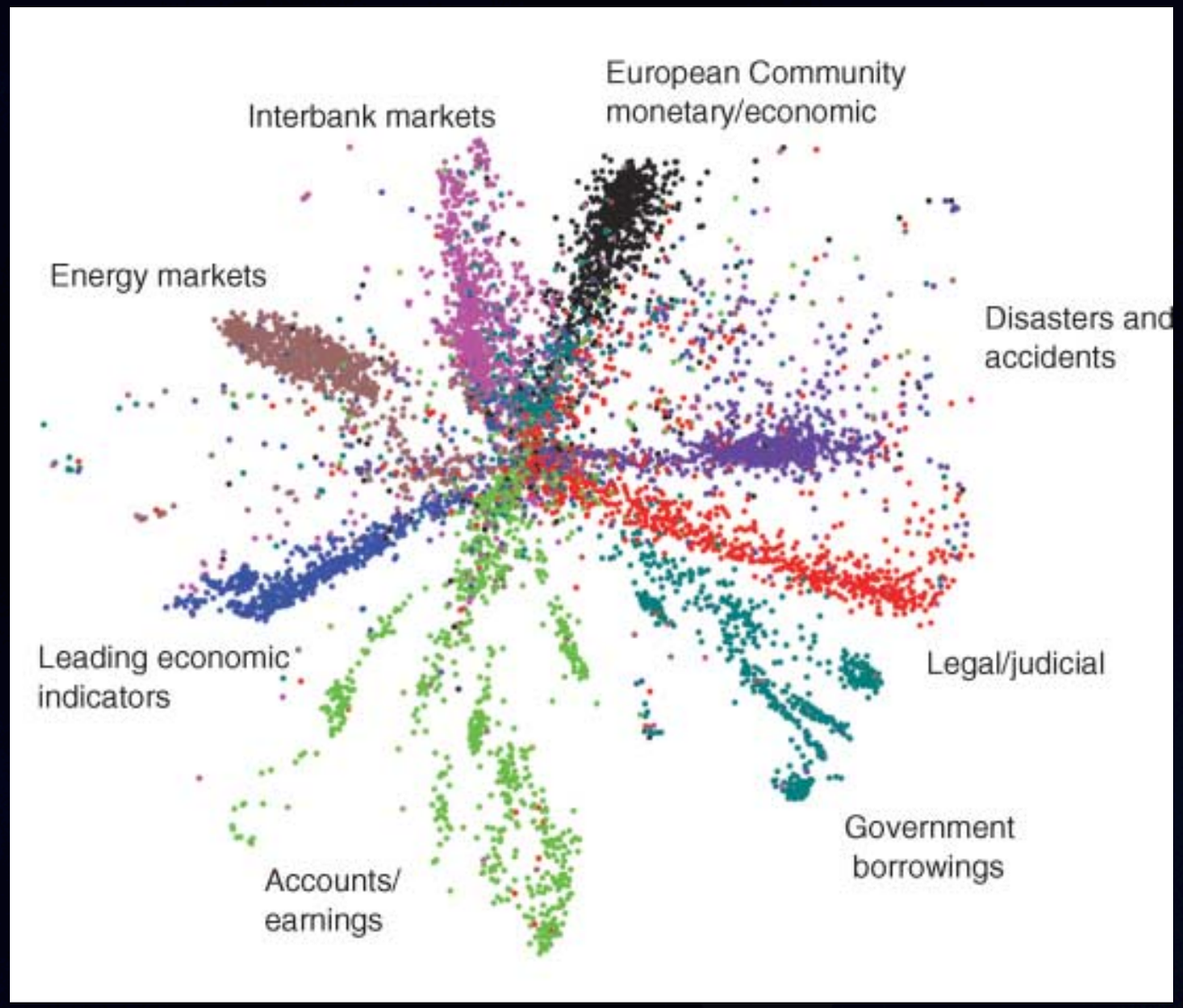


G. E. Hinton and R. R. Salakhutdinov (2006). Reducing the Dimensionality of Data with Neural Networks. *Science*.

Learning Deep Autoencoders

Stacking RBMs

2000-500-250-125-2 Autoencoder (Figure 4, C)



G. E. Hinton and R. R. Salakhutdinov (2006). Reducing the Dimensionality of Data with Neural Networks. *Science*.

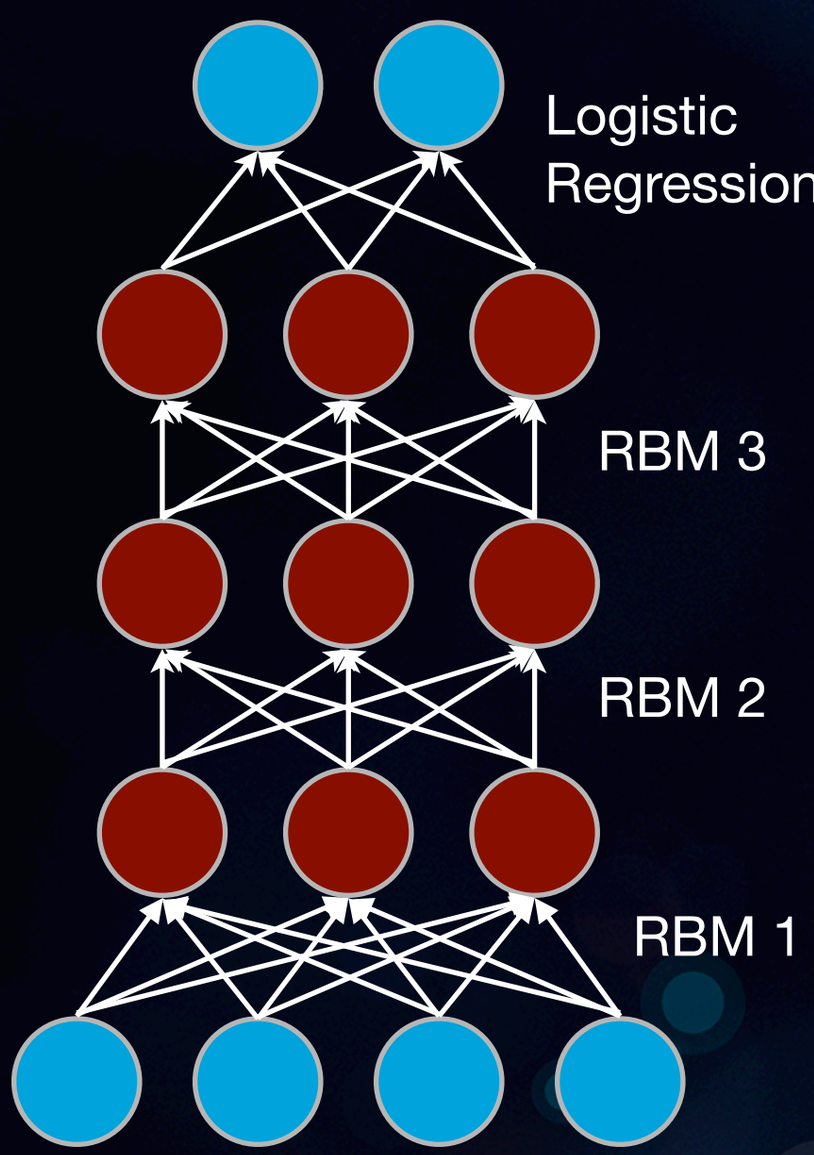
Greedy-Wise Pre-training

Initializing model with RBMs

- DBNs are pre-trained by “greedily” stacking restricted Boltzmann machines (RBMs)
- Logistic regression is attached last
- Model is fine-tuned with gradient descent:

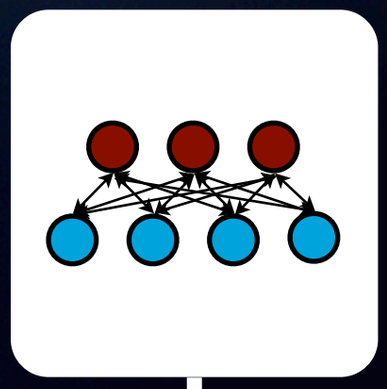
$$\theta^{(t+1)} = \theta^{(t)} - \epsilon \Delta E(\theta^{(t)})$$

  - Model parameters θ
  - Learning rate ε (chosen experimentally)
  - Error function negative log likelihood

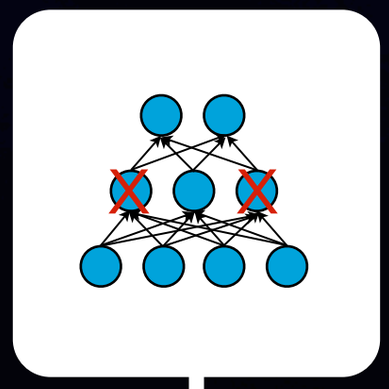


Deep Learning

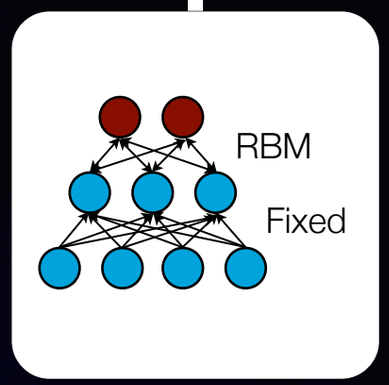
A little more recently...



Greedy-Wise
Pretraining
2006



2002
Restricted
Boltzman
Machines



2012
Dropout

Dropout

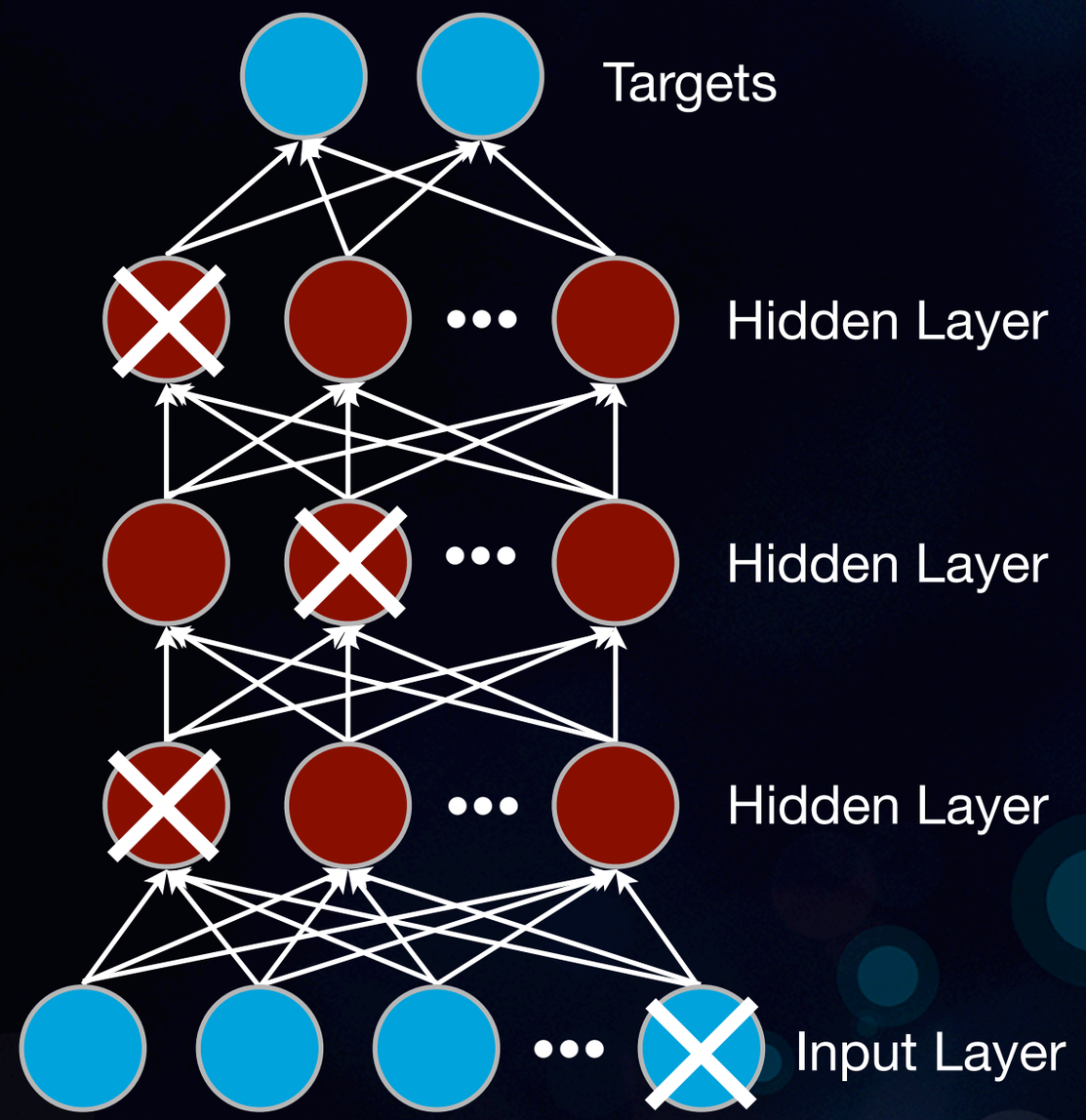
Regularization

- So far the methods discussed use very simplistic regularization strategies
  - A validation set is held out (separate from training and test data)
  - On each iteration we verify that error is reducing on both the training and on validation set
  - Since the validation set is unseen by the algorithm, this will help to prevent over fitting

Dropout

Regularization

- Dropout takes things a step further
- Independently set each hidden unit's activity to 0 with 0.5 probability
- Correlations between nodes cannot develop
  - Also called co-adaptions

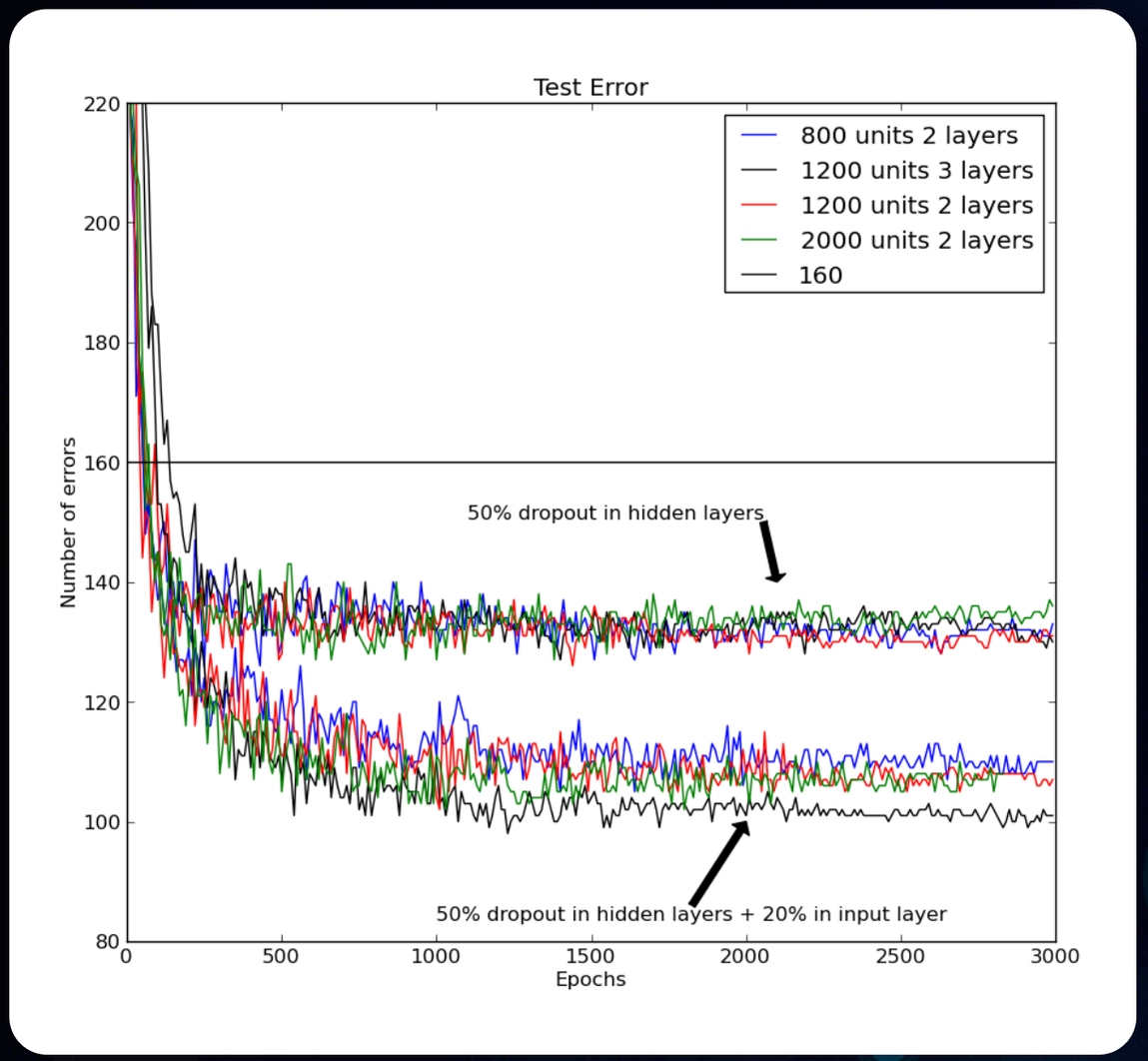


G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever and R. R. Salakhutdinov (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv*.

Dropout

MNIST

- Handwritten digit recognition
- 60,000 20x20 training images
- 20,000 test images
- Previously best result was 160 errors (black line)

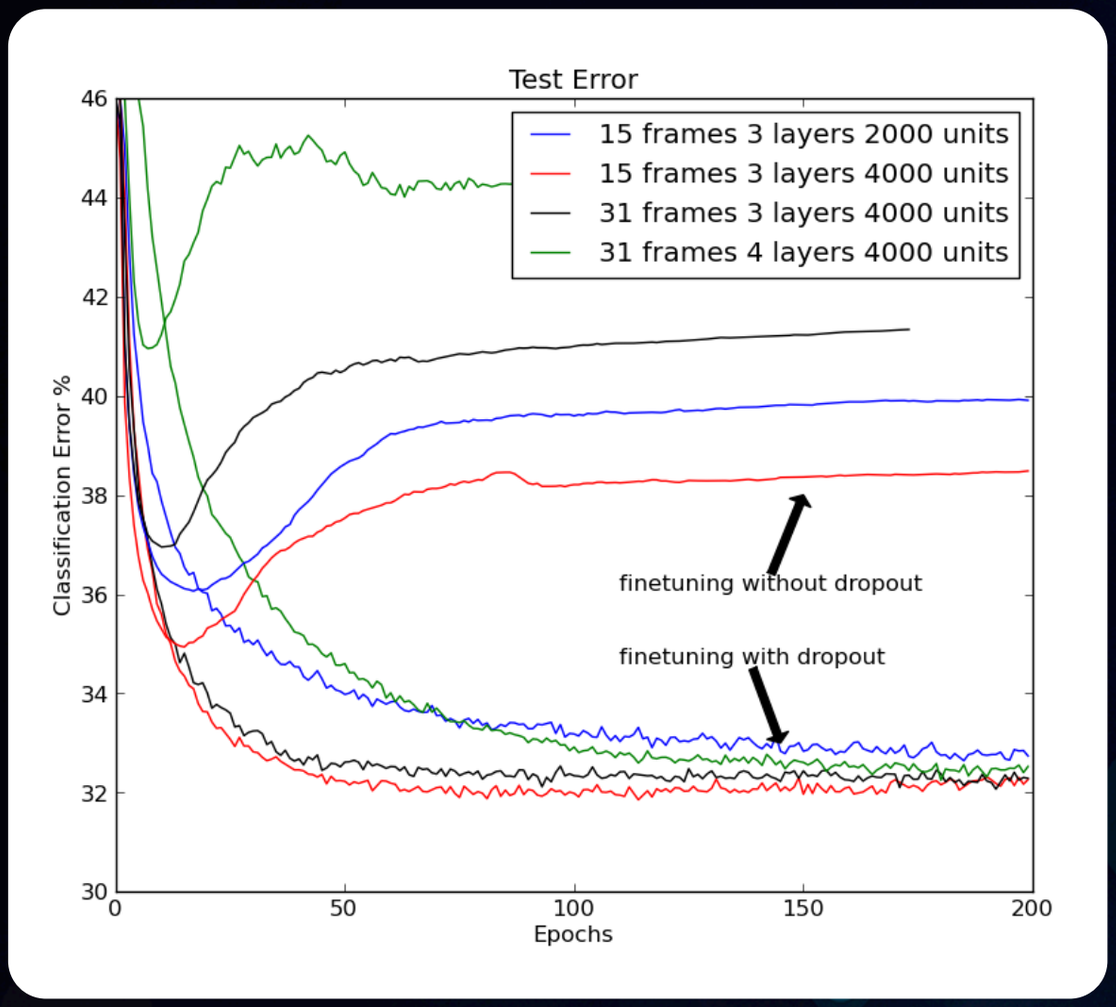


G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever and R. R. Salakhutdinov (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv*.

Dropout

TIMIT

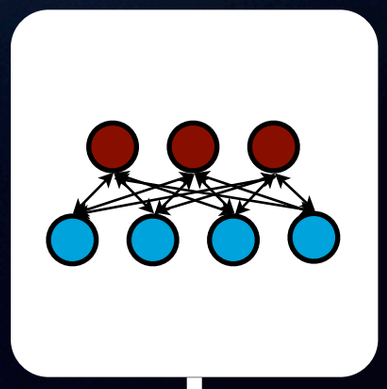
- Recognition of clean speech (no noise)
- Pre-trained DBNs had been shown previously to outperform GMMs
- Used in combination with hidden Markov model



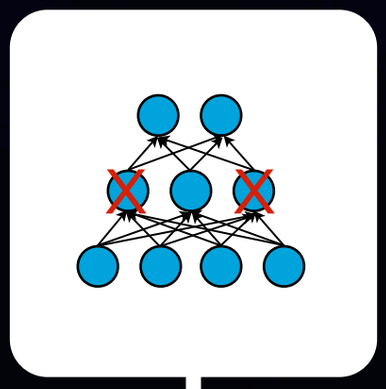
G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever and R. R. Salakhutdinov (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv*.

Deep Learning

A little more recently...

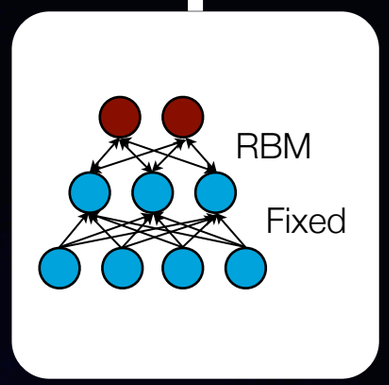


Greedy-Wise Pretraining 2006

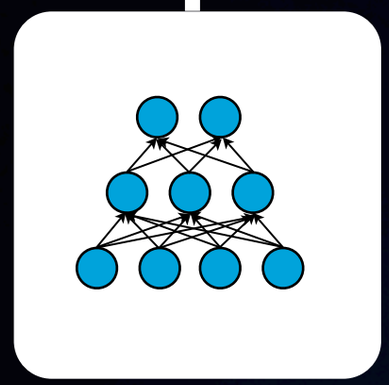


Artificial Neural Networks 1986/2013

2002 Restricted Boltzman Machines



2012 Dropout



Artificial Neural Networks

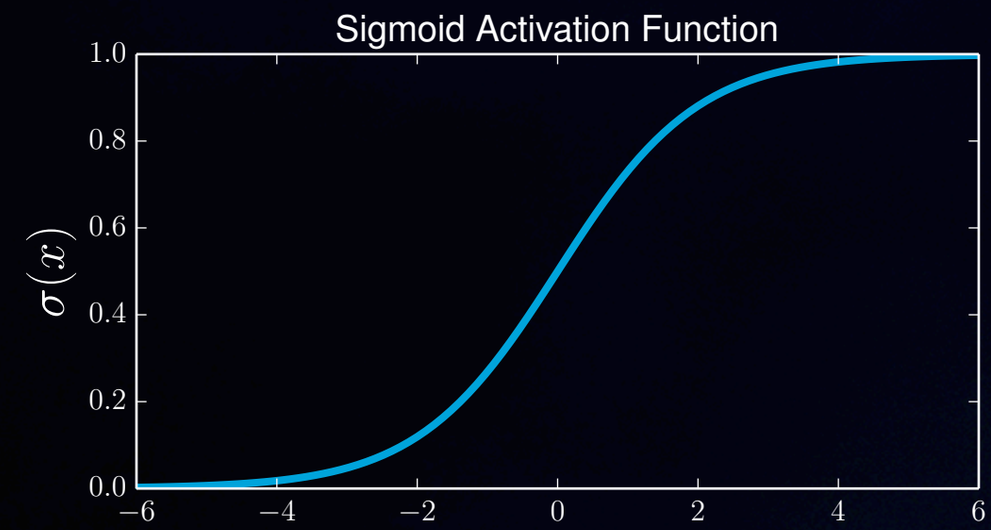
Back to 1986!

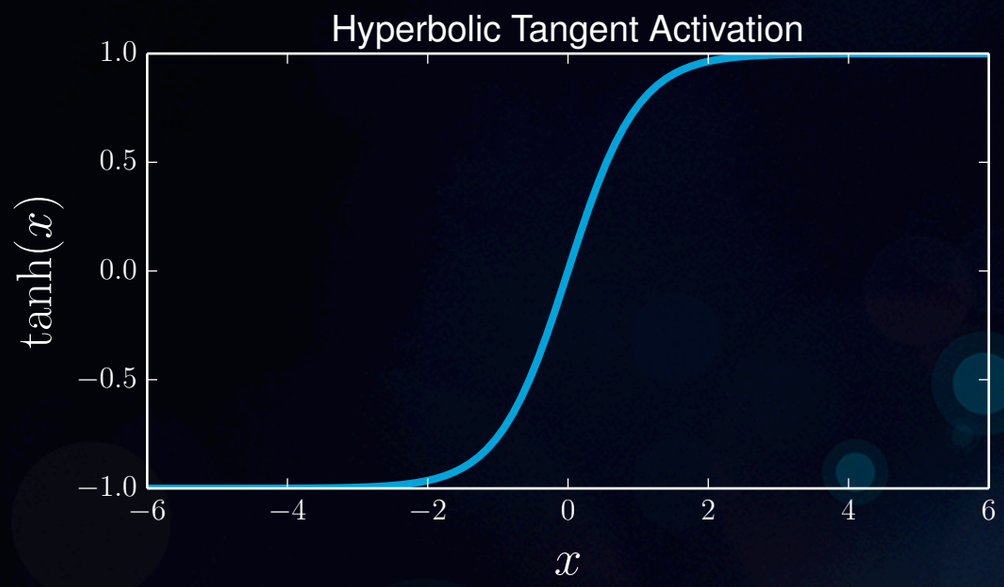
- Many researchers are now claiming that neither pre-training nor dropout are necessary given enough training data
- These methods are nearly identical to the original neural networks of 1986
- The only major change is the use of rectified linear units
    - And ***a lot*** more data
- As Geoff Hinton likes to put it, we just had to wait long enough for our datasets and computing power to scale

Rectified Linear Units

Change in activation

- Computing non-linear activations is computationally expensive
- Each dimension of each example must be evaluated
- With rectified linear units the values remain unchanged (or set to 0)

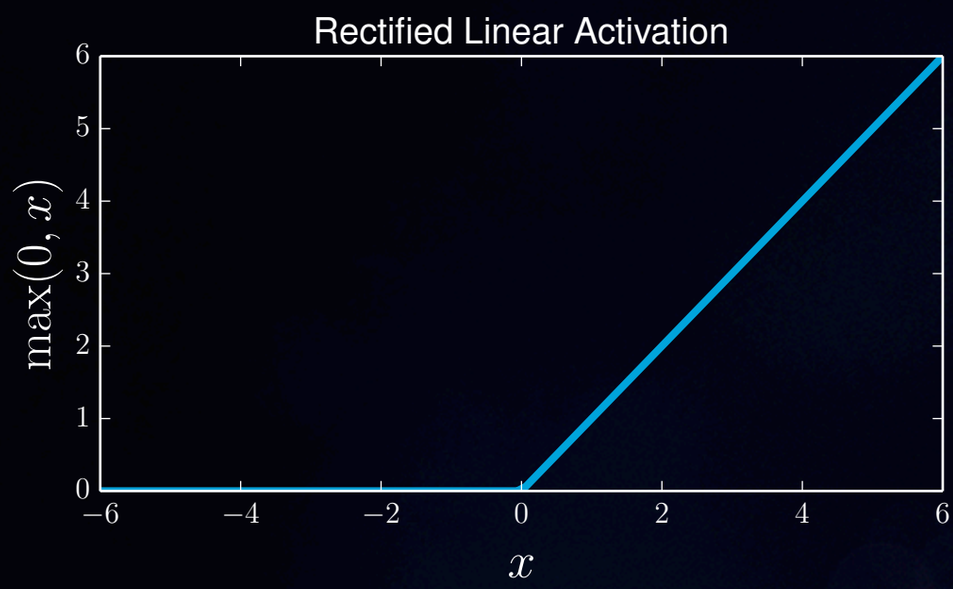




Rectified Linear Units

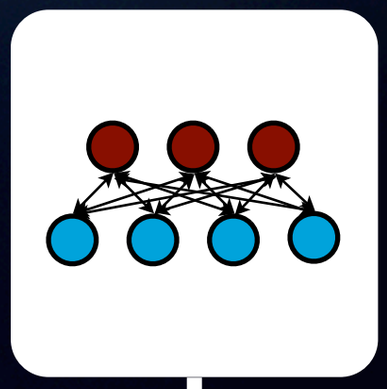
Change in activation

- Computing non-linear activations is computationally expensive
- Each dimension of each example must be evaluated
- With rectified linear units the values remain unchanged (or set to 0)

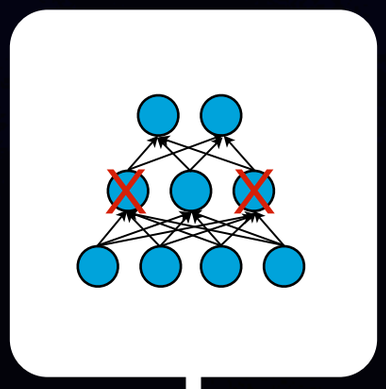


Deep Learning

A little more recently...

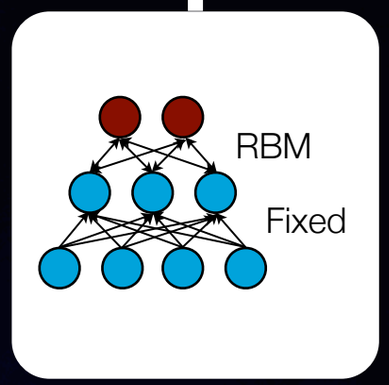


Greedy-Wise Pretraining 2006

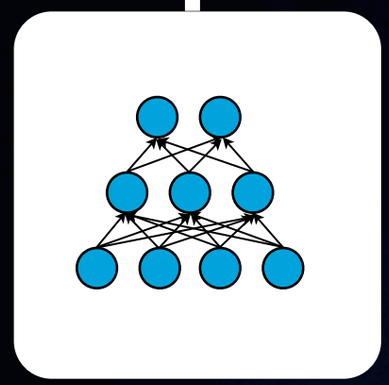


Artificial Neural Networks 1986/2013

2002 Restricted Boltzman Machines



2012 Dropout



The Future

Music Emotion Recognition

Music Emotion Recognition

Overview

Goal:

- Content-based modeling and prediction of musical emotion
- Applications in sorting, recommendation, playlist generation

Challenges:

1. Emotional descriptions often lack a singular, well-defined answer making ground truth labeling difficult
2. No dominant acoustic feature has yet arisen for musical emotion
3. Emotions change over time

Music Emotion Recognition

Overview

Goal:

- Content-based modeling and prediction of musical emotion
- Applications in sorting, recommendation, playlist generation

Challenges:

1. Emotional descriptions often lack a singular, well-defined answer making ground truth labeling difficult
2. No dominant acoustic feature has yet arisen for musical emotion
3. Emotions change over time

Ground Truth Data Collection

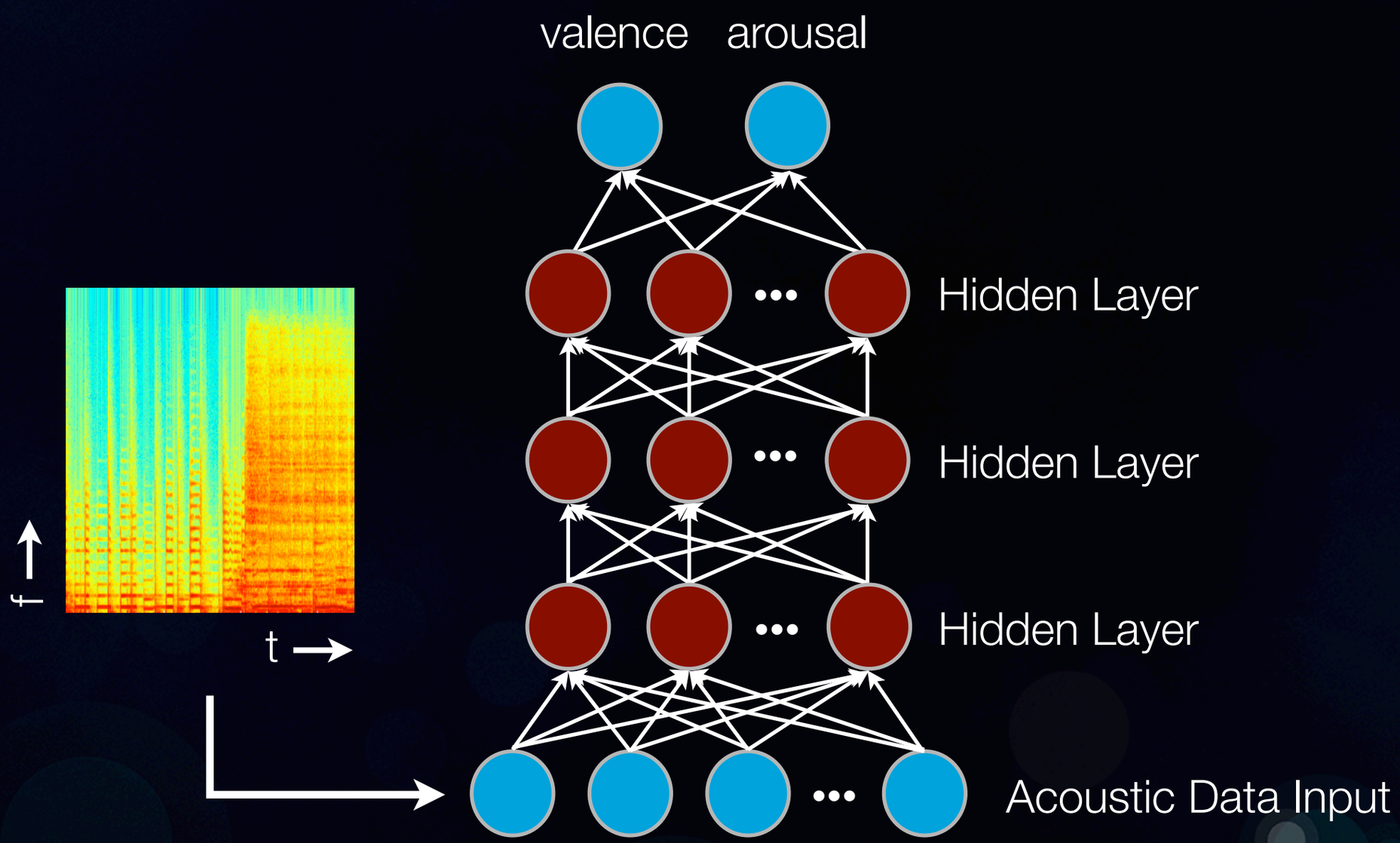
MoodSwings Turk

- Re-annotation of MoodSwings dataset with Amazon's Mechanical Turk
- Highly correlated to MoodSwings labels
  - Finds arousal r=0.712, valence r=0.846
- Annotated with 16.93 ± 2.690 ratings per clip
- Dataset has been made publicly available

J. A. Speck, E. M. Schmidt, B. G. Morton, and Y. E. Kim (2011). A comparative study of collaborative vs. traditional annotation methods. *ISMIR*. Miami, FL.

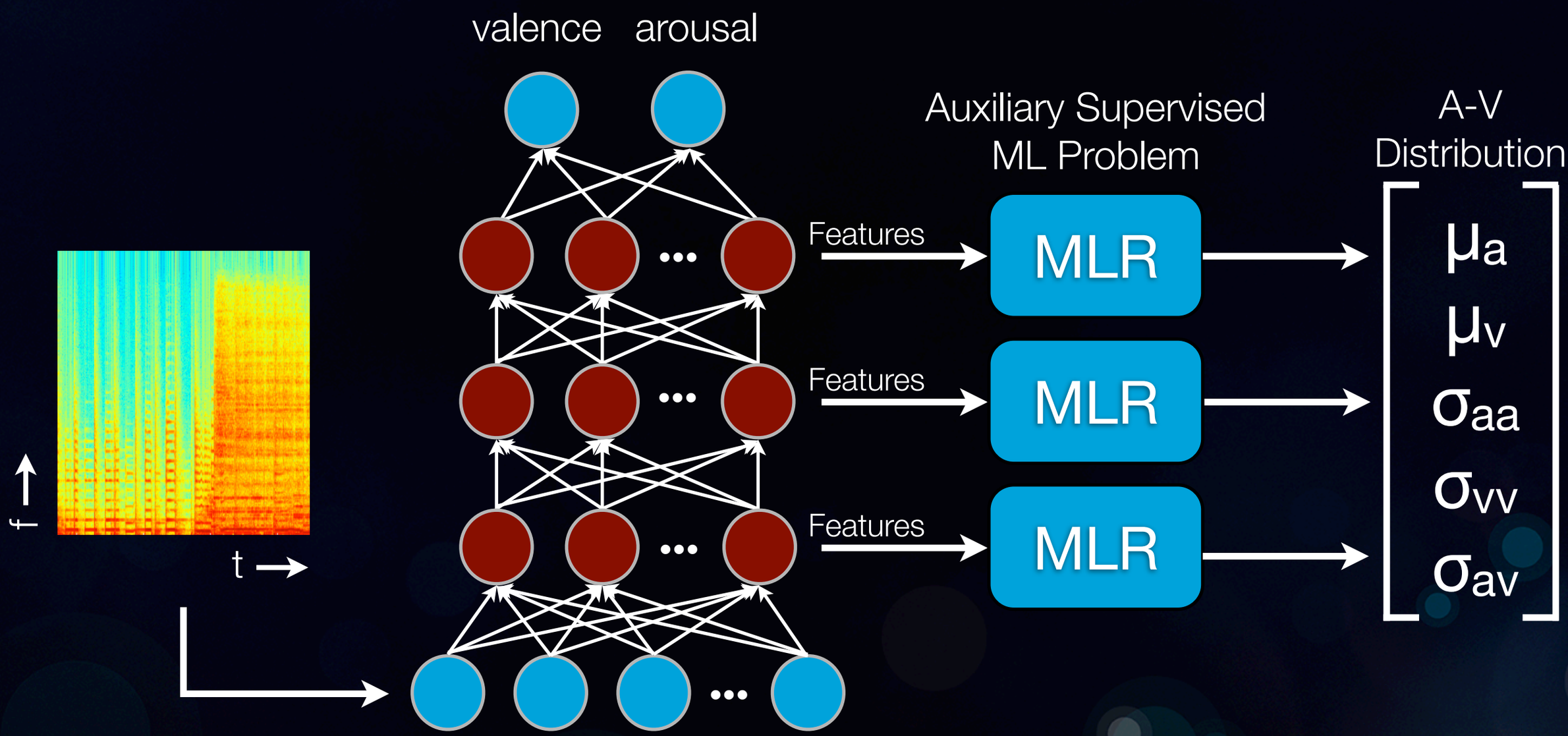
Framework

Deep learning in music emotion recognition



Framework

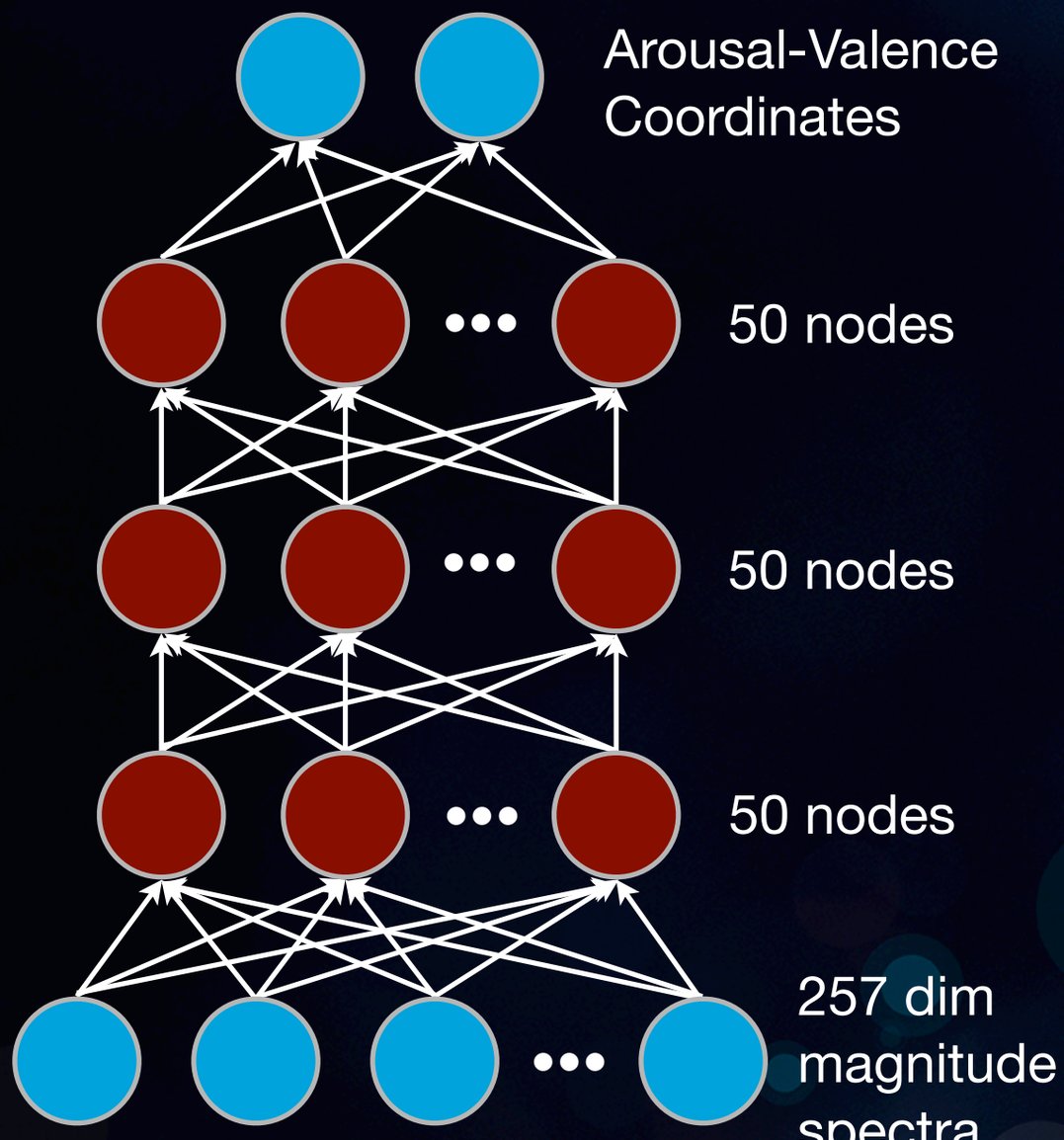
Deep learning in music emotion recognition



Feature Learning

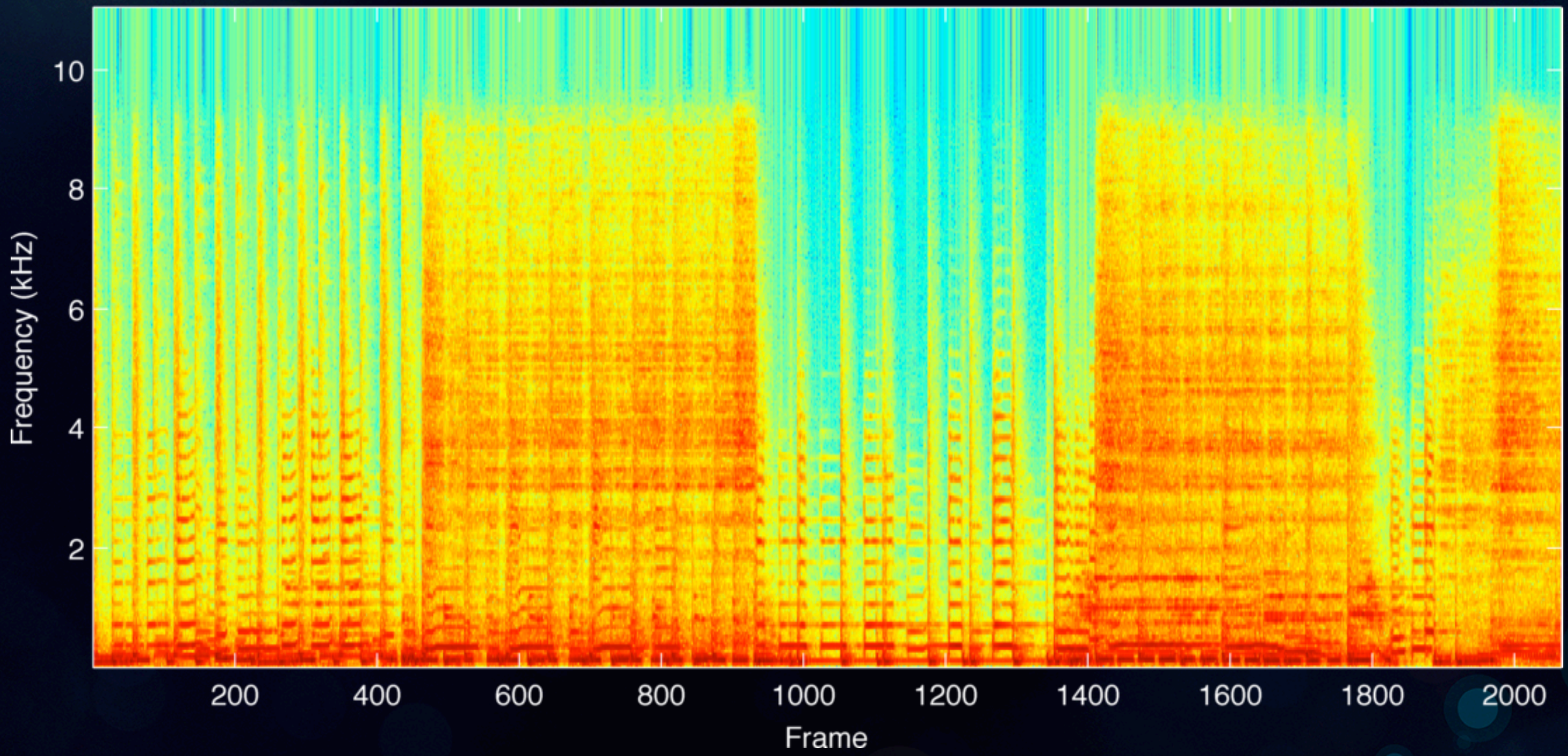
Single STFT Frame

- 50 dimensions are used in the hidden layers (chosen experimentally)
- Model is trained to predict A-V means
- Input is raw magnitude spectra
  - ~20msec windows

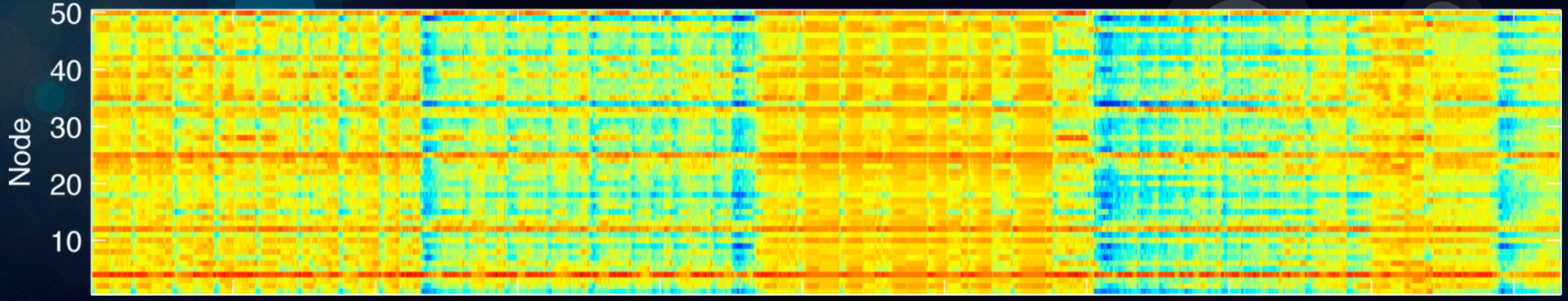


E. M. Schmidt and Y. E. Kim (2011). Learning emotion-based acoustic features with deep belief networks. *IEEE WASPAA.*

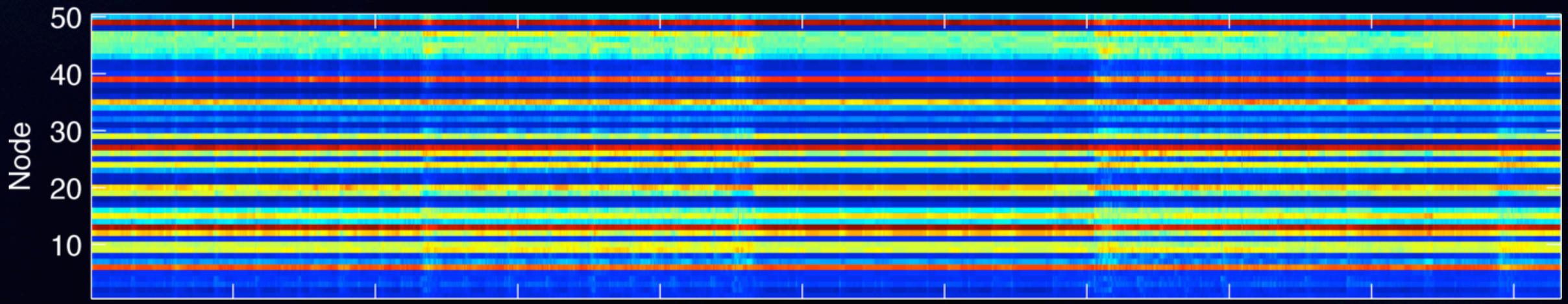
Magnitude Spectrogram
Soulive: Come Together, 50-70 secs



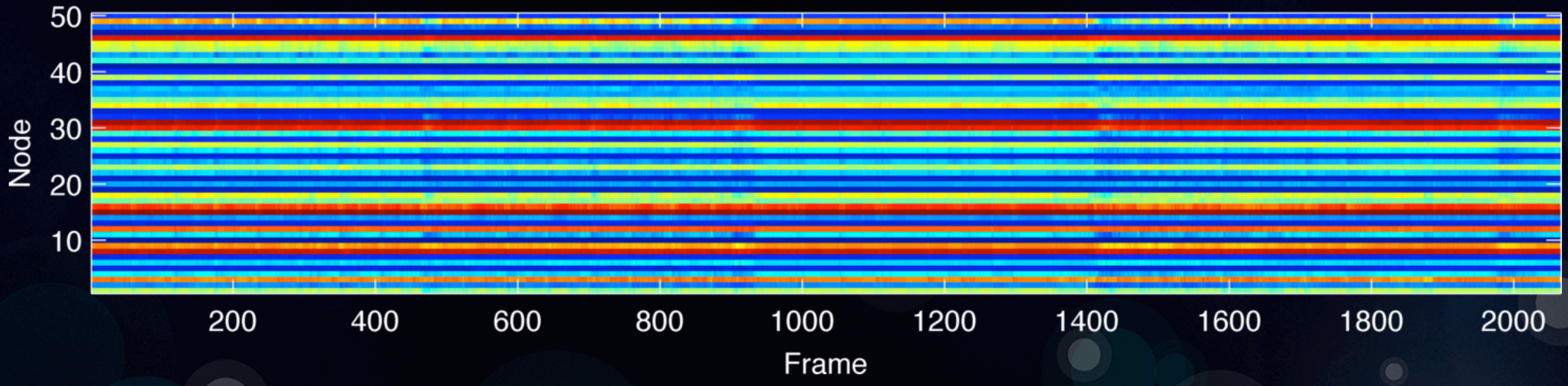
Layer 1 Features



Layer 2 Features



Layer 3 Features

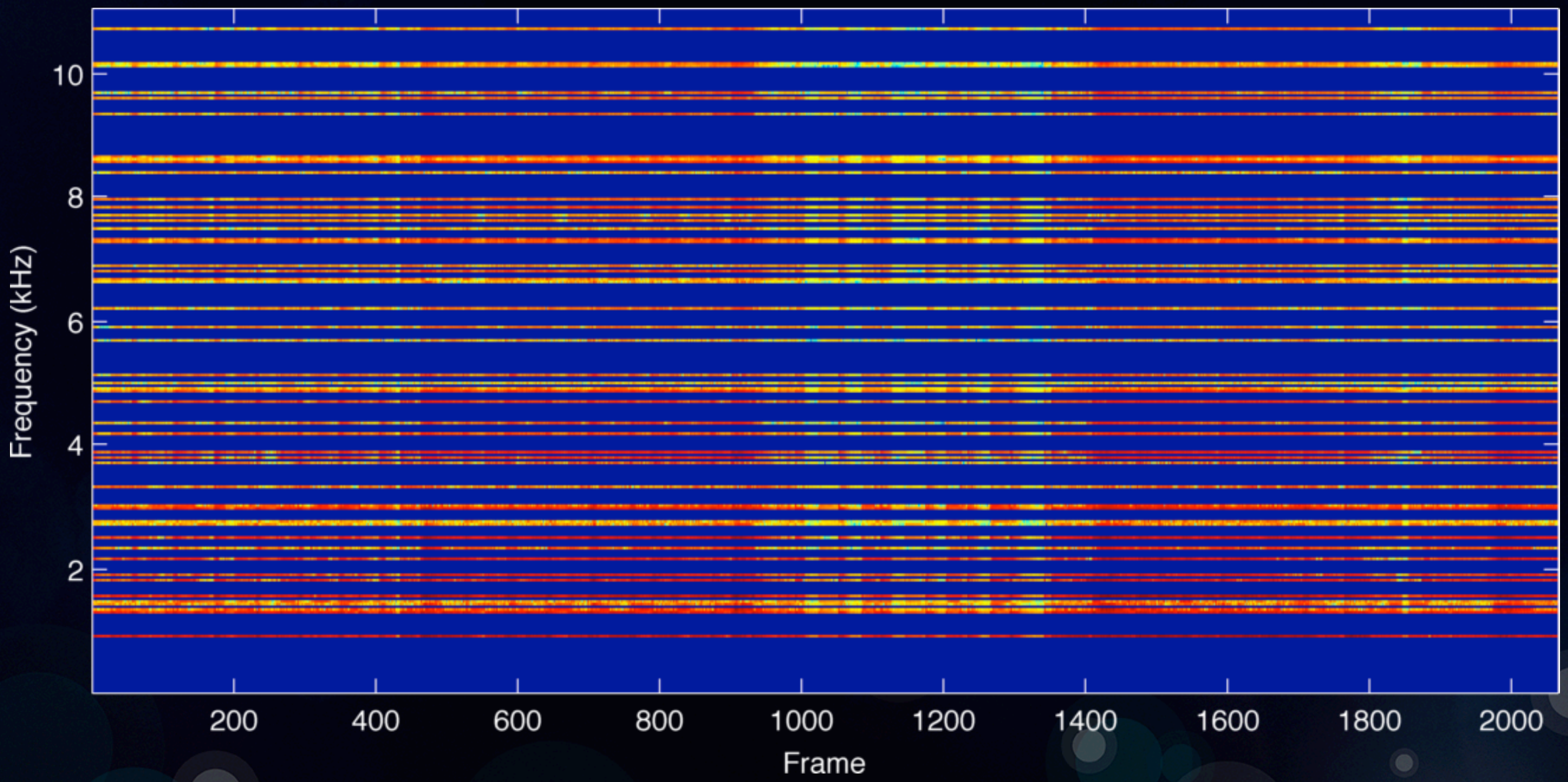


STFT Reconstruction

$$\mathbf{y}_{\mathrm{layer1}i} = \frac{1}{(1 + \exp(-\mathbf{x}_i\mathbf{w} - \mathbf{b}))}$$

STFT Reconstruction

$$\mathbf{x}_{\text{reconstructed}_i} = \left(-\ln\left(\frac{1}{\mathbf{y}_{\text{layer1}_i}} - 1 \right) - \mathbf{b} \right) \mathbf{w}^{+}$$



Results

Emotion prediction with learned features

Feature/ Topology	Average Mean Distance	Average KL Divergence
MFCCs	0.140 ± 0.005	1.28 ± 0.157
Chroma	0.182 ± 0.006	3.33 ± 0.294
Spectral Shape	0.153 ± 0.006	1.51 ± 0.160
Spectral Contrast	0.138 ± 0.005	1.29 ± 0.160
ENTs	0.151 ± 0.006	1.41 ± 0.175
DBN Model Error	0.203 ± 0.009	-
DBN Layer 1	0.138 ± 0.005	1.25 ± 0.142
DBN Layer 2	0.133 ± 0.004	1.19 ± 0.129
DBN Layer 3	0.133 ± 0.002	1.21 ± 0.180

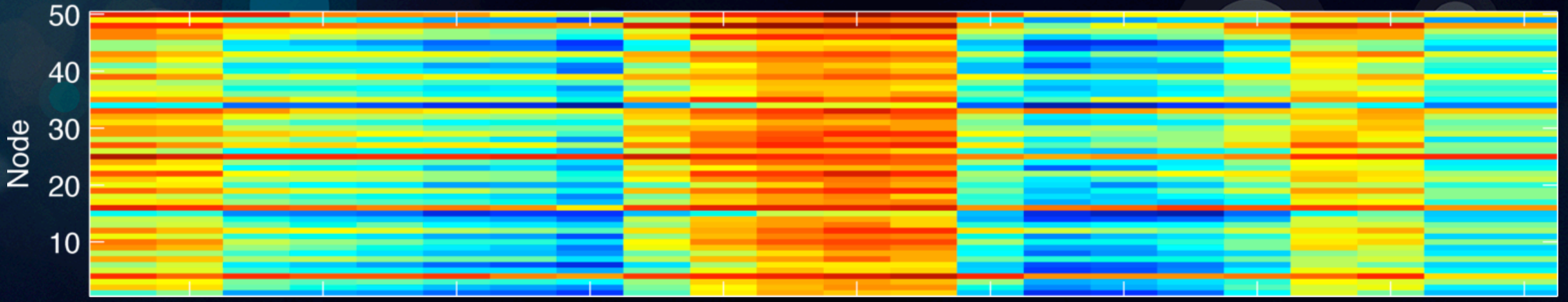
Next Steps

Developing more informed representations

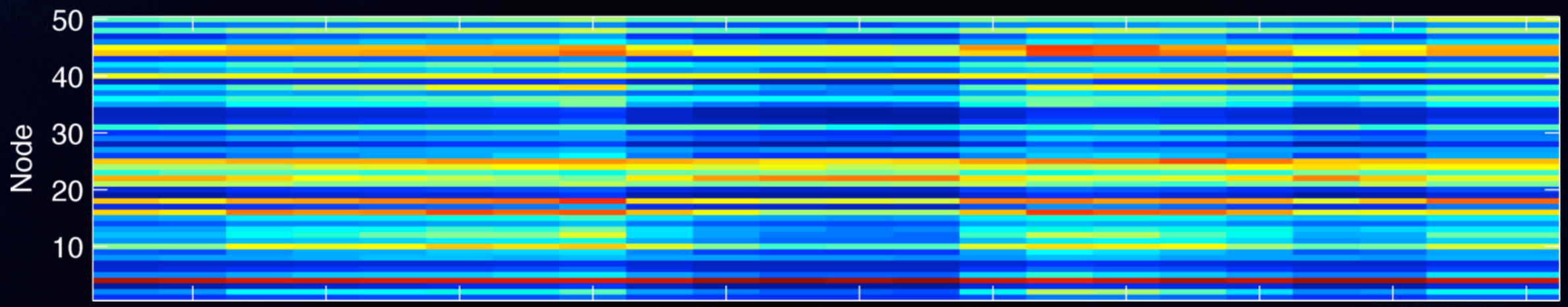
- Current features are extracted from ~20 msec frames
  - No human can identify emotion at this time scale
  - Emotion is not expected to vary at this high a rate
- Instead look at multiple aggregated time scales
  - We investigate 1 sec, 2 sec, 4 sec scales combined

E. M. Schmidt, J. Scott, and Y. E. Kim (2012). Feature Learning in Dynamic Environments: Modeling the Acoustic Structure of Musical Emotion. *ISMIR.*

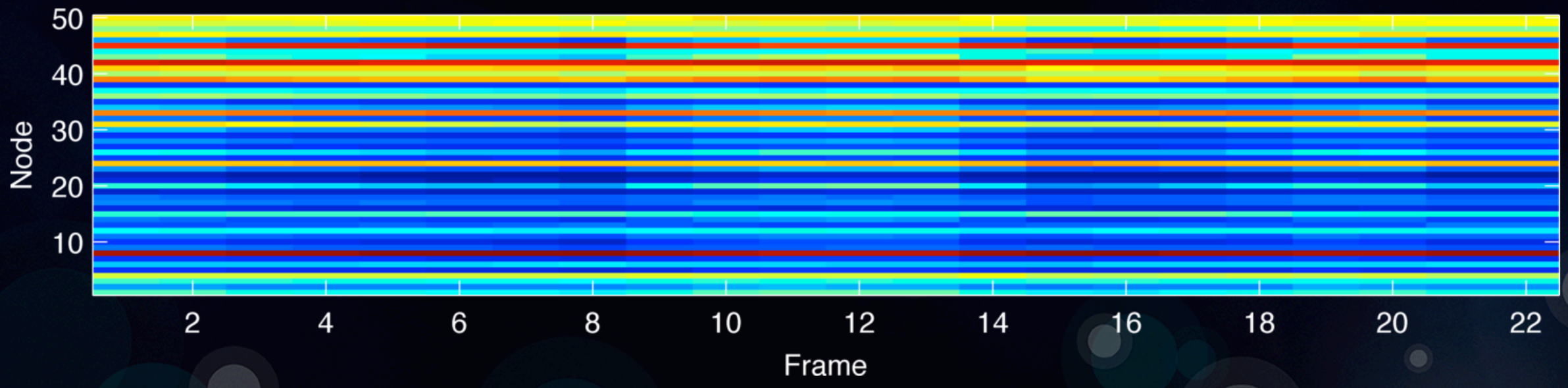
Layer 1 Features



Layer 2 Features



Layer 3 Features



Results

Developing more informed representations

Feature/ Topology	Average Mean Distance	Average KL Divergence
DBN-SF Model Error	0.203 ± 0.009	-
DBN-SF Layer 1	0.138 ± 0.005	1.25 ± 0.142
DBN-SF Layer 2	0.133 ± 0.004	1.19 ± 0.129
DBN-SF Layer 3	0.133 ± 0.002	1.21 ± 0.180
DBN-MF Model Error	0.194 ± 0.032	-
DBN-MF Layer 1	0.131± 0.006	1.15 ± 0.106
DBN-MF Layer 2	0.131 ± 0.004	1.14 ± 0.107
DBN-MF Layer 3	0.129 ± 0.004	1.12 ± 0.114

Harnessing Unlabeled Data

Universal background model

- Boost performance by including much more data in the unsupervised phase
  - No reason we should be limited to 240 song dataset
    - 50% training = 120 songs
  - Access to an additional ~7000 songs
    - ~1.5M examples
  - Develop generalized music model, fine-tune to predict emotion

Results

Universal background model

Feature/ Topology	Average Mean Distance	Average KL Divergence
DBN-MF Model Error	0.194 ± 0.032	-
DBN-MF Layer 1	0.131± 0.006	1.15 ± 0.106
DBN-MF Layer 2	0.131 ± 0.004	1.14± 0.107
DBN-MF Layer 3	0.129 ± 0.004	1.12 ± 0.114
DBN-UBM Model Error	0.140 ± 0.015	-
DBN-UBM Layer 1	0.129 ± 0.006	1.12 ± 0.091
DBN-UBM Layer 2	0.128 ± 0.004	1.13 ± 0.097
DBN-UBM Layer 3	0.128 ± 0.004	1.11 ± 0.090

Selecting Input Representations

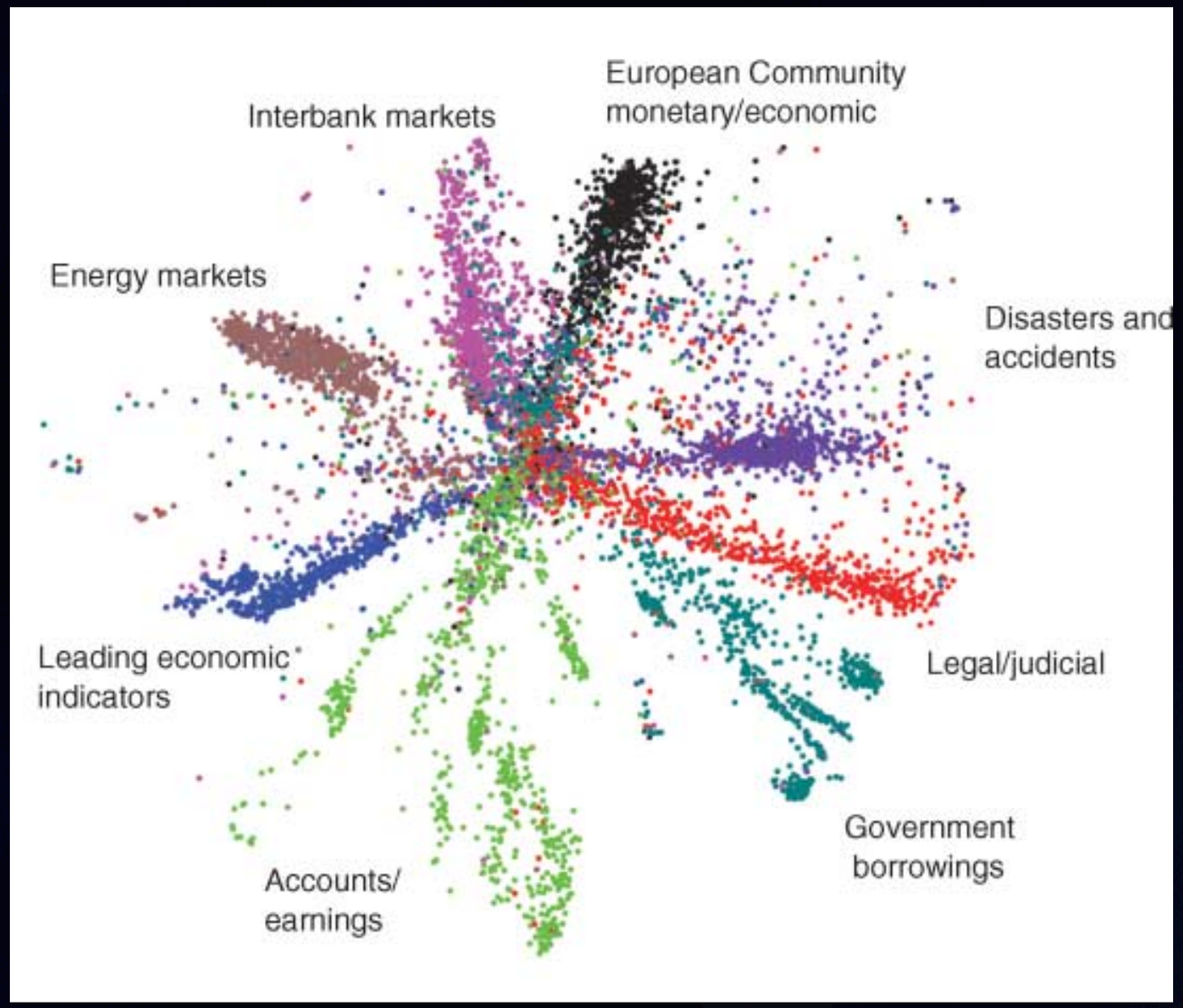
Taking best advantage of pre-training

- Work thus far used magnitude spectra as the input representation
  - Sought direct mapping between acoustic data and emotion space parameters
  - Saw modest performance gains
- Unsupervised methods tend to highlight the most prominent aspects of data
  - Such clustering with spectrograms may not provide a useful starting point for gradient descent

Learning Deep Autoencoders

Stacking RBMs

2000-500-250-125-2 Autoencoder (Figure 4, C)



G. E. Hinton and R. R. Salakhutdinov (2006). Reducing the Dimensionality of Data with Neural Networks. *Science*.

Rhythm and Melody DBNs

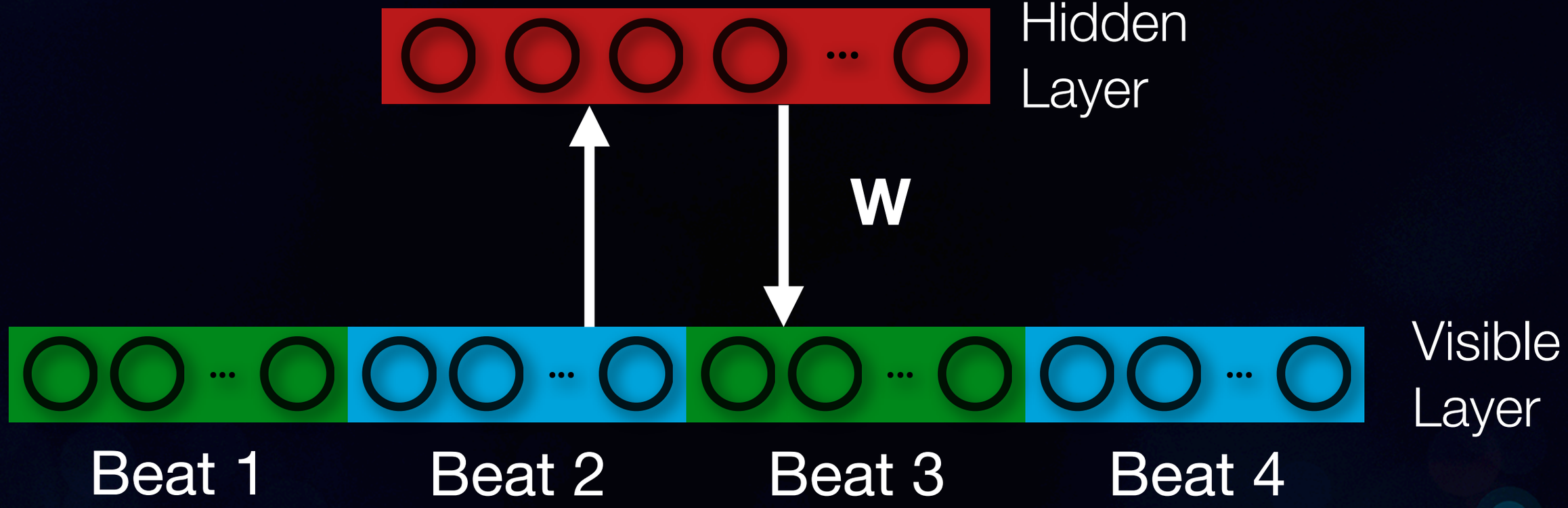
Learning rhythmic patterns and melodic contour

- Rhythm and melody are musical quantities that are understood to inform emotion
  - Learn informative domains from pre-training alone
  - Harmonic percussive/source separation
    - Parse rhythmic from melodic contributions
  - Beat synchronous representations
    - Parse musical events

E. M. Schmidt and Y. E. Kim (2013). Learning rhythm and melody features with deep belief networks. *ISMIR.*

Going Even Further

Learning rhythmic patterns and melodic contour



MediaEval Dataset

1000 songs for music emotion recognition

- 1000 songs for music emotion recognition
  - All songs selected from the Free Music Archive
    - Creative commons licensed
- Data is annotated using Amazon's Mechanical Turk
  - For each song a 40-second clip is analyzed
  - Each song is analyzed by a minimum of 10 participants
  - Continuous and discrete Valence-Arousal is collected

Joint work with Mohammad Soleymani, Yi-Hsuan Yang, Cheng-ya Sha, and Michael Caro

Thank You!

eschmidt@pandora.com