LEARNING REPRESENTATIONS OF SEQUENCES

IPAM GRADUATE SUMMER SCHOOL ON DEEP LEARNING

GRAHAM TAYLOR

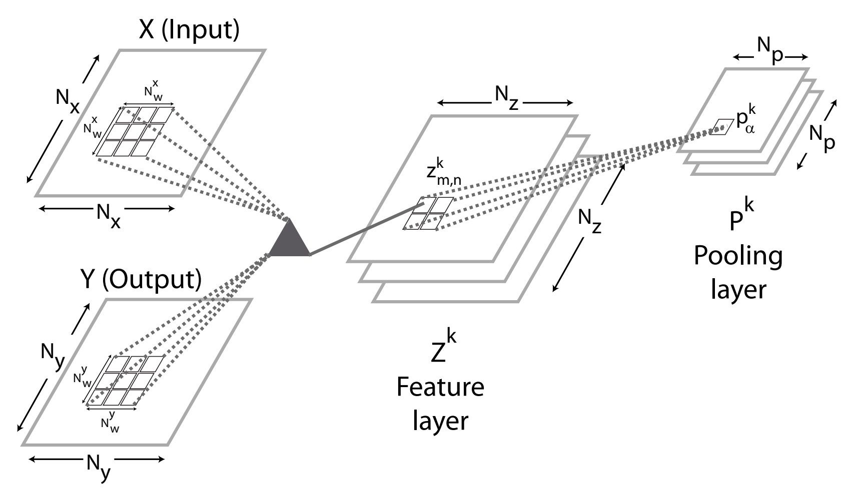
SCHOOL OF ENGINEERING
UNIVERSITY OF GUELPH

Papers and software available at: http://www.uoguelph.ca/~gwtaylor

OVERVIEW: THIS TALK

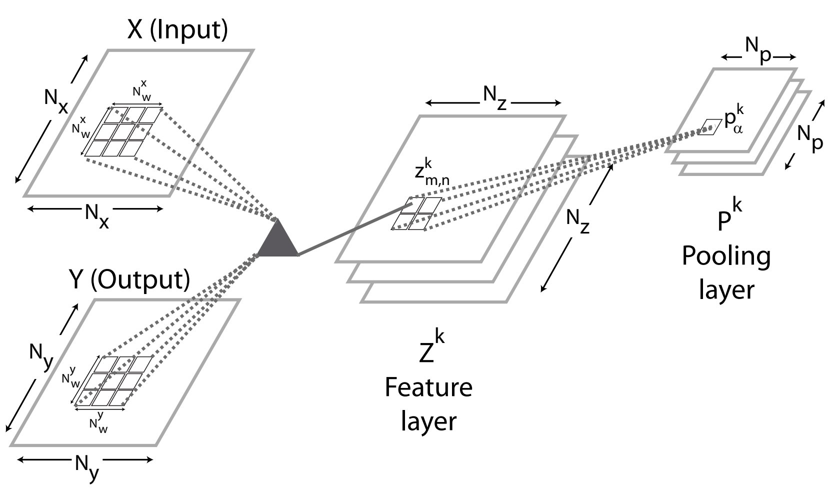
OVERVIEW: THIS TALK

- Learning representations of temporal data:
  - existing methods and challenges faced
  - recent methods inspired by deep learning and representation learning

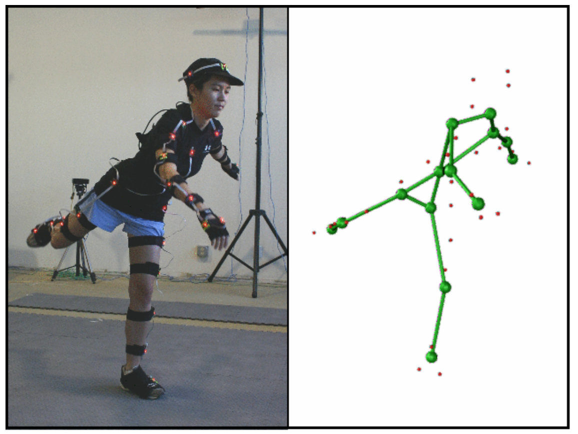


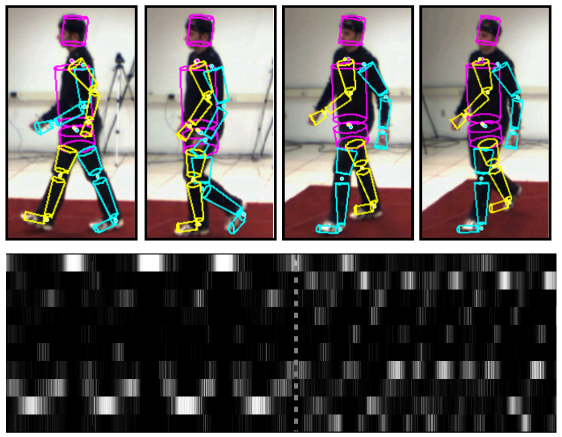
OVERVIEW: THIS TALK

- Learning representations of temporal data:
  - existing methods and challenges faced
  - recent methods inspired by deep learning and representation learning



- Applications: in particular, modeling human pose and activity
  - highly structured data: e.g. motion capture
  - weakly structured data: e.g. video



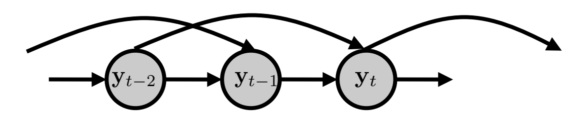


OUTLINE

OUTLINE

Learning representations from sequences

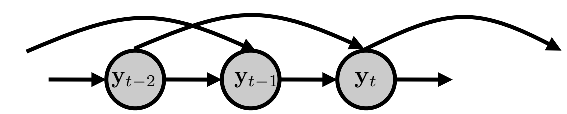
Existing methods, challenges



OUTLINE

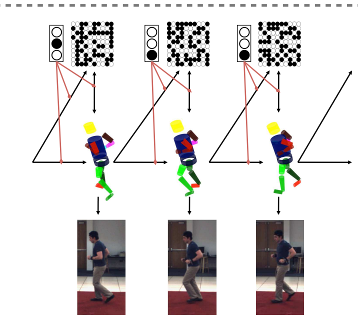
Learning representations from sequences

Existing methods, challenges



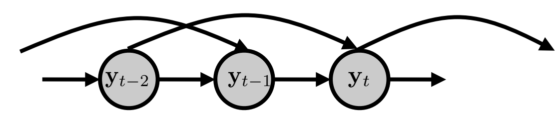
Composable, distributed-state models for sequences

Conditional Restricted Boltzmann Machines and their variants

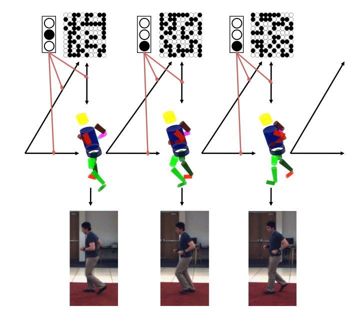


OUTLINE

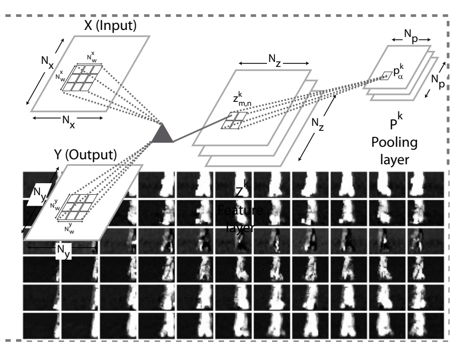
Learning representations from sequences
Existing methods, challenges



Composable, distributed-state models for sequences
Conditional Restricted Boltzmann Machines and their variants

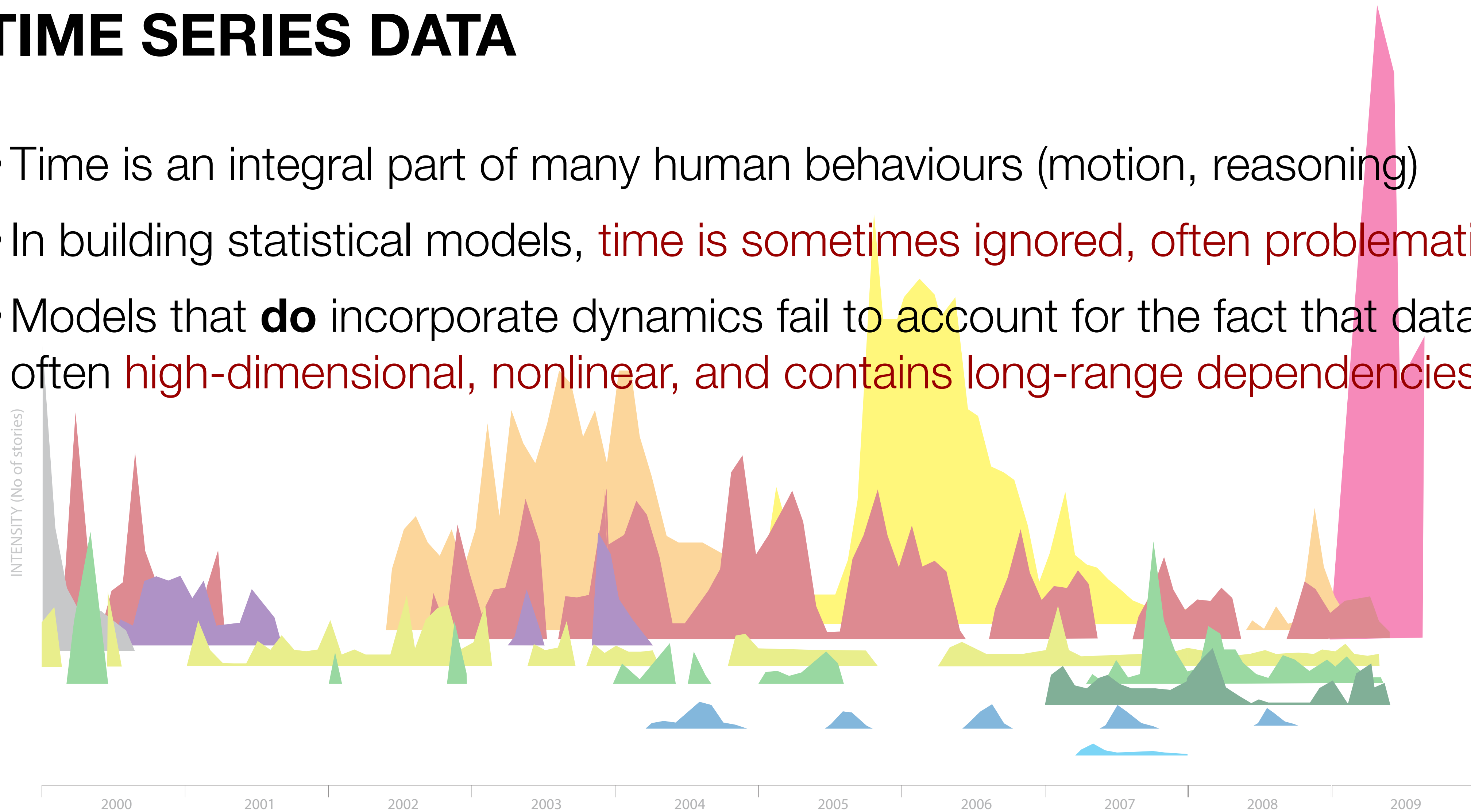


Using learned representations to analyze video
A brief and (incomplete survey of deep learning for activity recognition



TIME SERIES DATA

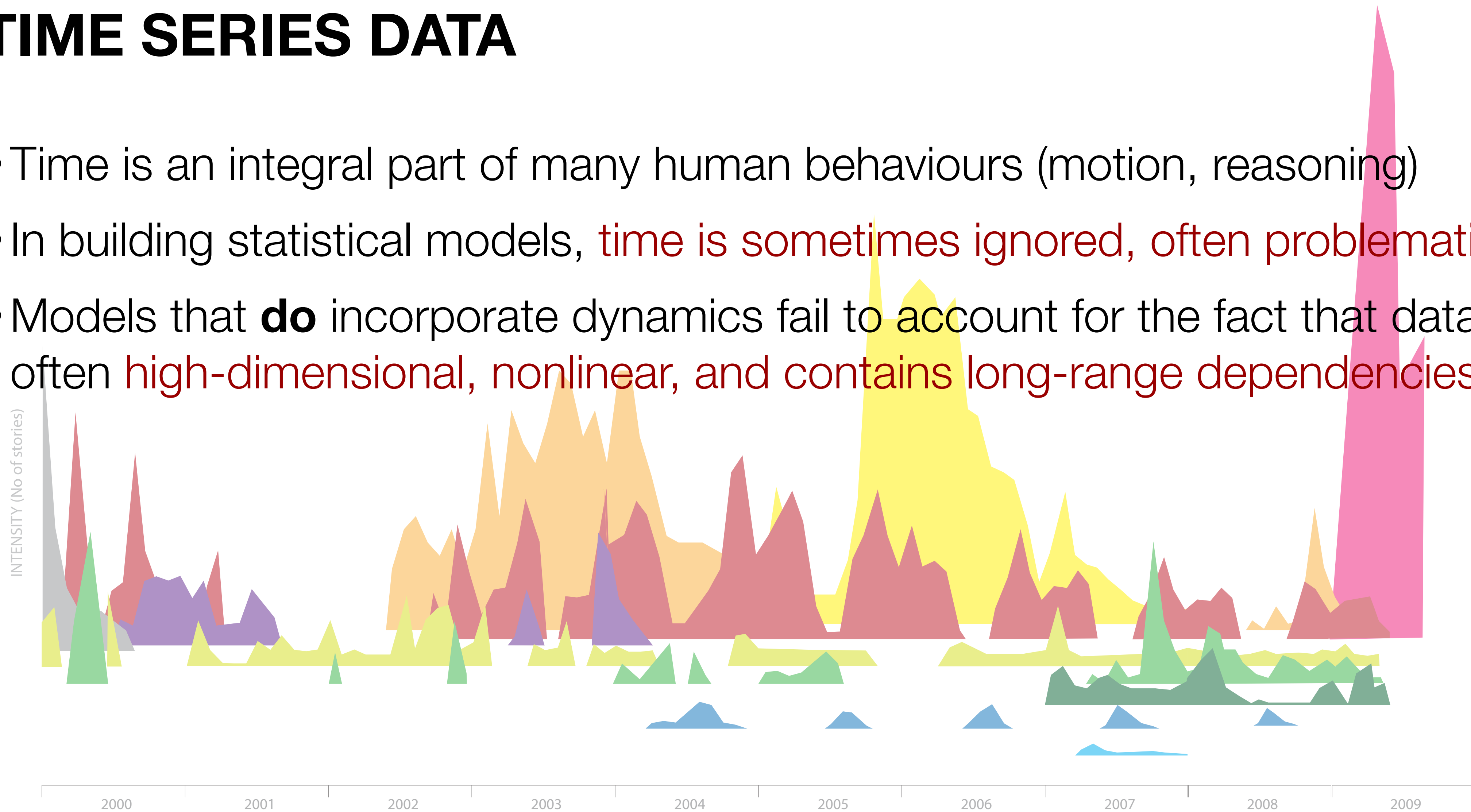
- Time is an integral part of many human behaviours (motion, reasoning)
- In building statistical models, time is sometimes ignored, often problematic
- Models that **do** incorporate dynamics fail to account for the fact that data is often high-dimensional, nonlinear, and contains long-range dependencies



Graphic: David McCandless, informationisbeautiful.net

TIME SERIES DATA

- Time is an integral part of many human behaviours (motion, reasoning)
- In building statistical models, time is sometimes ignored, often problematic
- Models that **do** incorporate dynamics fail to account for the fact that data is often high-dimensional, nonlinear, and contains long-range dependencies



Today we will discuss a number of models that have been developed to address these challenges.

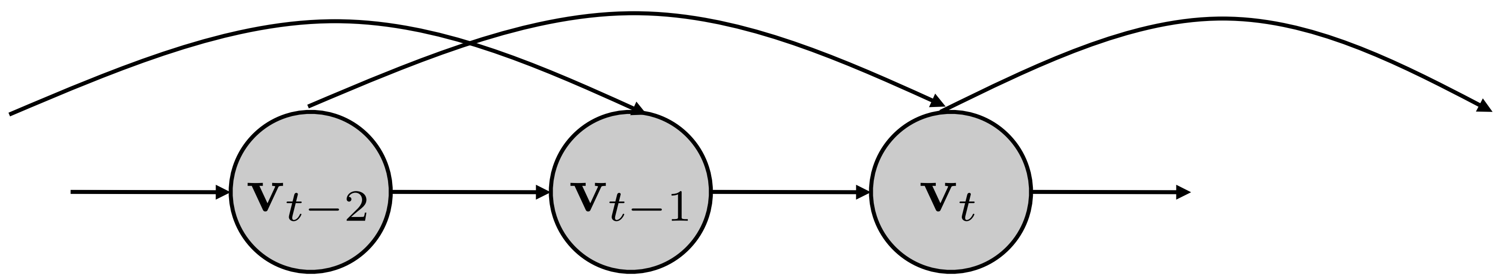
Graphic: David McCandless, informationisbeautiful.net

VECTOR AUTOREGRESSIVE MODELS

$$\mathbf{v}_t = \mathbf{b} + \sum_{m=1}^{M} A_m \mathbf{v}_{t-m} + \mathbf{e}_t$$

- Have dominated statistical time-series analysis for approx. 50 years
- Can be fit easily by least-squares regression
- Can fail even for simple nonlinearities present in the system
  - but many data sets can be modeled well by a linear system
- Well understood; many extensions exist

MARKOV ("N-GRAM") MODELS



- Fully observable
- Sequential observations may have nonlinear dependence
- Derived by assuming sequences have Markov property:

$$p(\mathbf{v}_t|\{\mathbf{v}_1^{t-1}\}) = p(\mathbf{v}_t|\{\mathbf{v}_{t-N}^{t-1}\})$$

- This leads to joint: $p(\{\mathbf{v}_1^T\}) = p(\{\mathbf{v}_1^N\}) \prod_{t=N+1}^{T} p(\mathbf{v}_t|\{\mathbf{v}_{t-N}^{t-1}\})$
- Number of parameters exponential in N !

EXPONENTIAL INCREASE IN PARAMETERS

$$|\theta| = Q^{N+1}$$

Here, $Q = 3$

p(a|a)	p(b|a)	p(c|a)
p(a|b)	p(b|b)	p(c|b)
p(a|c)	p(b|c)	p(c|c)

1st order Markov $(N = 1)$

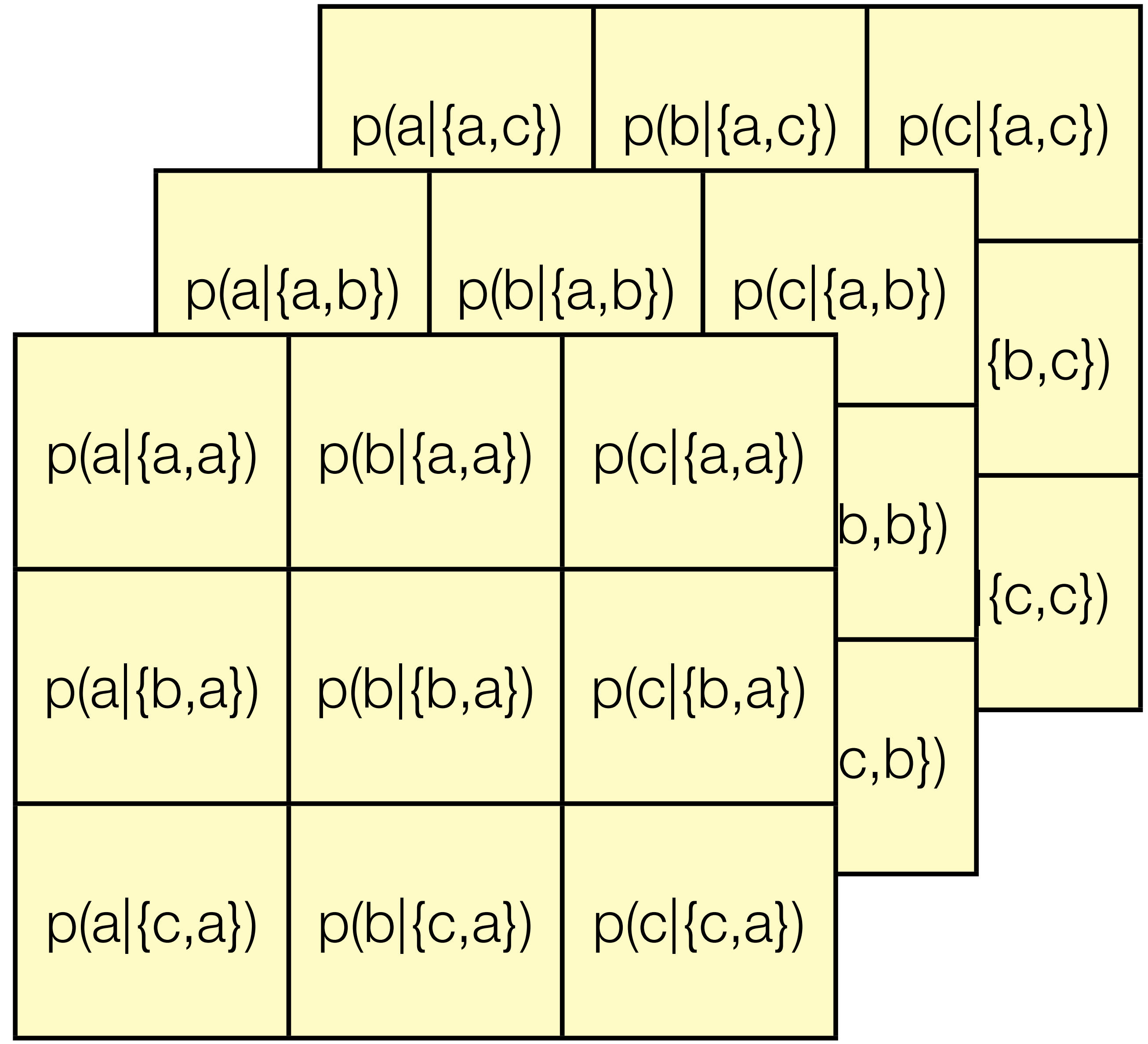
EXPONENTIAL INCREASE IN PARAMETERS

$$|\theta| = Q^{N+1}$$

Here, $Q = 3$

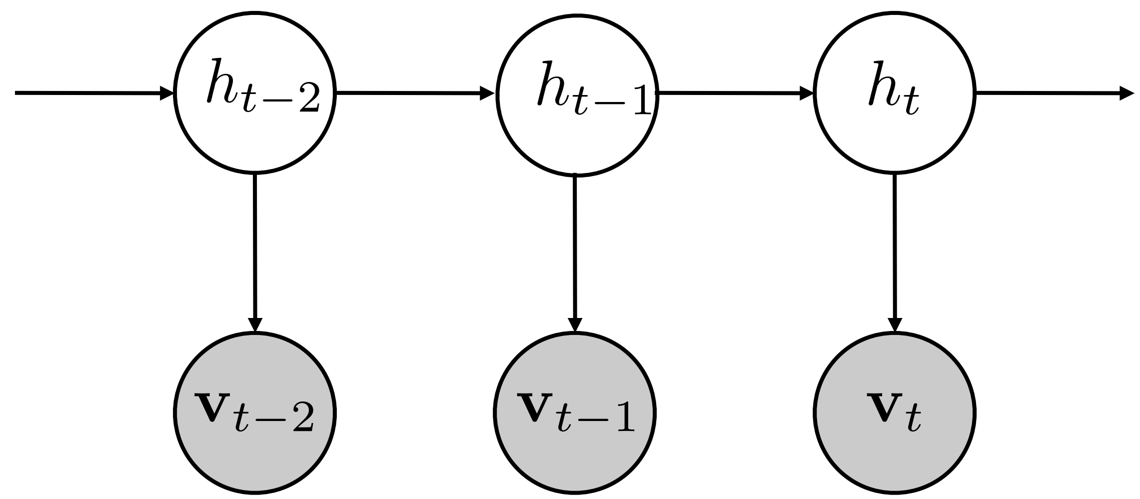
p(a|a)	p(b|a)	p(c|a)
p(a|b)	p(b|b)	p(c|b)
p(a|c)	p(b|c)	p(c|c)

1st order Markov $(N = 1)$



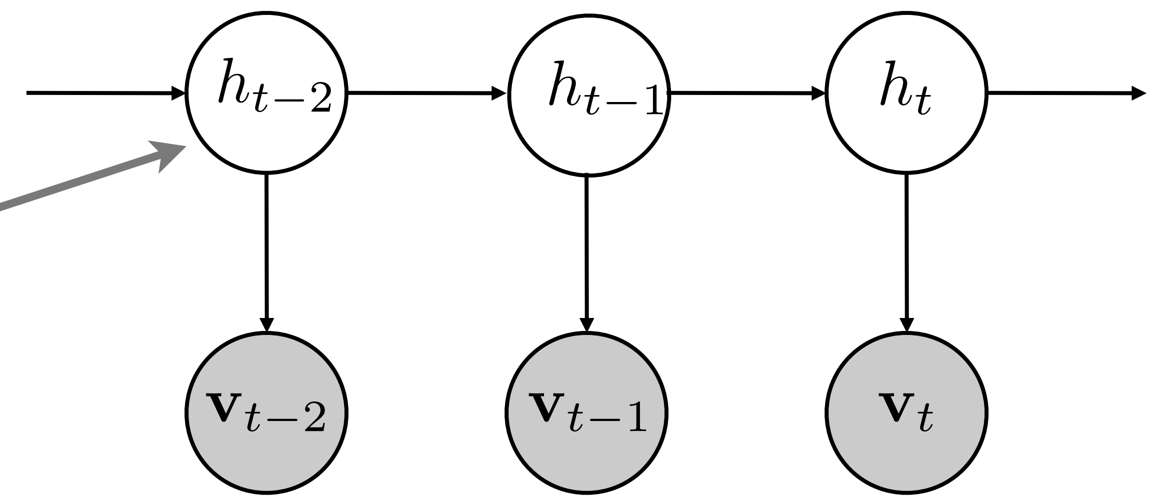
2nd order Markov $(N = 2)$

HIDDEN MARKOV MODELS (HMM)



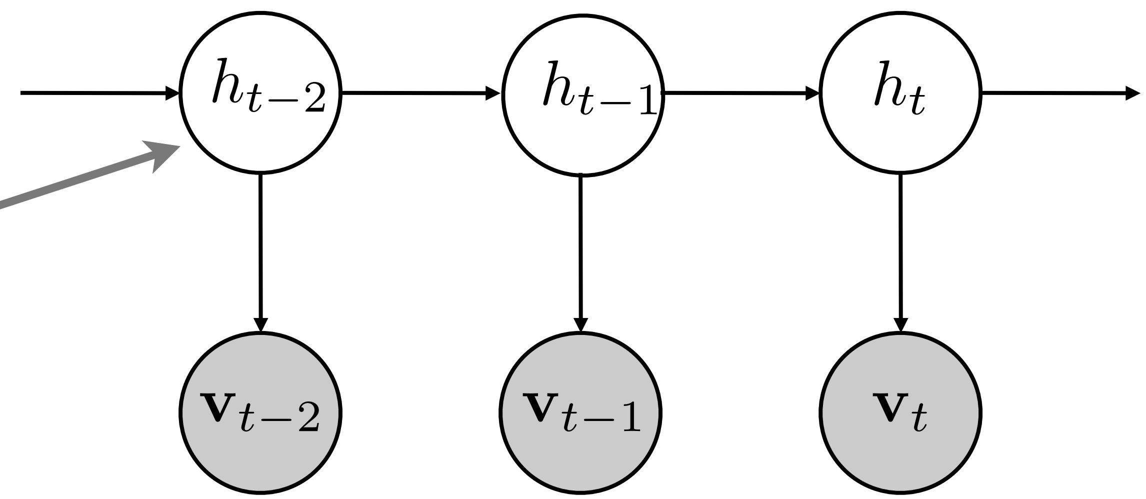
HIDDEN MARKOV MODELS (HMM)

Introduces a hidden state that controls the dependence of the current observation on the past



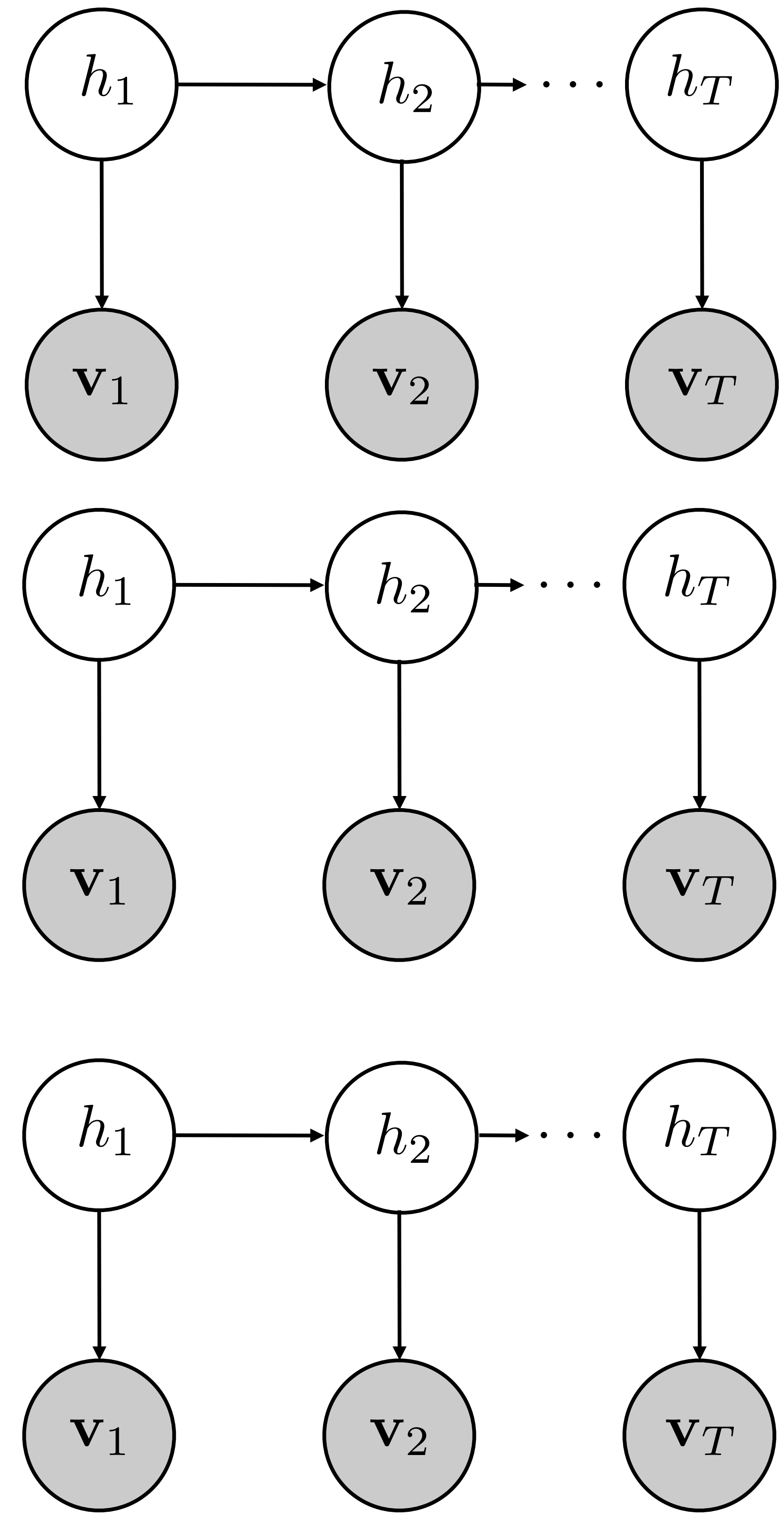
HIDDEN MARKOV MODELS (HMM)

Introduces a hidden state that controls the dependence of the current observation on the past



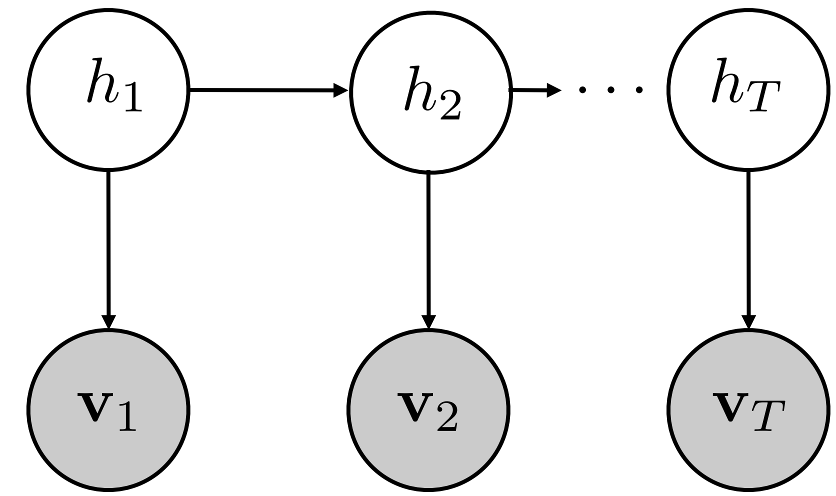
- Successful in speech & language modeling, biology
- Defined by 3 sets of parameters:
  - Initial state parameters, π
  - Transition matrix, A
  - Emission distribution, $p(\mathbf{v}_t|h_t)$
- Factored joint distribution: $p(\{h_t\}, \{\mathbf{v}_t\}) = p(h_1)p(\mathbf{v}_1|h_1)\prod_{t=2}^{T} p(h_t|h_{t-1})p(\mathbf{v}_t|h_t)$

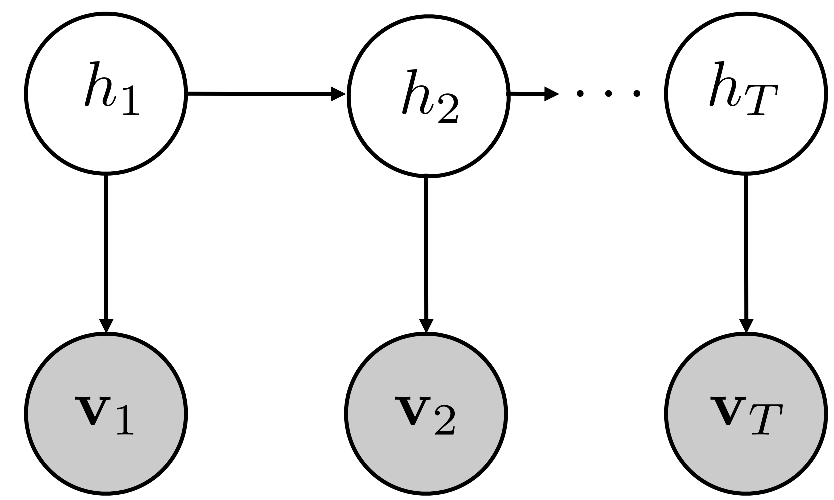
INFERENCE AND LEARNING

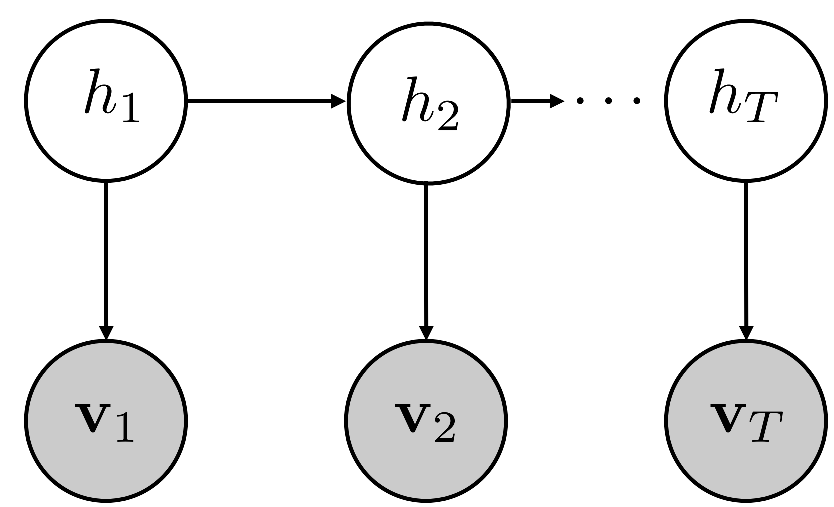


INFERENCE AND LEARNING

- Typically three tasks we want to perform in an HMM:

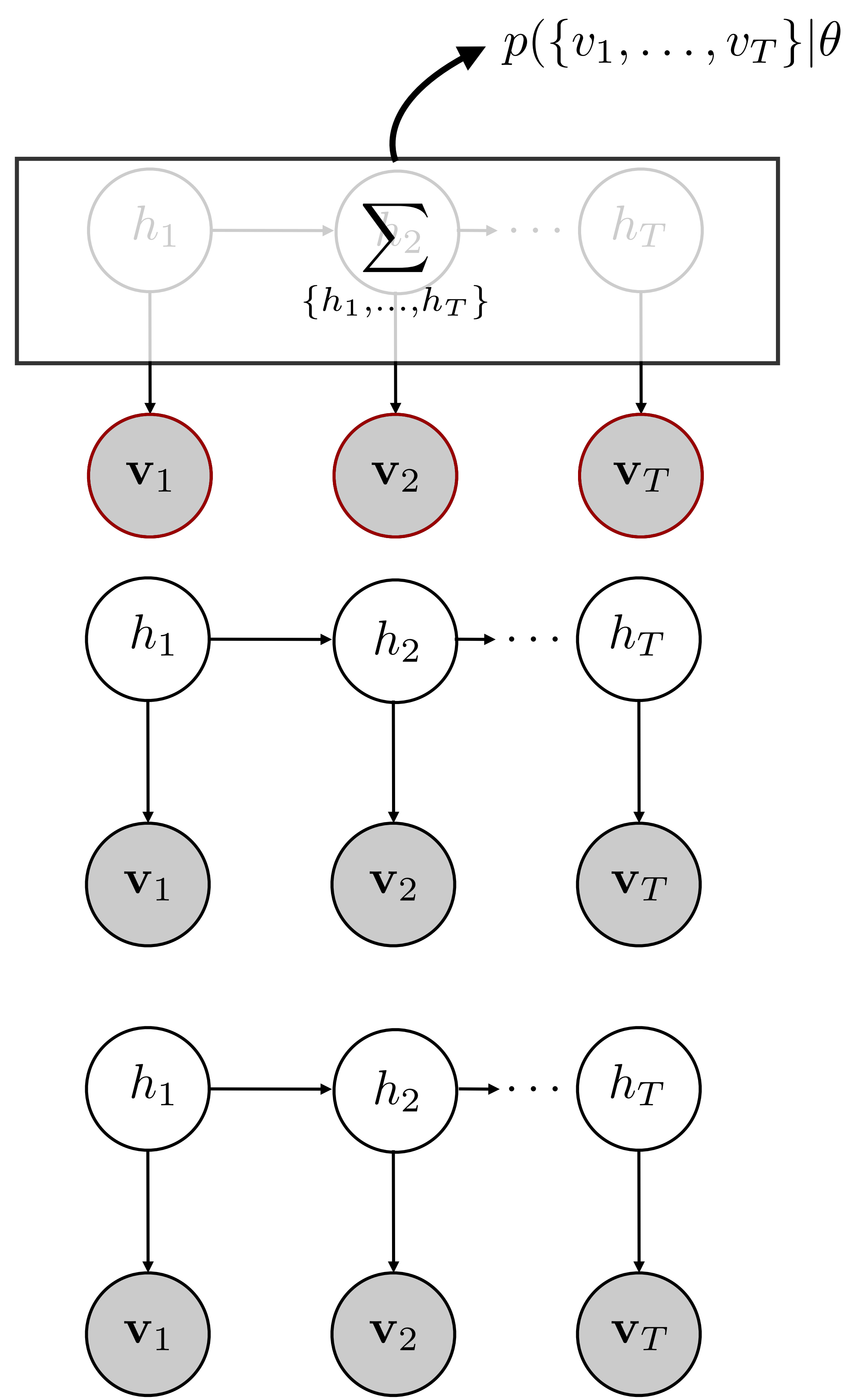






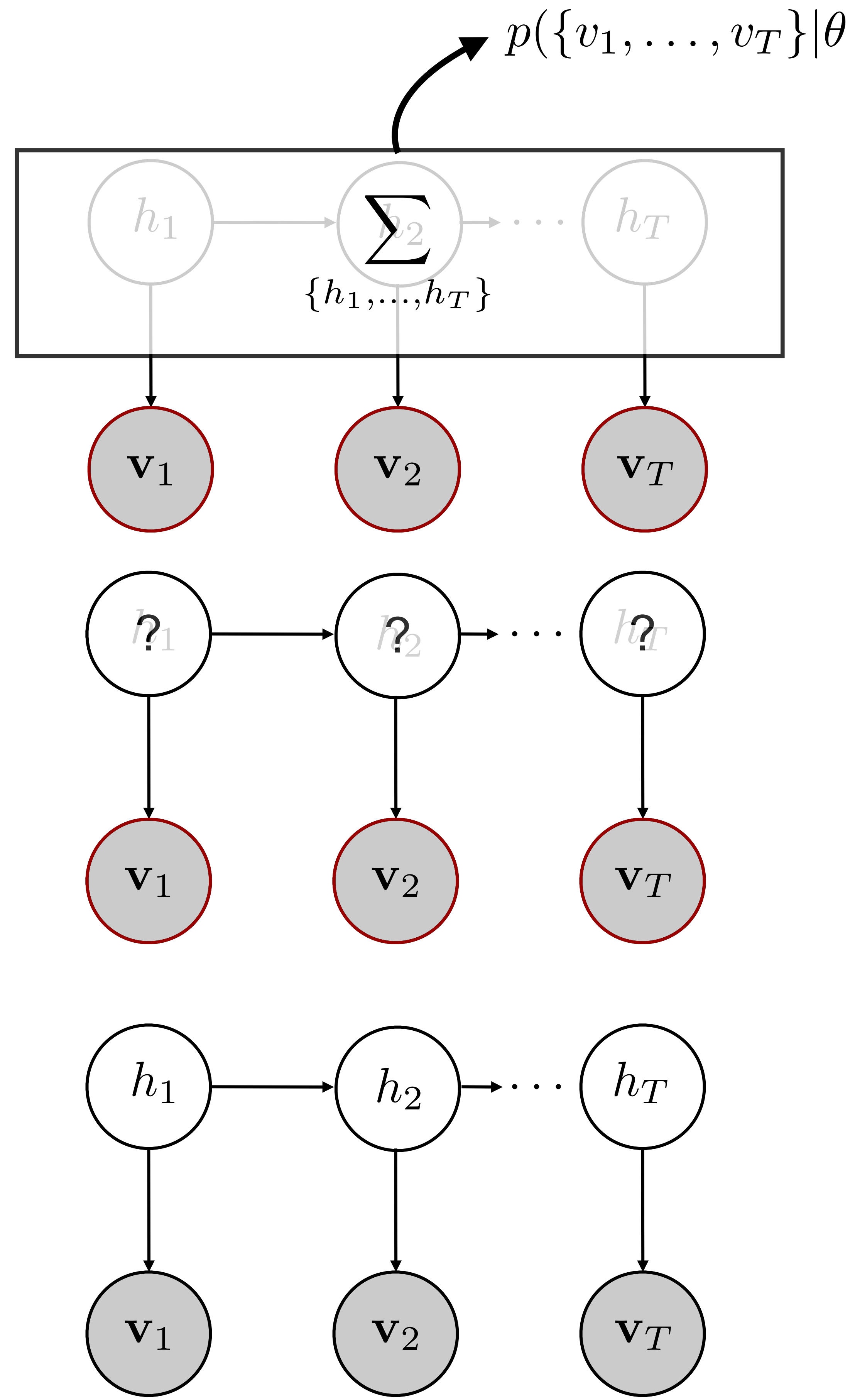
INFERENCE AND LEARNING

- Typically three tasks we want to perform in an HMM:
  - Likelihood estimation



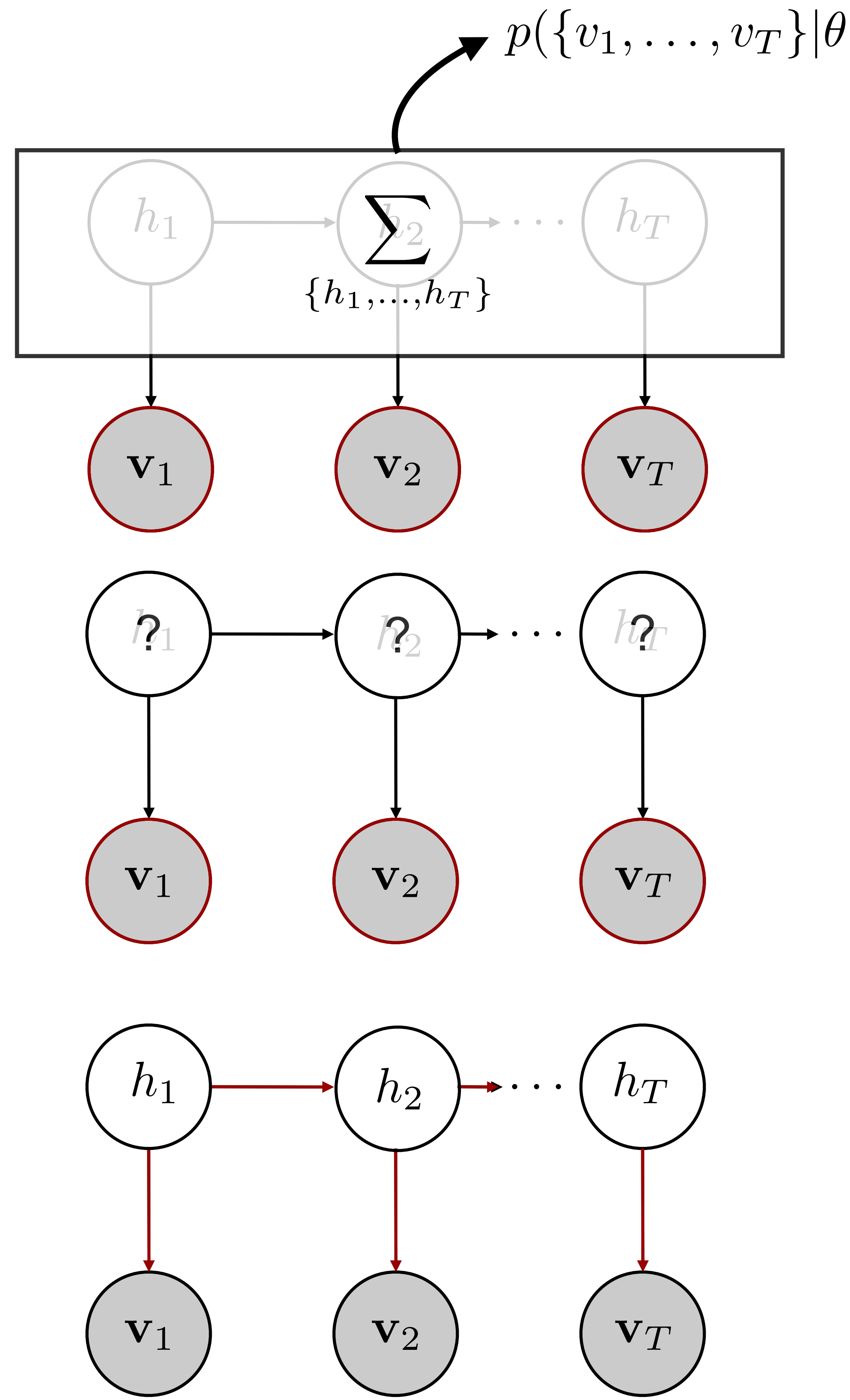
INFERENCE AND LEARNING

- Typically three tasks we want to perform in an HMM:
  - Likelihood estimation
  - Inference



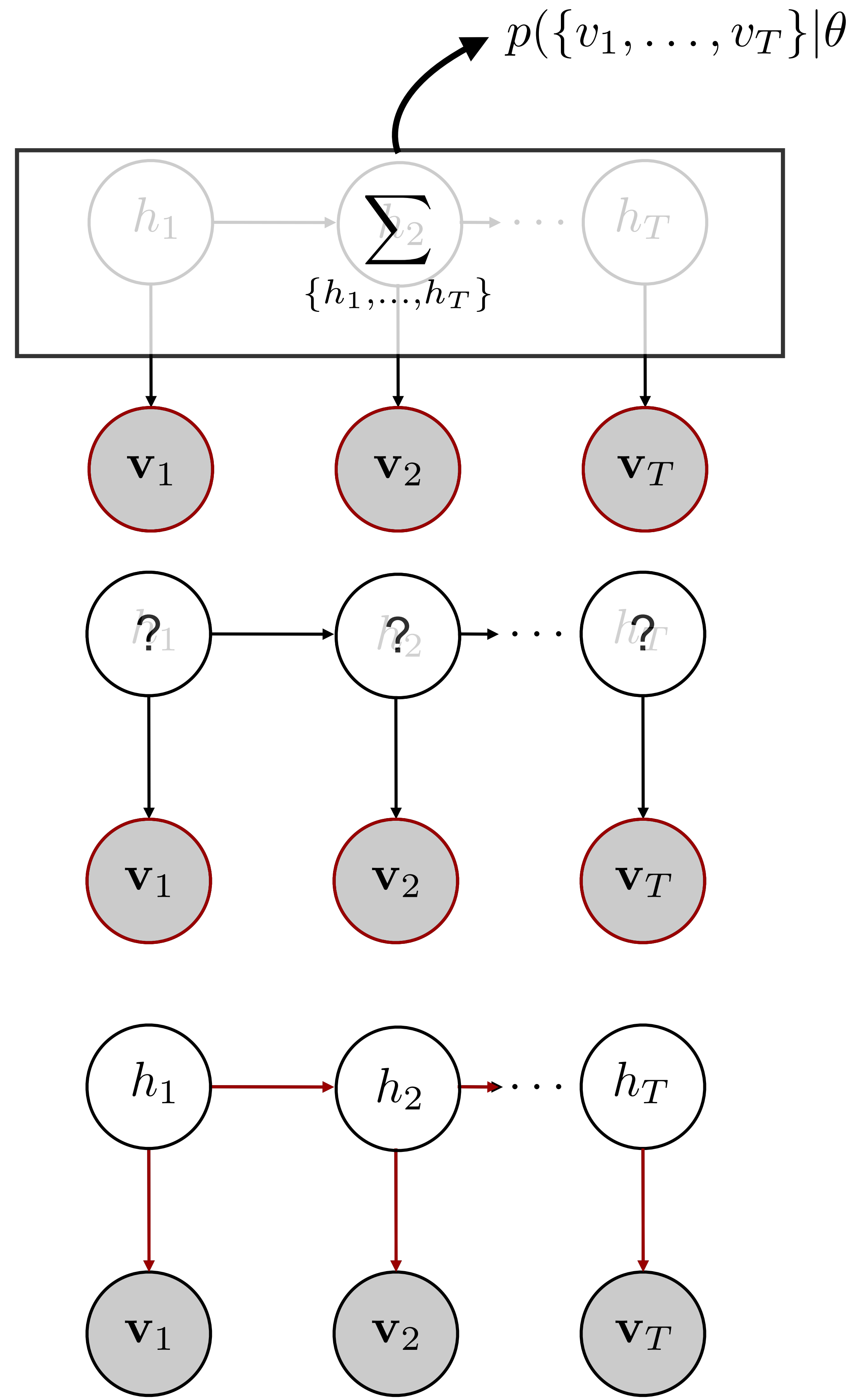
INFERENCE AND LEARNING

- Typically three tasks we want to perform in an HMM:
  - Likelihood estimation
  - Inference
  - Learning



INFERENCE AND LEARNING

- Typically three tasks we want to perform in an HMM:
  - Likelihood estimation
  - Inference
  - Learning
- All are exact and tractable due to the simple structure of the HMM



LIMITATIONS OF HMMS

LIMITATIONS OF HMMS

- Many high-dimensional data sets contain rich componential structure

LIMITATIONS OF HMMS

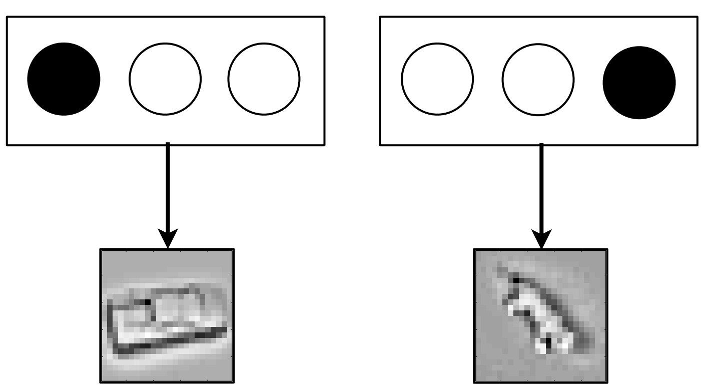
- Many high-dimensional data sets contain rich componential structure
- Hidden Markov Models cannot model such data efficiently: a single, discrete K-state multinomial must represent the history of the time series

LIMITATIONS OF HMMS

- Many high-dimensional data sets contain rich componential structure
- Hidden Markov Models cannot model such data efficiently: a single, discrete K-state multinomial must represent the history of the time series
- To model K bits of information, they need 2^K hidden states

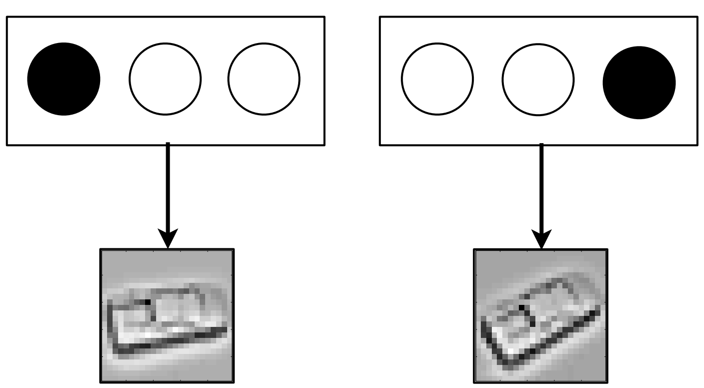
LIMITATIONS OF HMMS

- Many high-dimensional data sets contain rich componential structure
- Hidden Markov Models cannot model such data efficiently: a single, discrete K-state multinomial must represent the history of the time series
- To model K bits of information, they need 2^K hidden states



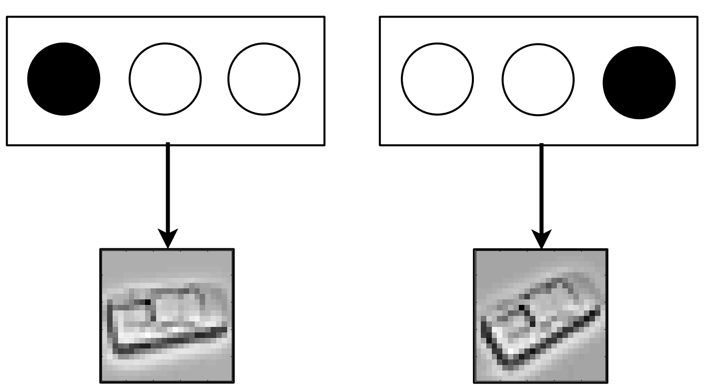
LIMITATIONS OF HMMS

- Many high-dimensional data sets contain rich componential structure
- Hidden Markov Models cannot model such data efficiently: a single, discrete K-state multinomial must represent the history of the time series
- To model K bits of information, they need 2^K hidden states



LIMITATIONS OF HMMS

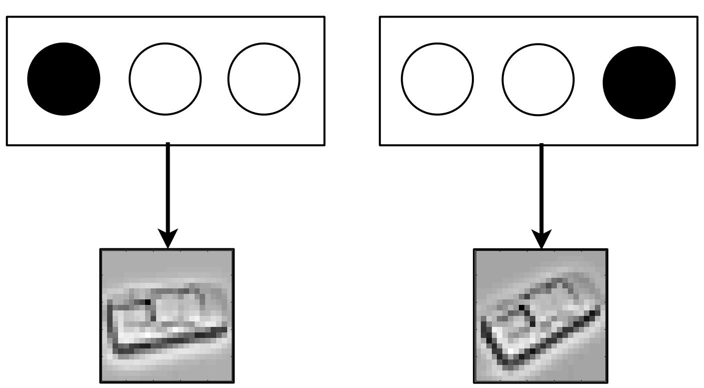
- Many high-dimensional data sets contain rich componential structure
- Hidden Markov Models cannot model such data efficiently: a single, discrete K-state multinomial must represent the history of the time series
- To model K bits of information, they need 2^K hidden states
- We seek models with distributed hidden state:



LIMITATIONS OF HMMS

- Many high-dimensional data sets contain rich componential structure
- Hidden Markov Models cannot model such data efficiently: a single, discrete K-state multinomial must represent the history of the time series
- To model K bits of information, they need 2^K hidden states
- We seek models with distributed hidden state:

- capacity linear in the number of components

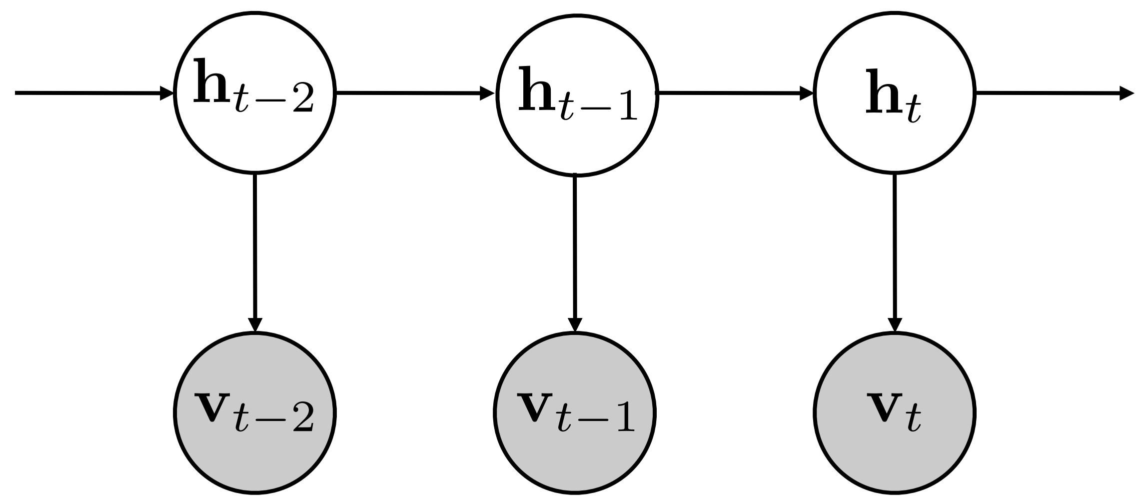


LIMITATIONS OF HMMS

- Many high-dimensional data sets contain rich componential structure
- Hidden Markov Models cannot model such data efficiently: a single, discrete K-state multinomial must represent the history of the time series
- To model K bits of information, they need 2^K hidden states
- We seek models with distributed hidden state:

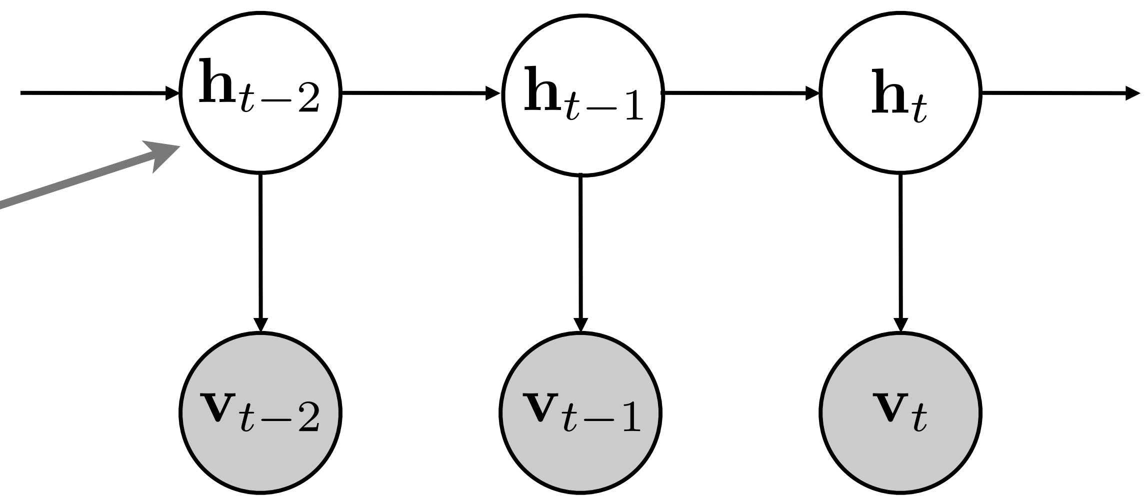
 - capacity linear in the number of components

LINEAR DYNAMICAL SYSTEMS



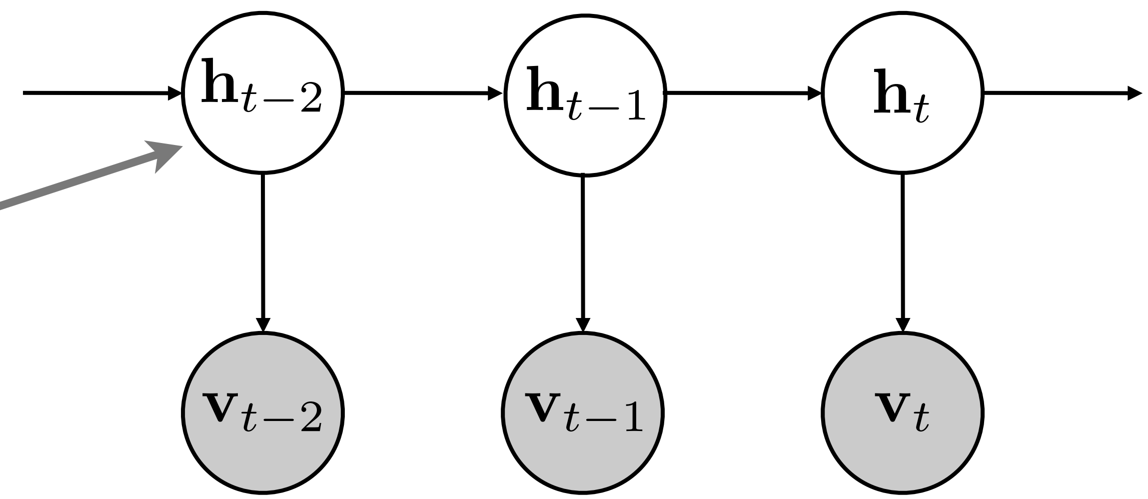
LINEAR DYNAMICAL SYSTEMS

Graphical model is the same as HMM but with real-valued state *vectors*



LINEAR DYNAMICAL SYSTEMS

Graphical model is the same as HMM but with real-valued state *vectors*



- Characterized by linear-Gaussian dynamics and observations:

$$p(\mathbf{h}_t|\mathbf{h}_{t-1}) = \mathcal{N}(\mathbf{h}_t; A\mathbf{h}_{t-1}, Q) \qquad p(\mathbf{v}_t|\mathbf{h}_t) = \mathcal{N}(\mathbf{v}_t; C\mathbf{h}_t, R)$$

- Inference is performed using Kalman smoothing (belief propagation)
- Learning can be done by EM
- Dynamics, observations may also depend on an observed input (control)

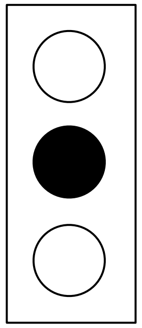
LATENT REPRESENTATIONS FOR REAL-WORLD DATA

Data for many real-world problems (e.g. vision, motion capture) is high-dimensional, containing complex non-linear relationships between components

LATENT REPRESENTATIONS FOR REAL-WORLD DATA

Data for many real-world problems (e.g. vision, motion capture) is high-dimensional, containing complex non-linear relationships between components

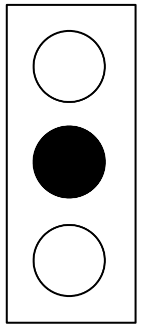
Hidden Markov Models
Pro: complex, nonlinear emission model
Con: single K-state multinomial represents entire history



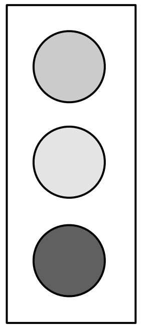
LATENT REPRESENTATIONS FOR REAL-WORLD DATA

Data for many real-world problems (e.g. vision, motion capture) is high-dimensional, containing complex non-linear relationships between components

Hidden Markov Models
Pro: complex, nonlinear emission model
Con: single K -state multinomial represents entire history



Linear Dynamical Systems
Pro: state can convey much more information
Con: emission model constrained to be linear



LEARNING DISTRIBUTED REPRESENTATIONS

LEARNING DISTRIBUTED REPRESENTATIONS

- Simple networks are capable of discovering useful and interesting internal representations of static data (e.g. many of the talks so far!)

LEARNING DISTRIBUTED REPRESENTATIONS

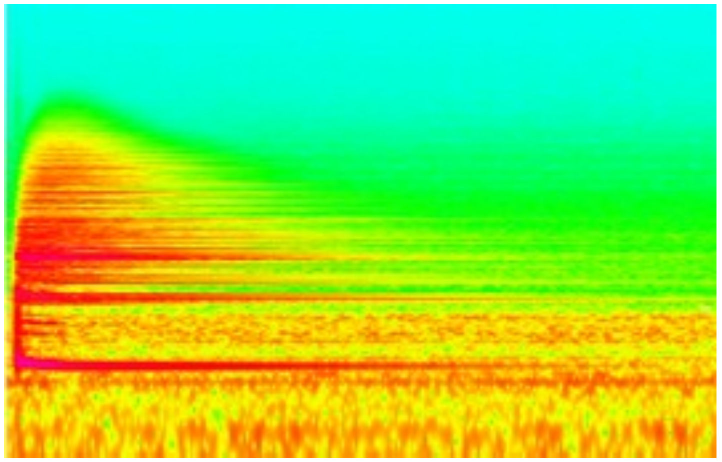
- Simple networks are capable of discovering useful and interesting internal representations of static data (e.g. many of the talks so far!)
- Can we learn, in a similar way, representations of temporal data?

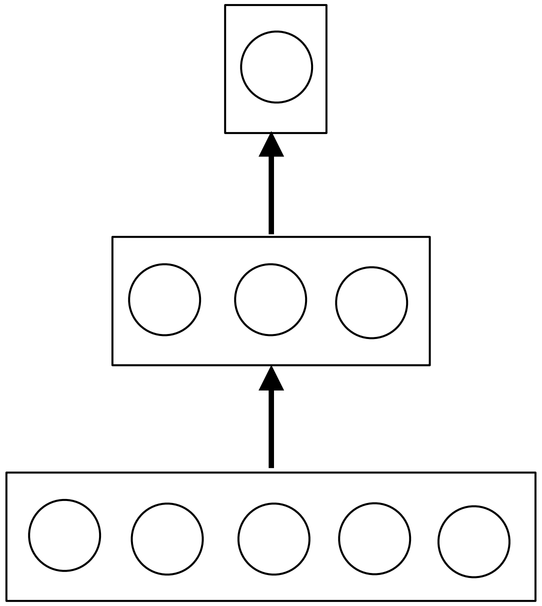
LEARNING DISTRIBUTED REPRESENTATIONS

- Simple networks are capable of discovering useful and interesting internal representations of static data (e.g. many of the talks so far!)
- Can we learn, in a similar way, representations of temporal data?
- Simple idea: spatial representation of time:

LEARNING DISTRIBUTED REPRESENTATIONS

- Simple networks are capable of discovering useful and interesting internal representations of static data (e.g. many of the talks so far!)
- Can we learn, in a similar way, representations of temporal data?
- Simple idea: spatial representation of time:

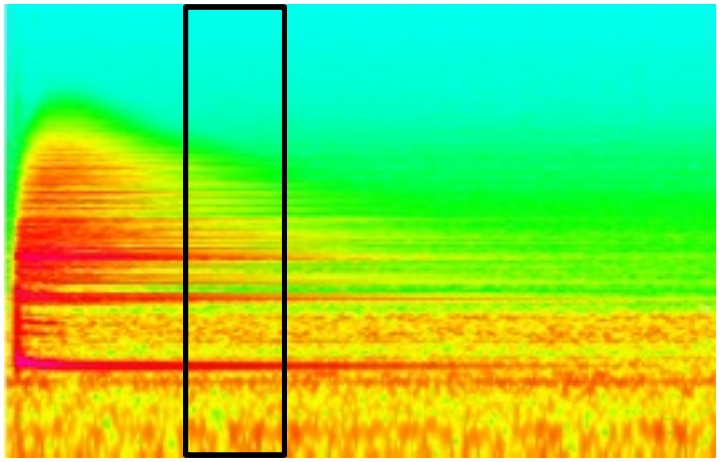


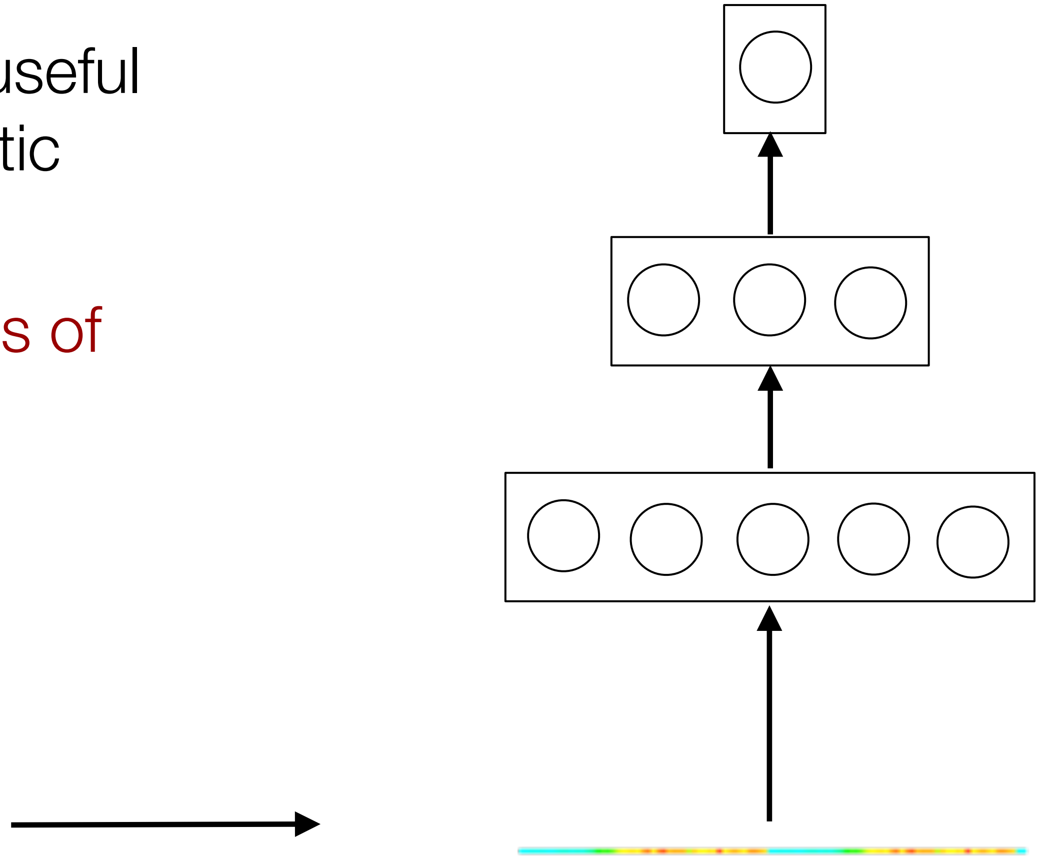


Spectrogram: http://soundsofstanford.wordpress.com

LEARNING DISTRIBUTED REPRESENTATIONS

- Simple networks are capable of discovering useful and interesting internal representations of static data (e.g. many of the talks so far!)
- Can we learn, in a similar way, representations of temporal data?
- Simple idea: spatial representation of time:

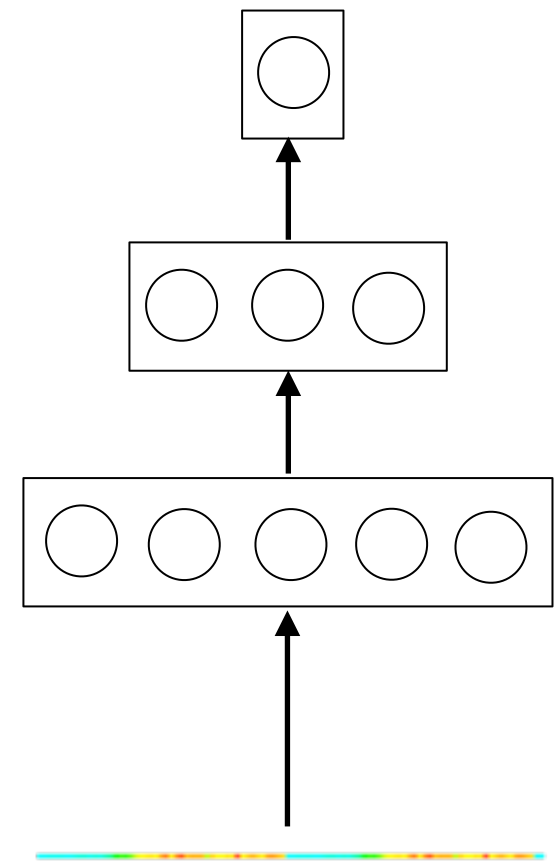




Spectrogram: http://soundsofstanford.wordpress.com

LEARNING DISTRIBUTED REPRESENTATIONS

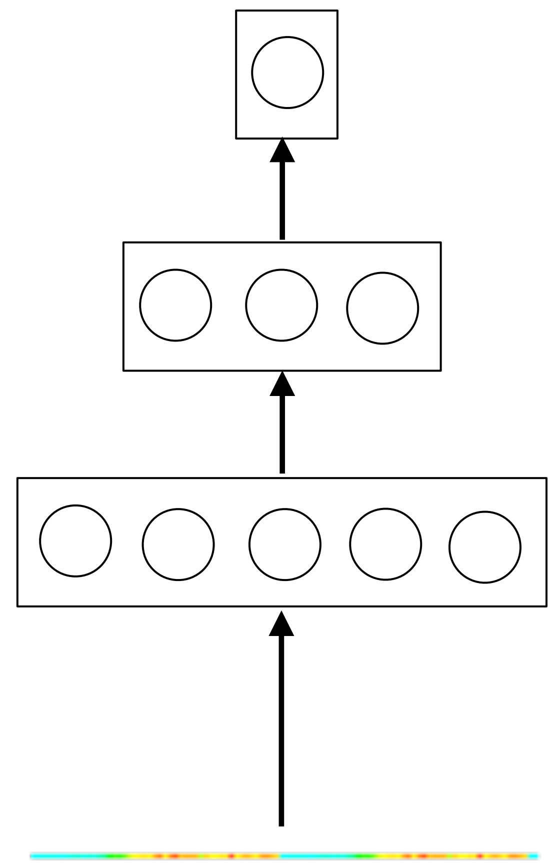
- Simple networks are capable of discovering useful and interesting internal representations of static data (e.g. many of the talks so far!)
- Can we learn, in a similar way, representations of temporal data?
- Simple idea: spatial representation of time:



Spectrogram: http://soundsofstanford.wordpress.com

LEARNING DISTRIBUTED REPRESENTATIONS

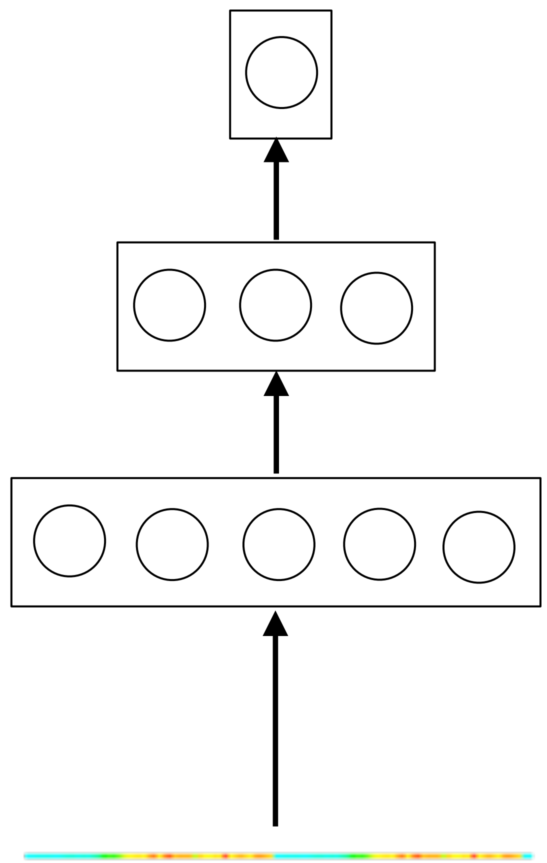
- Simple networks are capable of discovering useful and interesting internal representations of static data (e.g. many of the talks so far!)
- Can we learn, in a similar way, representations of temporal data?
- Simple idea: spatial representation of time:
  - Need a buffer; not biologically plausible
  - Cannot process inputs of differing length
  - Cannot distinguish between absolute and relative position



Spectrogram: http://soundsofstanford.wordpress.com

LEARNING DISTRIBUTED REPRESENTATIONS

- Simple networks are capable of discovering useful and interesting internal representations of static data (e.g. many of the talks so far!)
- Can we learn, in a similar way, representations of temporal data?
- Simple idea: spatial representation of time:
  - Need a buffer; not biologically plausible
  - Cannot process inputs of differing length
  - Cannot distinguish between absolute and relative position
- This motivates an **implicit** representation of time in connectionist models where time is represented by its effect on processing



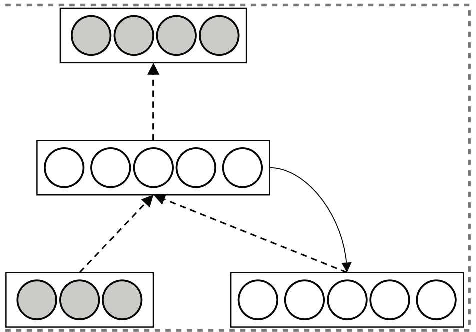
Spectrogram: http://soundsofstanford.wordpress.com

A RICH HISTORY

A RICH HISTORY

Elman networks

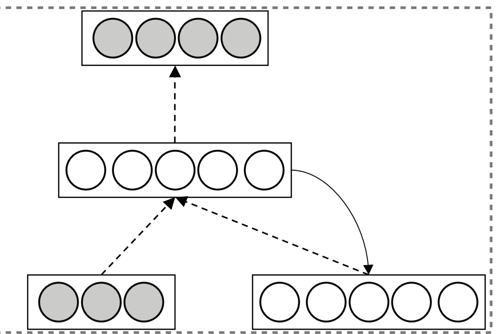
Time-delayed “context” units, truncated BPTT.
(Elman, 1990), (Jordan, 1986)



A RICH HISTORY

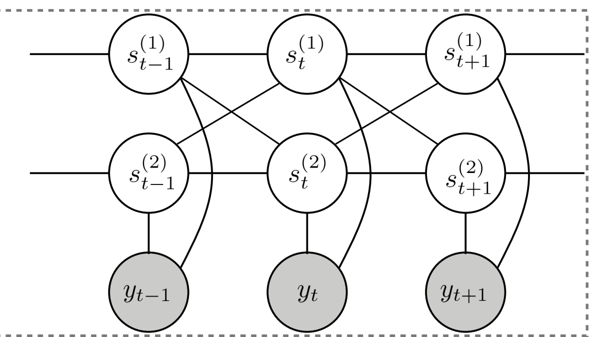
Elman networks

Time-delayed "context" units, truncated BPTT.
(Elman, 1990), (Jordan, 1986)



Mean-field Boltzmann Machines through Time

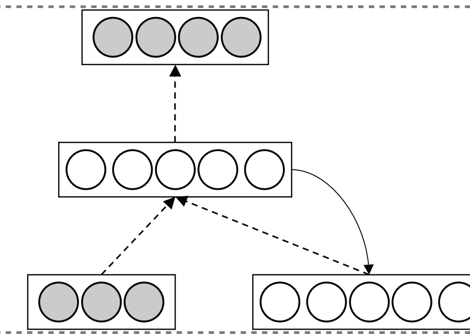
Inference is approximate, learning less efficient than HMMs.
(Williams and Hinton, 1990)



A RICH HISTORY

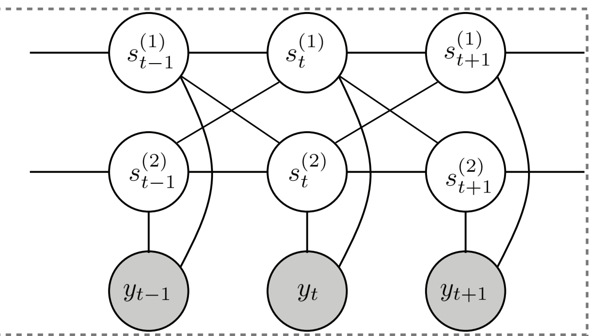
Elman networks

Time-delayed “context” units, truncated BPTT.
(Elman, 1990), (Jordan, 1986)



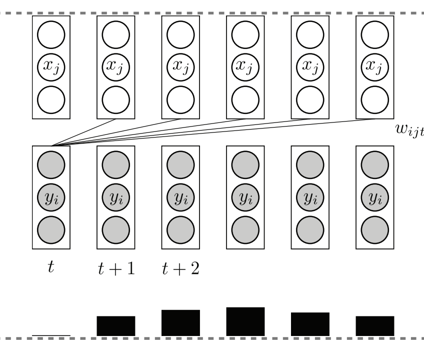
Mean-field Boltzmann Machines through Time

Inference is approximate, learning less efficient than HMMs.
(Williams and Hinton, 1990)

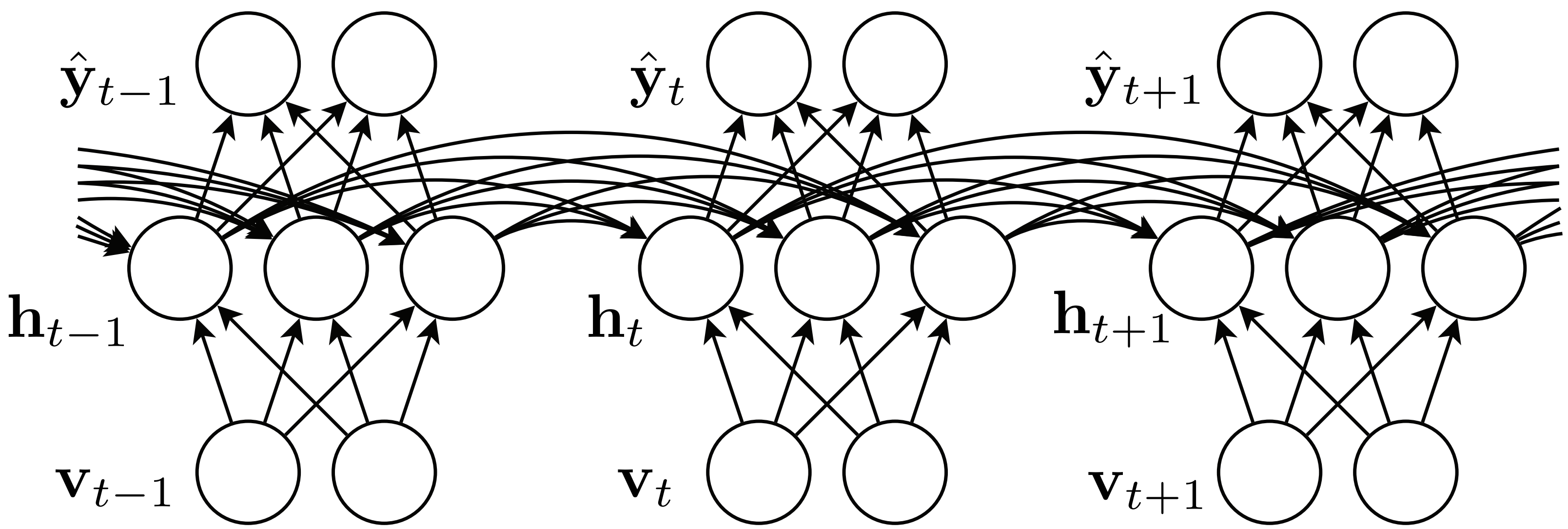


Spiking Boltzmann Machines

Hidden-state dynamics and smoothness constraints on observed data.
(Hinton and Brown, 2000)



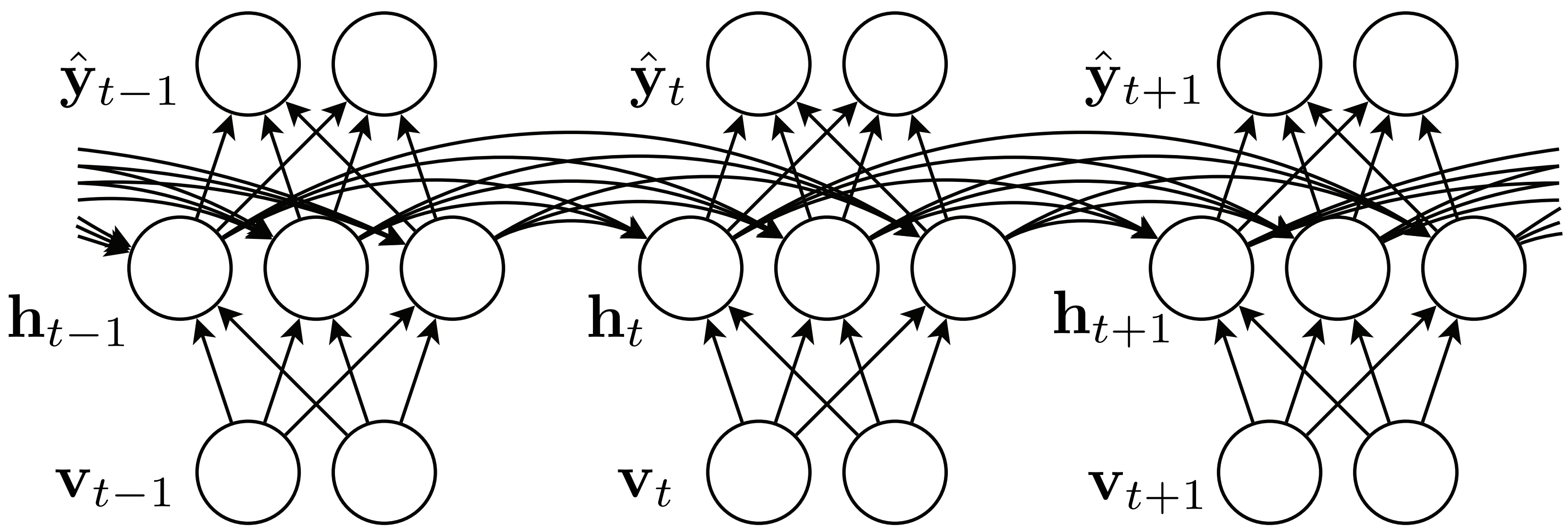
RECURRENT NEURAL NETWORKS



$$
\begin{aligned}
\mathbf{x}_t &= W^{hv}\mathbf{v}_t + W^{hh}\mathbf{h}_{t-1} + \mathbf{b}_h \\
h_{j,t} &= f(x_{j,t}) \\
\mathbf{s}_t &= W^{yh}\mathbf{h}_t + \mathbf{b}_y \\
\hat{y}_{k,t} &= g(s_{k,t})
\end{aligned}
$$

(Figure from Martens and Sutskever)

RECURRENT NEURAL NETWORKS

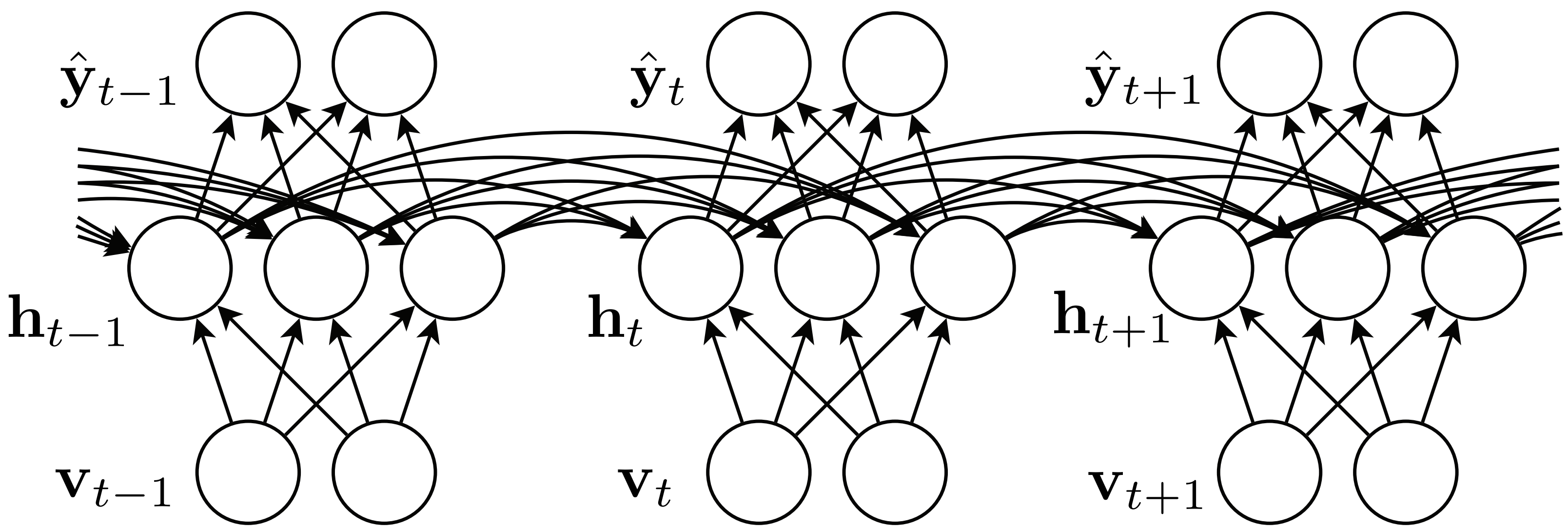


- Neural network replicated in time

$$
\begin{aligned}
\mathbf{x}_t &= W^{hv}\mathbf{v}_t + W^{hh}\mathbf{h}_{t-1} + \mathbf{b}_h \\
h_{j,t} &= f(x_{j,t}) \\
\mathbf{s}_t &= W^{yh}\mathbf{h}_t + \mathbf{b}_y \\
\hat{y}_{k,t} &= g(s_{k,t})
\end{aligned}
$$

(Figure from Martens and Sutskever)

RECURRENT NEURAL NETWORKS



- Neural network replicated in time
- At each step, receives input vector, updates its internal representation via nonlinear activation functions, and makes a prediction:

$$
\begin{aligned}
\mathbf{x}_t &= W^{hv}\mathbf{v}_t + W^{hh}\mathbf{h}_{t-1} + \mathbf{b}_h \\
h_{j,t} &= f(x_{j,t}) \\
\mathbf{s}_t &= W^{yh}\mathbf{h}_t + \mathbf{b}_y \\
\hat{y}_{k,t} &= g(s_{k,t})
\end{aligned}
$$

(Figure from Martens and Sutskever)

TRAINING RECURRENT NEURAL NETWORKS

TRAINING RECURRENT NEURAL NETWORKS

- Possibly high-dimensional, distributed, internal representation and nonlinear dynamics allow model, in theory, model complex time series

TRAINING RECURRENT NEURAL NETWORKS

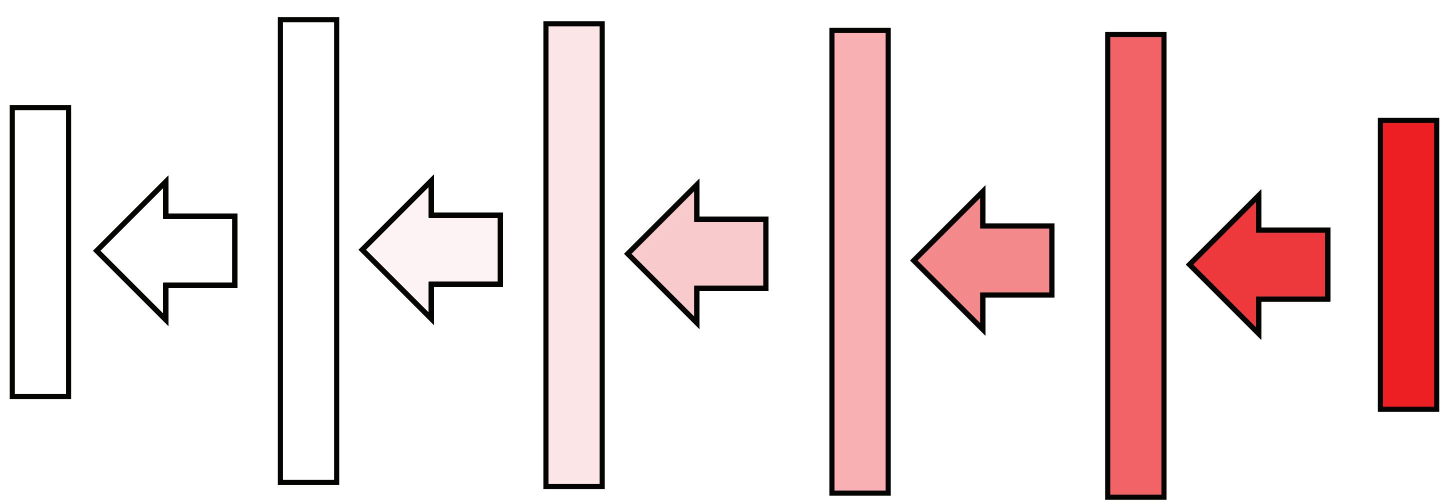
- Possibly high-dimensional, distributed, internal representation and nonlinear dynamics allow model, in theory, model complex time series
- Exact gradients can be computed exactly via Backpropagation Through Time

TRAINING RECURRENT NEURAL NETWORKS

- Possibly high-dimensional, distributed, internal representation and nonlinear dynamics allow model, in theory, model complex time series
- Exact gradients can be computed exactly via Backpropagation Through Time
- It is an interesting and powerful model. What's the catch?
  - Training RNNs via gradient descent fails on simple problems
  - Attributed to "vanishing" or "exploding" gradients
  - Much work in the 1990's focused on identifying and addressing these issues: none of these methods were widely adopted

TRAINING RECURRENT NEURAL NETWORKS

- Possibly high-dimensional, distributed, internal representation and nonlinear dynamics allow model, in theory, model complex time series
- Exact gradients can be computed exactly via Backpropagation Through Time
- It is an interesting and powerful model. What's the catch?
  - Training RNNs via gradient descent fails on simple problems
  - Attributed to "vanishing" or "exploding" gradients
  - Much work in the 1990's focused on identifying and addressing these issues: none of these methods were widely adopted



(Figure adapted from James Martens)

TRAINING RECURRENT NEURAL NETWORKS

- Possibly high-dimensional, distributed, internal representation and nonlinear dynamics allow model, in theory, model complex time series
- Exact gradients can be computed exactly via Backpropagation Through Time
- It is an interesting and powerful model. What's the catch?
  - Training RNNs via gradient descent fails on simple problems
  - Attributed to "vanishing" or "exploding" gradients
  - Much work in the 1990's focused on identifying and addressing these issues: none of these methods were widely adopted
- Best-known attempts to resolve the problem of RNN training:
  - Long Short-term Memory (LSTM) (Hochreiter and Schmidhuber 1997)
  - Echo-State Network (ESN) (Jaeger and Haas 2004)

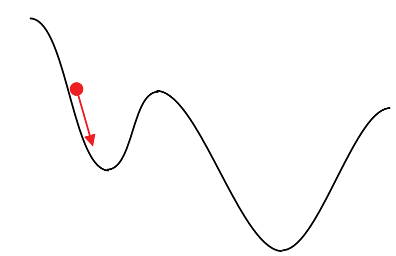
FAILURE OF GRADIENT DESCENT

Two hypotheses for why gradient descent fails for NN:

FAILURE OF GRADIENT DESCENT

Two hypotheses for why gradient descent fails for NN:

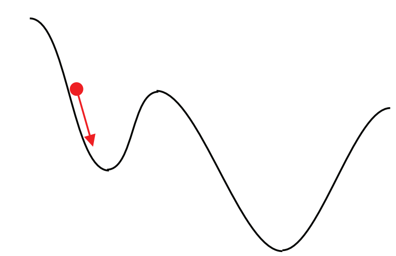
- increased frequency and severity of bad local minima



(Figures from James Martens)

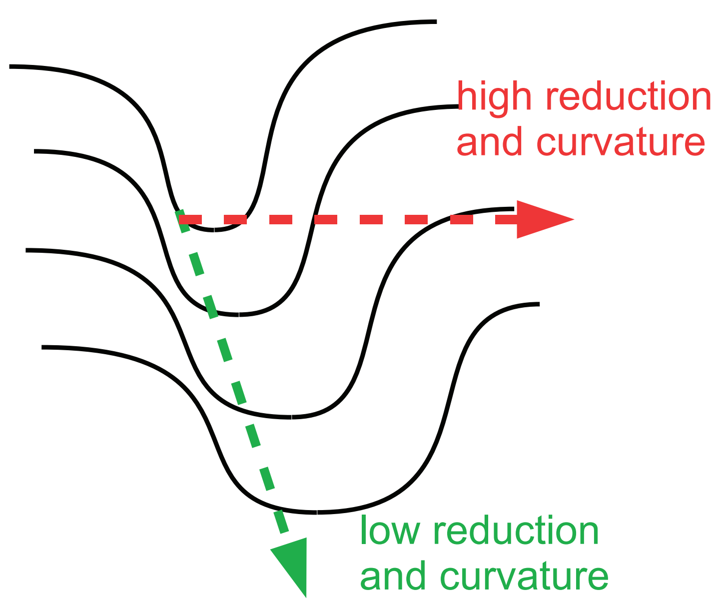
FAILURE OF GRADIENT DESCENT

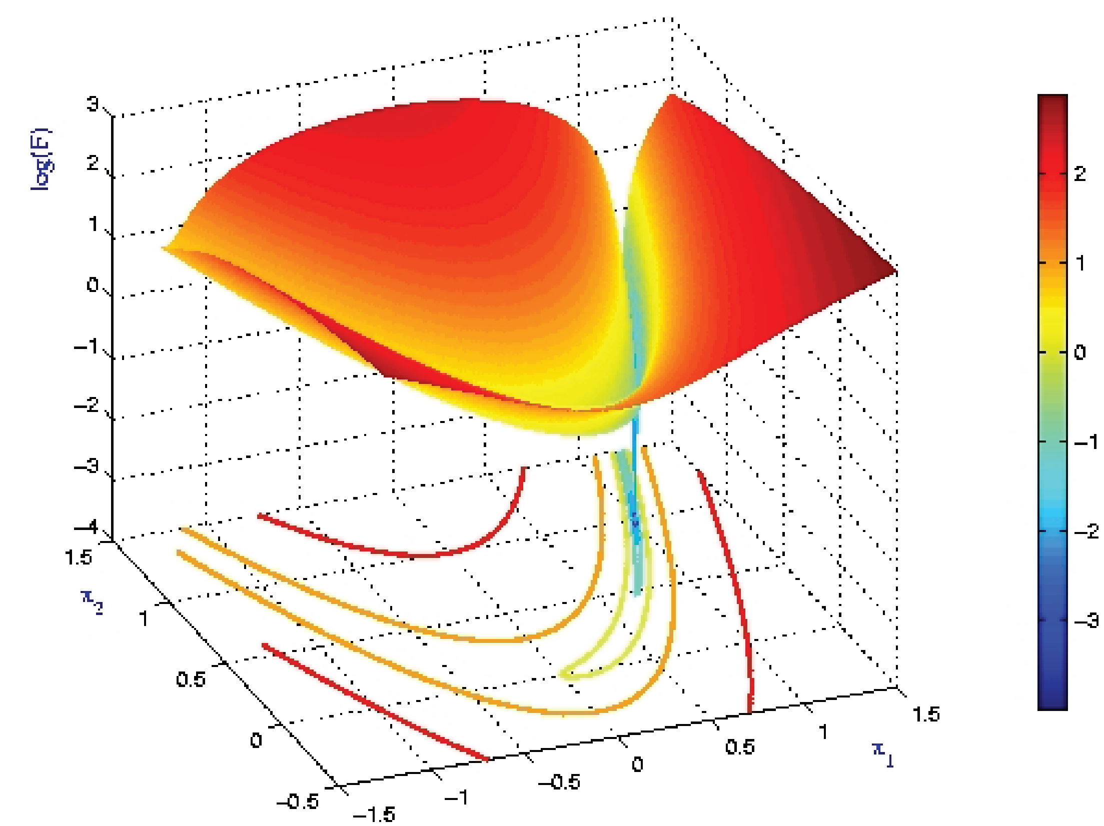
Two hypotheses for why gradient descent fails for NN:



- increased frequency and severity of bad local minima
- pathological curvature, like the type seen in the Rosenbrock function:

$$f(x,y) = (1-x)^2 + 100(y-x^2)^2$$





(Figures from James Martens)

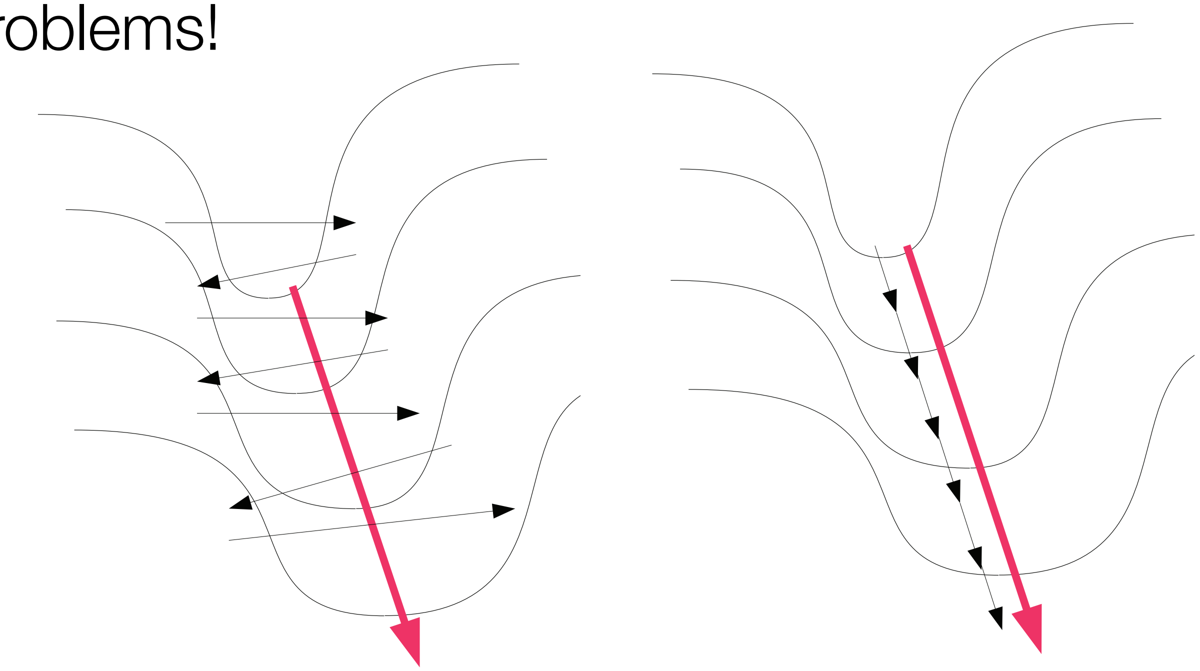
SECOND ORDER METHODS

- Model the objective function by the local approximation:

$$f(\theta + p) \approx q_\theta(p) \equiv f(\theta) + \Delta f(\theta)^T p + \frac{1}{2} p^T B p$$

where p is the search direction and B is a matrix which quantifies curvature

- In Newton's method, B is the Hessian matrix, H
- By taking the curvature information into account, Newton's method "rescales" the gradient so it is a much more sensible direction to follow
- Not feasible for high-dimensional problems!



(Figure from James Martens)

HESSIAN-FREE OPTIMIZATION

Based on exploiting two simple ideas (and some additional “tricks”):

HESSIAN-FREE OPTIMIZATION

Based on exploiting two simple ideas (and some additional "tricks"):

- For an n-dimensional vector d, the Hessian-vector product Hd can easily be computed using finite differences at the cost of a single extra gradient evaluation
  - In practice, the R-operator (Perlmutter 1994) is used instead of finite differences

HESSIAN-FREE OPTIMIZATION

Based on exploiting two simple ideas (and some additional “tricks”):

- For an n-dimensional vector d, the Hessian-vector product Hd can easily be computed using finite differences at the cost of a single extra gradient evaluation
  - In practice, the R-operator (Perlmutter 1994) is used instead of finite differences
- There is a very effective algorithm for optimizing quadratic objectives which requires only Hessian-vector products: linear conjugate-gradient (CG)

HESSIAN-FREE OPTIMIZATION

Based on exploiting two simple ideas (and some additional “tricks”):

- For an n-dimensional vector d, the Hessian-vector product Hd can easily be computed using finite differences at the cost of a single extra gradient evaluation
  - In practice, the R-operator (Perlmutter 1994) is used instead of finite differences

- There is a very effective algorithm for optimizing quadratic objectives which requires only Hessian-vector products: linear conjugate-gradient (CG)

This method was shown to effectively train RNNs in the pathological long-term dependency problems they were previously not able to solve (Martens and Sutskever 2011)

HESSIAN-FREE OPTIMIZATION

Based on exploiting two simple ideas (and some additional "tricks"):

- For an n-dimensional vector d, the Hessian-vector product Hd can easily be computed using finite differences at the cost of a single extra gradient evaluation
  - In practice, the R-operator (Perlmutter 1994) is used instead of finite differences

- There is a very effective algorithm for optimizing quadratic objectives which requires only Hessian-vector products: linear conjugate-gradient (CG)

This method was shown to effectively train RNNs in the pathological long-term dependency problems they were previously not able to solve (Martens and Sutskever 2011)

RNN demo code (using Theano): http://github.com/gwtaylor/theano-rnn

GENERATIVE MODELS WITH DISTRIBUTED STATE

GENERATIVE MODELS WITH DISTRIBUTED STATE

- Many sequences are high-dimensional and have complex structure
  - music, human motion, weather/climate data
  - RNNs simply predict the expected value at the next time step
  - They can't capture multi-modality

GENERATIVE MODELS WITH DISTRIBUTED STATE

- Many sequences are high-dimensional and have complex structure
  - music, human motion, weather/climate data
  - RNNs simply predict the expected value at the next time step
  - They can't capture multi-modality
- Generative models (like Restricted Boltzmann Machines) can capture complex distributions

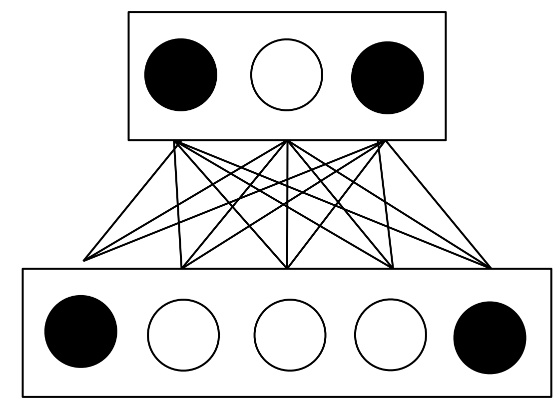
GENERATIVE MODELS WITH DISTRIBUTED STATE

- Many sequences are high-dimensional and have complex structure
  - music, human motion, weather/climate data
  - RNNs simply predict the expected value at the next time step
  - They can’t capture multi-modality
- Generative models (like Restricted Boltzmann Machines) can capture complex distributions
- Use binary hidden state and gain the best of HMM & LDS:
  - the nonlinear dynamics and observation model of the HMM without the limited hidden state
  - the efficient, expressive state of the LDS without the linear-Gaussian restriction on dynamics and observations

DISTRIBUTED BINARY HIDDEN STATE

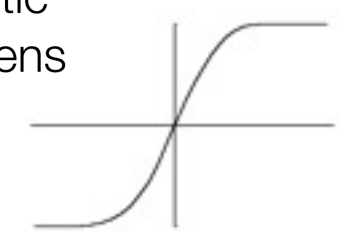
- Using distributed binary representations for hidden state in directed models of time series makes inference difficult. But we can:
  - Use a Restricted Boltzmann Machine (RBM) for the interactions between hidden and visible variables. A factorial posterior makes inference and sampling easy.
  - Treat the visible variables in the previous time slice as additional **fixed** inputs

Hidden variables (factors) at time t



Visible variables (observations) at time t

One typically uses binary logistic units for both visibles and hiddens



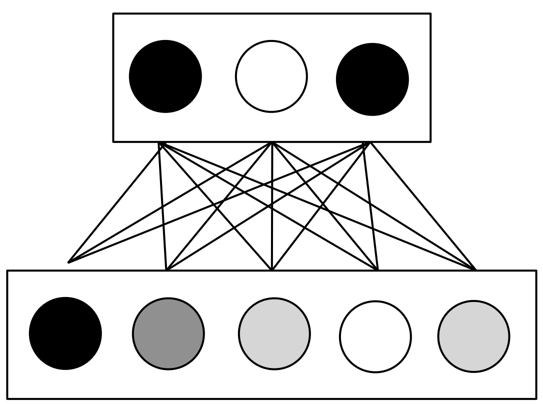
$$p(h_j = 1|\mathbf{v}) = \sigma(b_j + \sum_i v_i W_{ij})$$

$$p(v_i = 1|\mathbf{h}) = \sigma(b_i + \sum_j h_j W_{ij})$$

MODELING OBSERVATIONS WITH AN RBM

- So the distributed binary latent (hidden) state of an RBM lets us:
  - Model complex, nonlinear dynamics
  - Easily and exactly infer the latent binary state given the observations
- But RBMs treat data as static (i.i.d.)

Hidden variables (factors) at time t

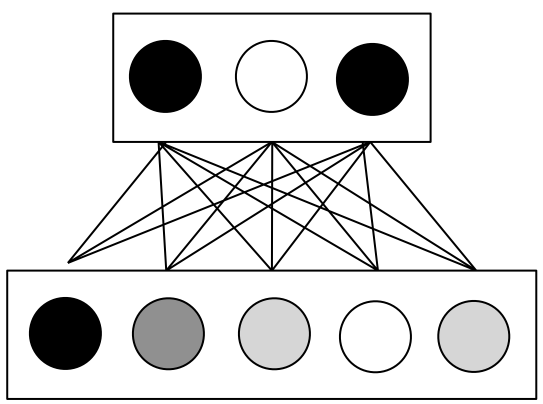


Visible variables (joint angles) at time t

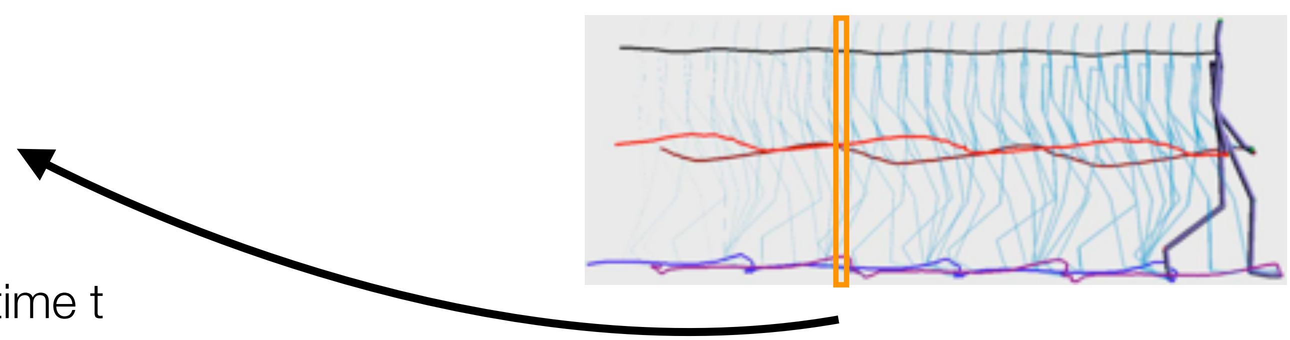
MODELING OBSERVATIONS WITH AN RBM

- So the distributed binary latent (hidden) state of an RBM lets us:
  - Model complex, nonlinear dynamics
  - Easily and exactly infer the latent binary state given the observations
- But RBMs treat data as static (i.i.d.)

Hidden variables (factors) at time t

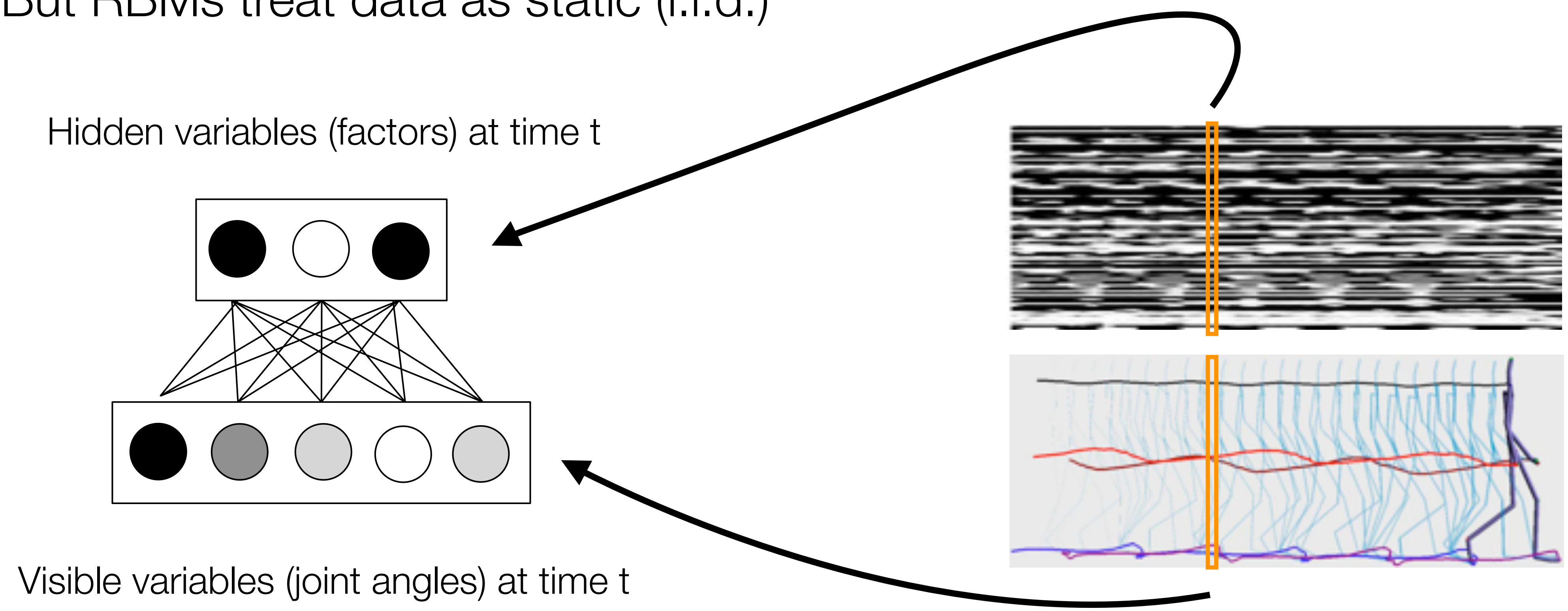


Visible variables (joint angles) at time t



MODELING OBSERVATIONS WITH AN RBM

- So the distributed binary latent (hidden) state of an RBM lets us:
  - Model complex, nonlinear dynamics
  - Easily and exactly infer the latent binary state given the observations
- But RBMs treat data as static (i.i.d.)



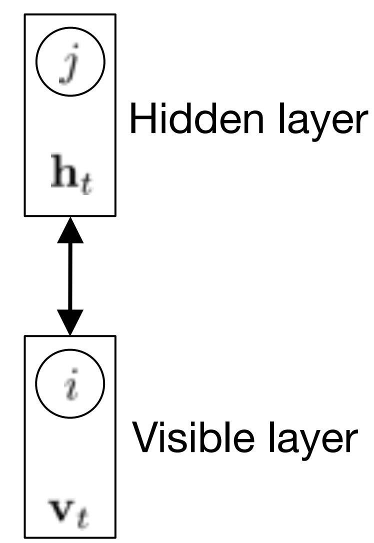
CONDITIONAL RESTRICTED BOLTZMANN MACHINES

(Taylor, Hinton and Roweis NIPS 2006, JMLR 2011)

CONDITIONAL RESTRICTED BOLTZMANN MACHINES

(Taylor, Hinton and Roweis NIPS 2006, JMLR 2011)

- Start with a Restricted Boltzmann Machine (RBM)

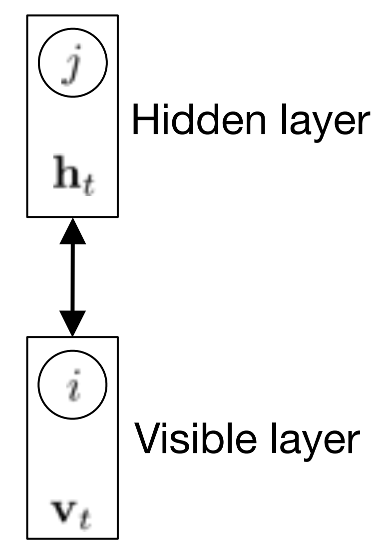


CONDITIONAL RESTRICTED BOLTZMANN MACHINES

(Taylor, Hinton and Roweis NIPS 2006, JMLR 2011)

- Start with a Restricted Boltzmann Machine (RBM)

- Add two types of directed connections:



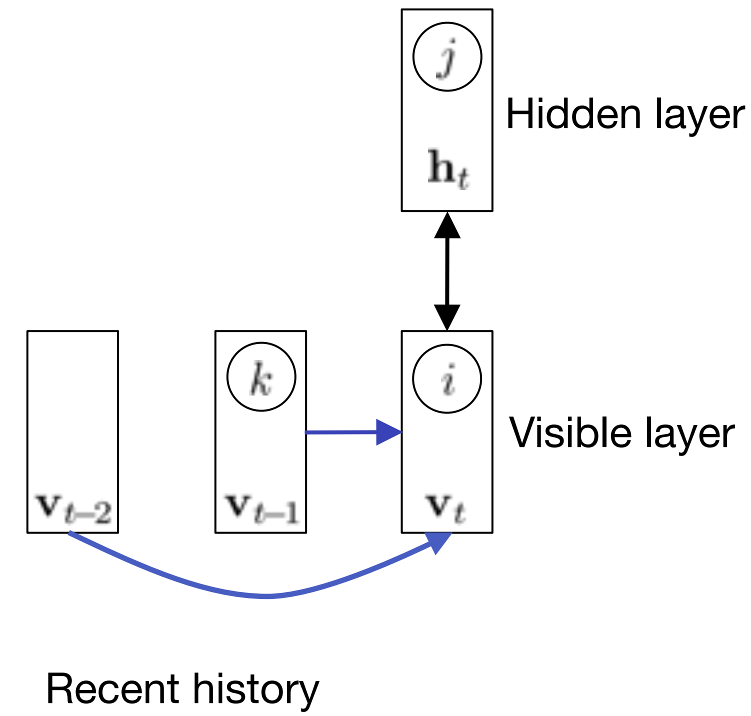
CONDITIONAL RESTRICTED BOLTZMANN MACHINES

(Taylor, Hinton and Roweis NIPS 2006, JMLR 2011)

- Start with a Restricted Boltzmann Machine (RBM)

- Add two types of directed connections:

  - Autoregressive connections model short-term, linear structure

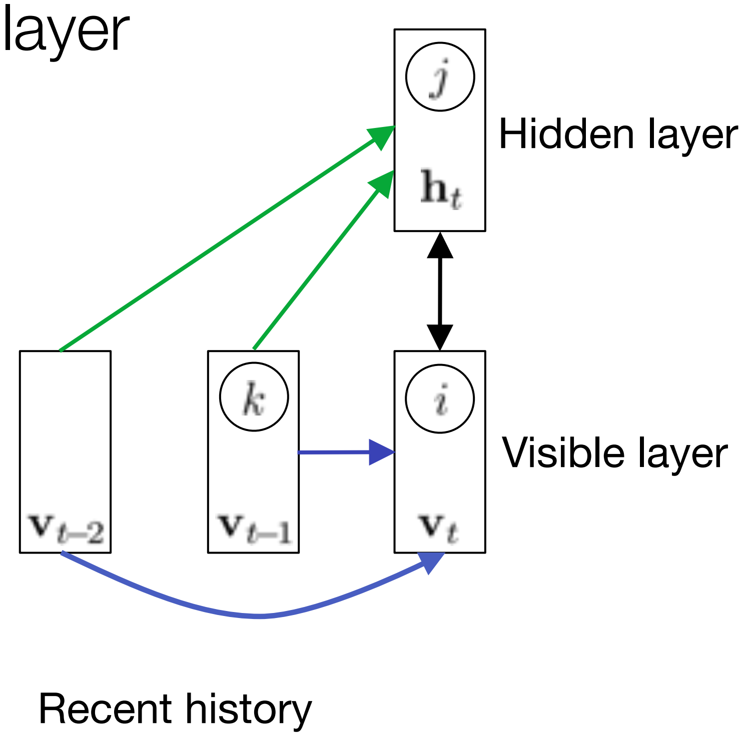


CONDITIONAL RESTRICTED BOLTZMANN MACHINES

(Taylor, Hinton and Roweis NIPS 2006, JMLR 2011)

- Start with a Restricted Boltzmann Machine (RBM)

- Add two types of directed connections:
  - Autoregressive connections model short-term, linear structure
  - History can also influence dynamics through hidden layer



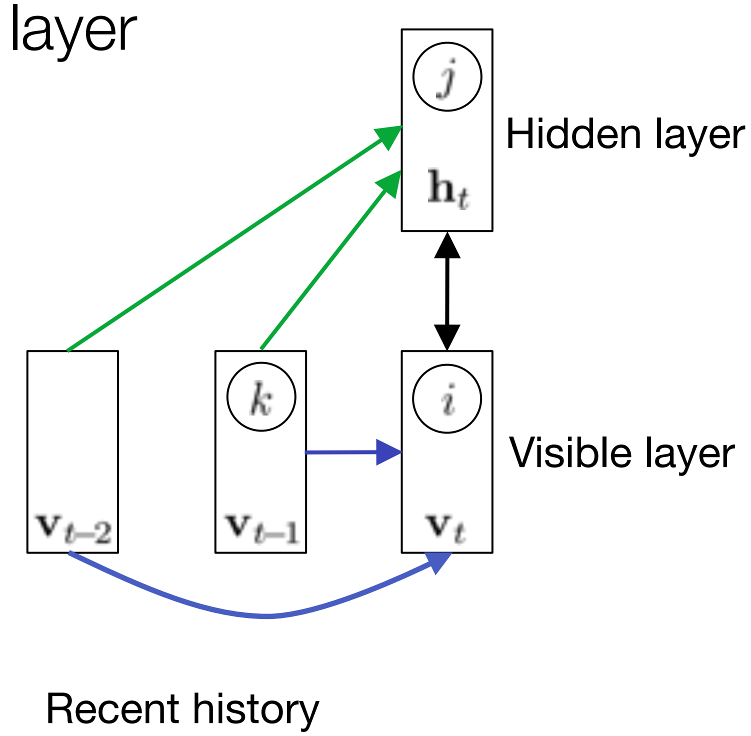
CONDITIONAL RESTRICTED BOLTZMANN MACHINES

(Taylor, Hinton and Roweis NIPS 2006, JMLR 2011)

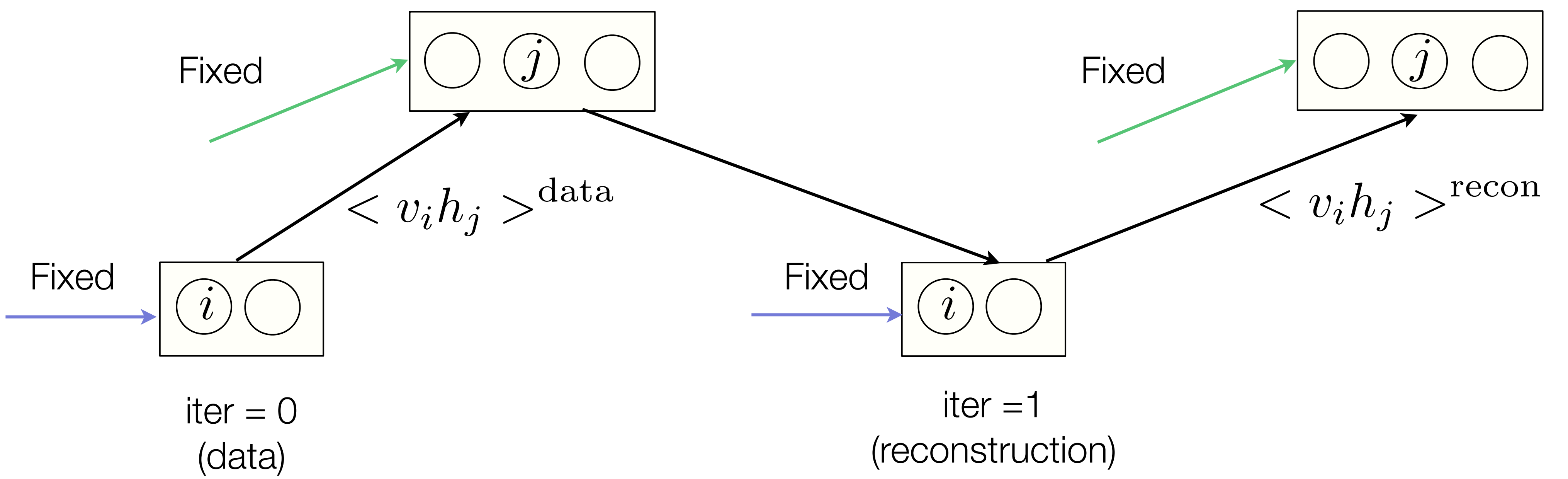
- Start with a Restricted Boltzmann Machine (RBM)

- Add two types of directed connections:
  - Autoregressive connections model short-term, linear structure
  - History can also influence dynamics through hidden layer

- Conditioning does not change inference nor learning



CONTRASTIVE DIVERGENCE LEARNING IN A CRBM

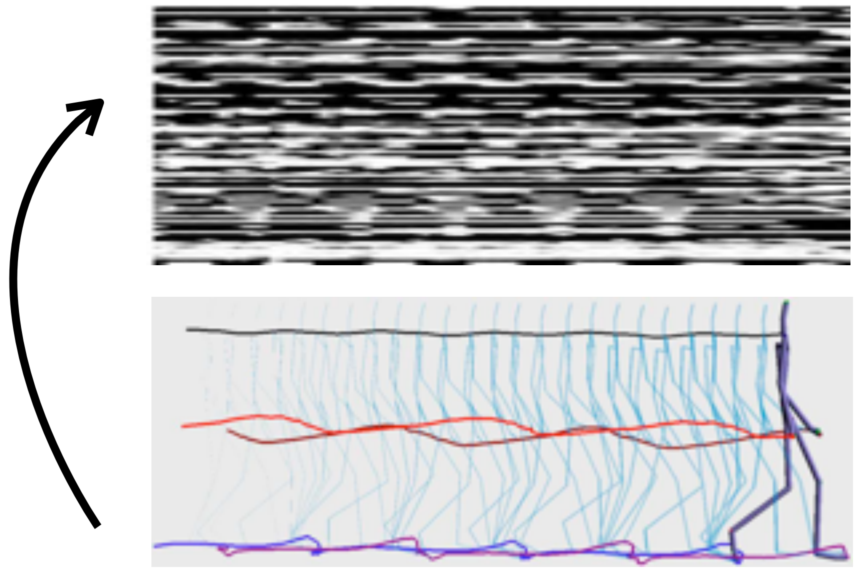


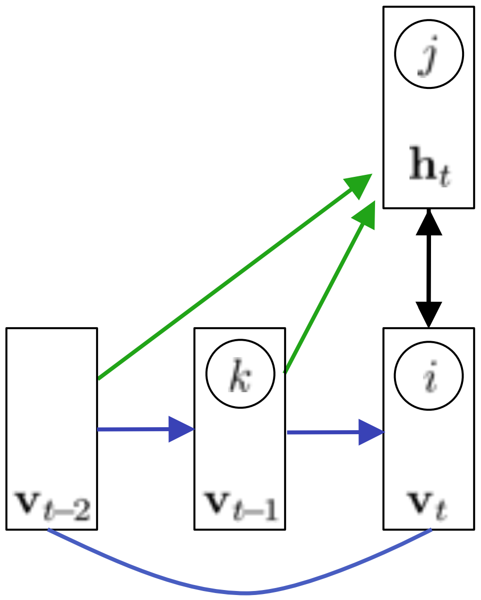
- When updating visible and hidden units, we implement directed connections by treating data from previous time steps as a dynamically changing bias
- Inference and learning do not change

STACKING: THE CONDITIONAL DEEP BELIEF NETWORK

STACKING: THE CONDITIONAL DEEP BELIEF NETWORK

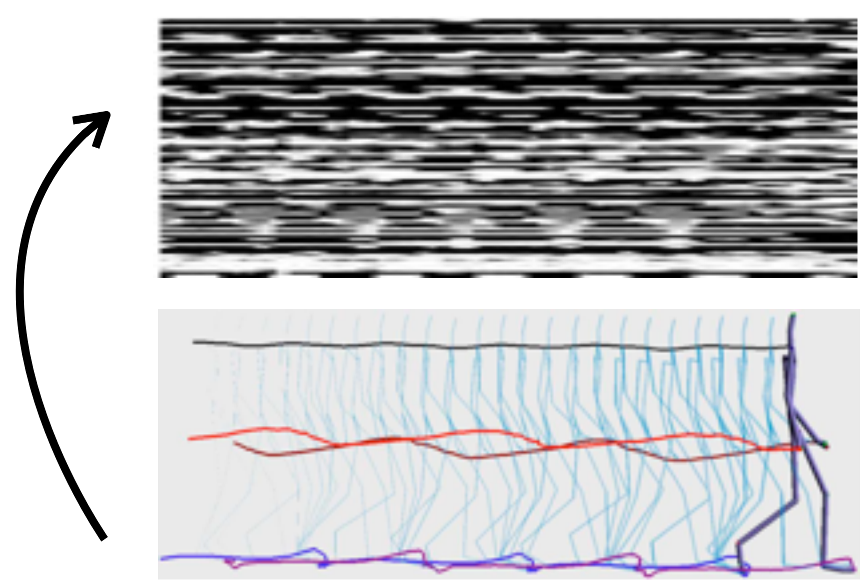
- Learn a CRBM

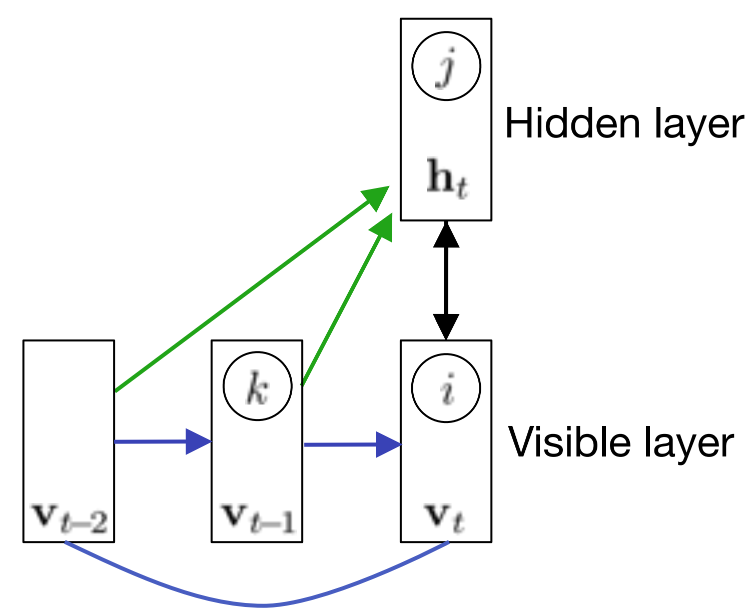




STACKING: THE CONDITIONAL DEEP BELIEF NETWORK

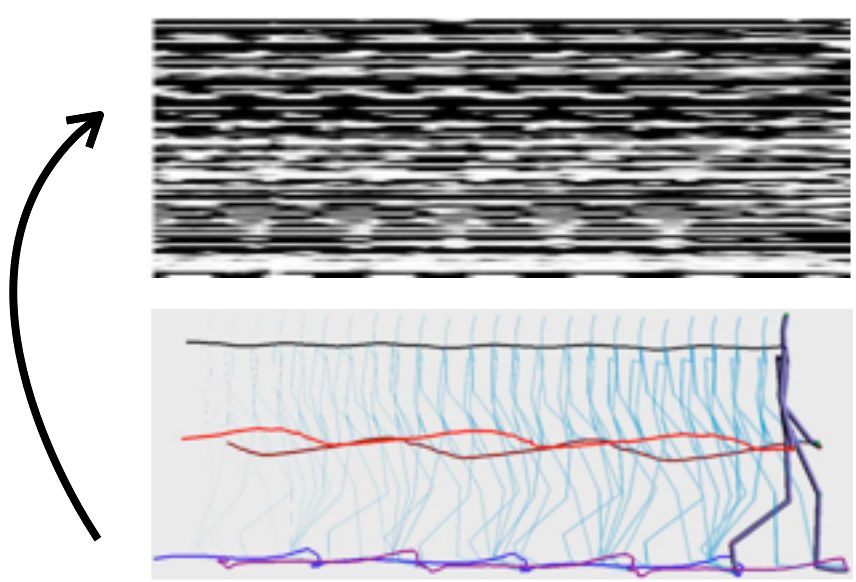
- Learn a CRBM
- Now, treat the sequence of hidden units as “fully observed” data and train a second CRBM

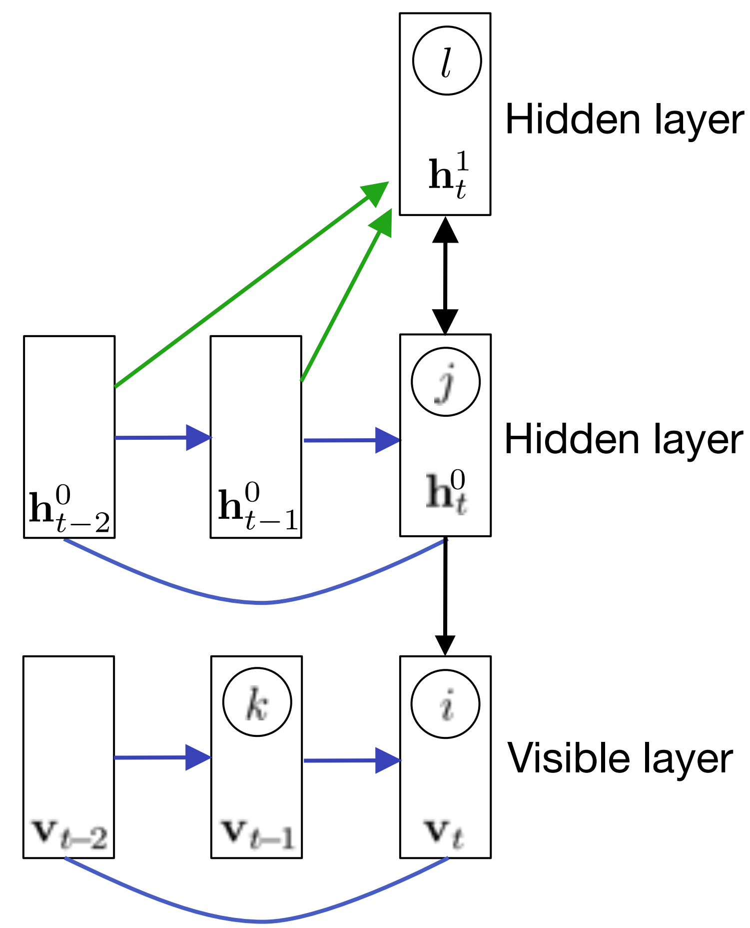




STACKING: THE CONDITIONAL DEEP BELIEF NETWORK

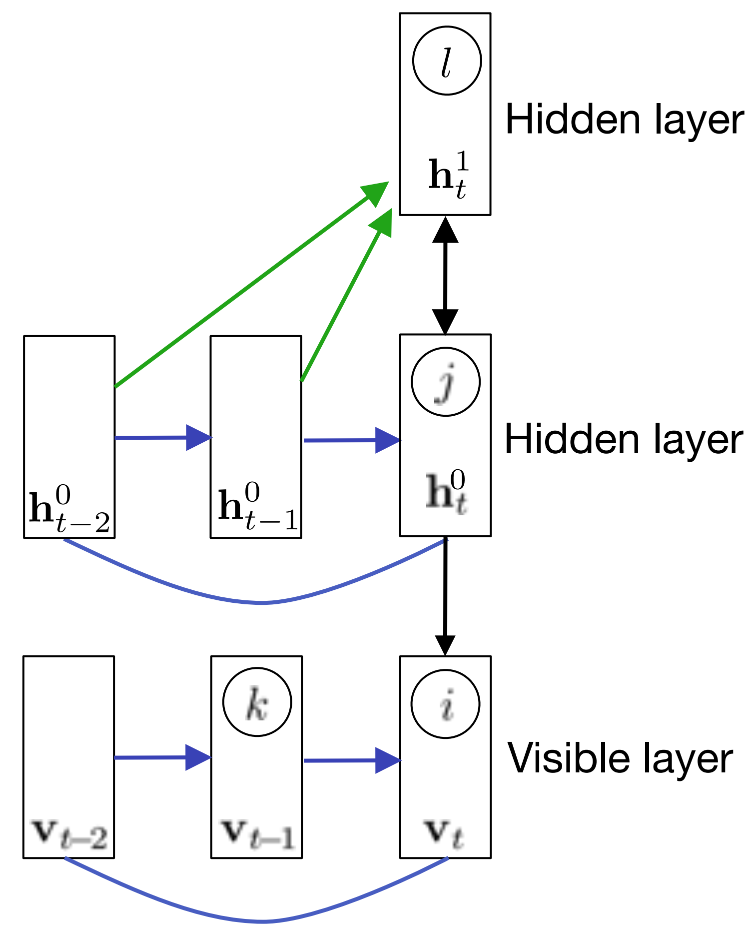
- Learn a CRBM
- Now, treat the sequence of hidden units as “fully observed” data and train a second CRBM
- The composition of CRBMs is a conditional deep belief net

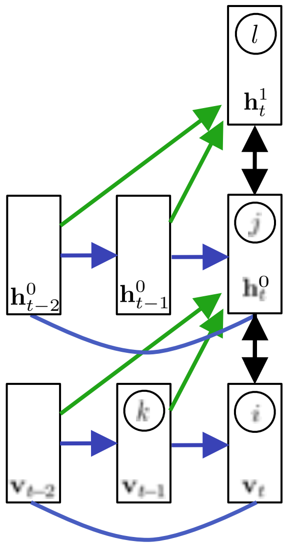




STACKING: THE CONDITIONAL DEEP BELIEF NETWORK

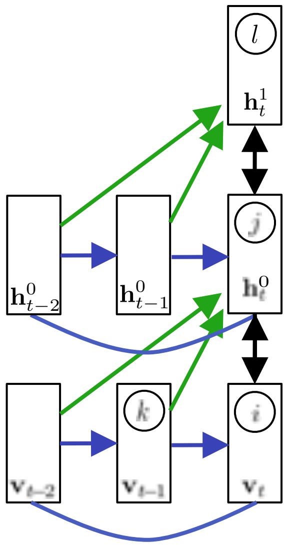
- Learn a CRBM
- Now, treat the sequence of hidden units as “fully observed” data and train a second CRBM
- The composition of CRBMs is a conditional deep belief net
- It can be fine-tuned generatively or discriminatively





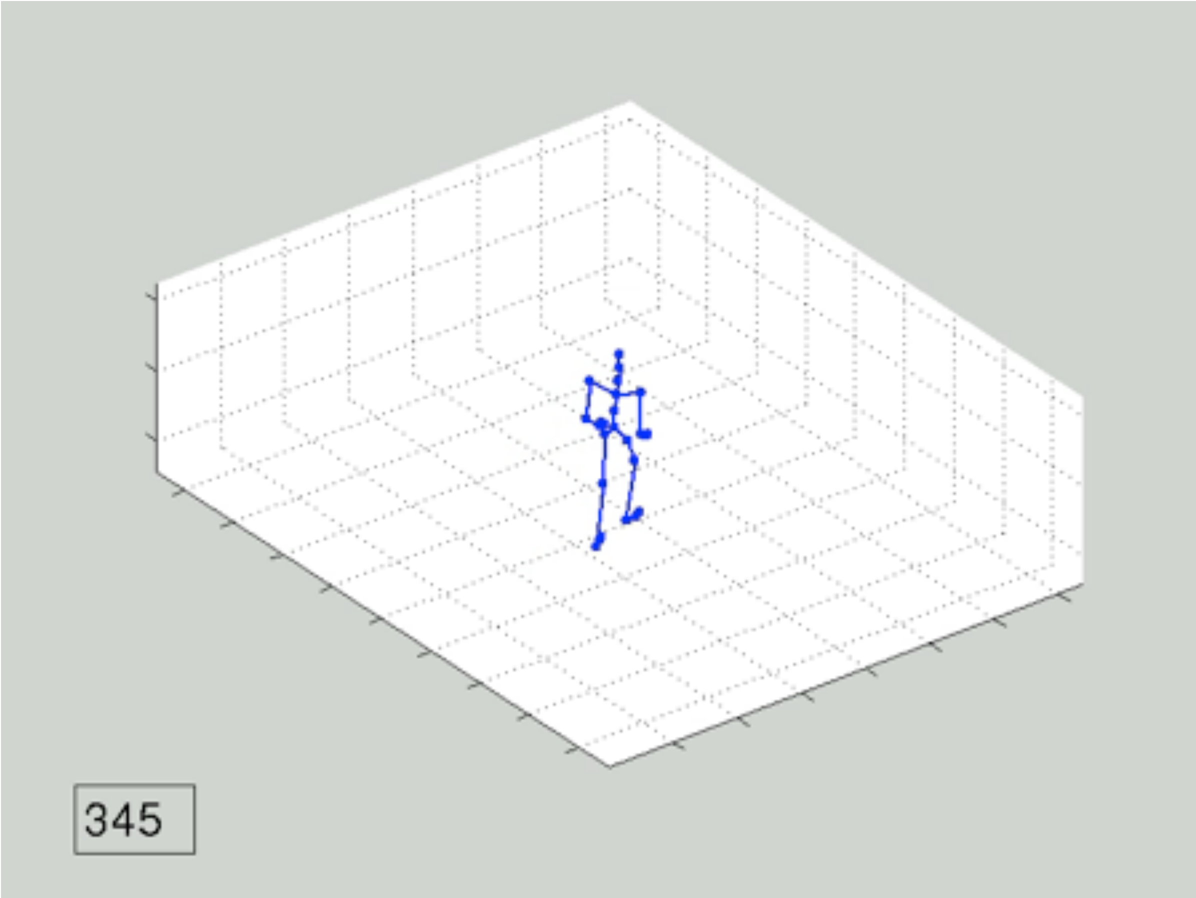
MOTION SYNTHESIS WITH A 2-LAYER CDBN

- Model is trained on ~8000 frames of 60fps data (49 dimensions)
- 10 styles of walking: cat, chicken, dinosaur, drunk, gangly, graceful, normal, old-man, sexy and strong
- 600 binary hidden units per layer
- < 1 hour training on a modern workstation

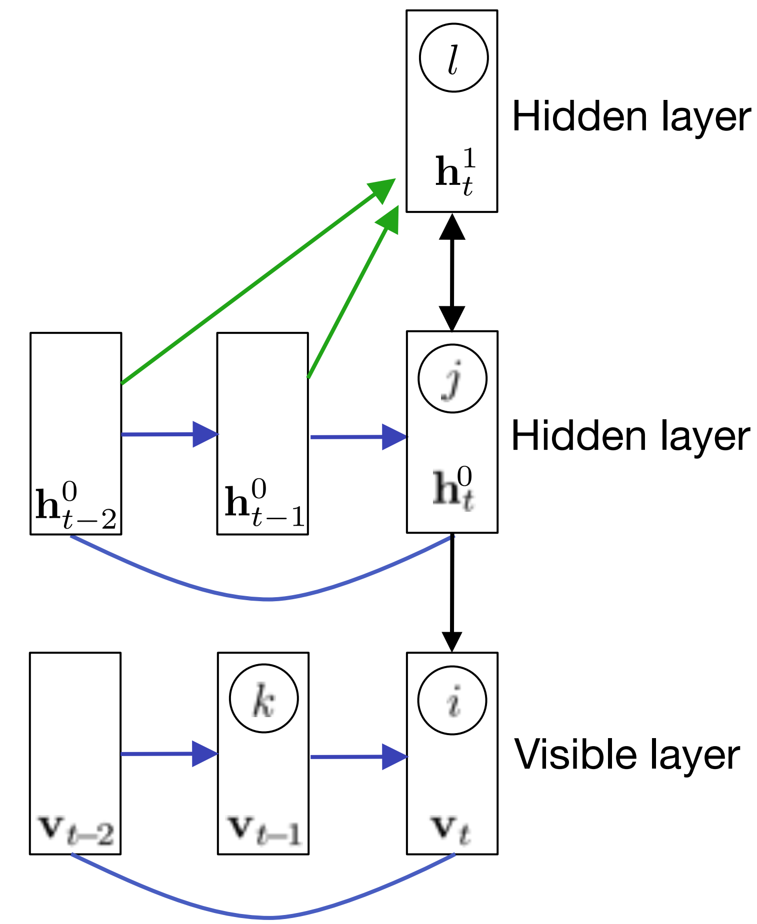


MOTION SYNTHESIS WITH A 2-LAYER CDBN

- Model is trained on ~8000 frames of 60fps data (49 dimensions)
- 10 styles of walking: cat, chicken, dinosaur, drunk, gangly, graceful, normal, old-man, sexy and strong
- 600 binary hidden units per layer
- < 1 hour training on a modern workstation

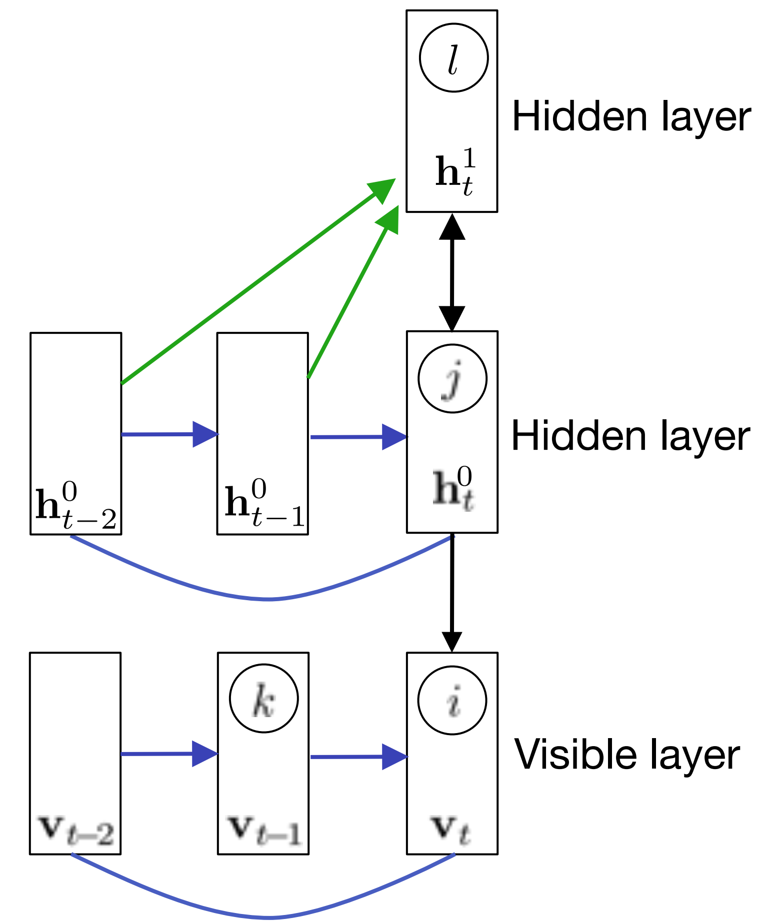


MODELING CONTEXT



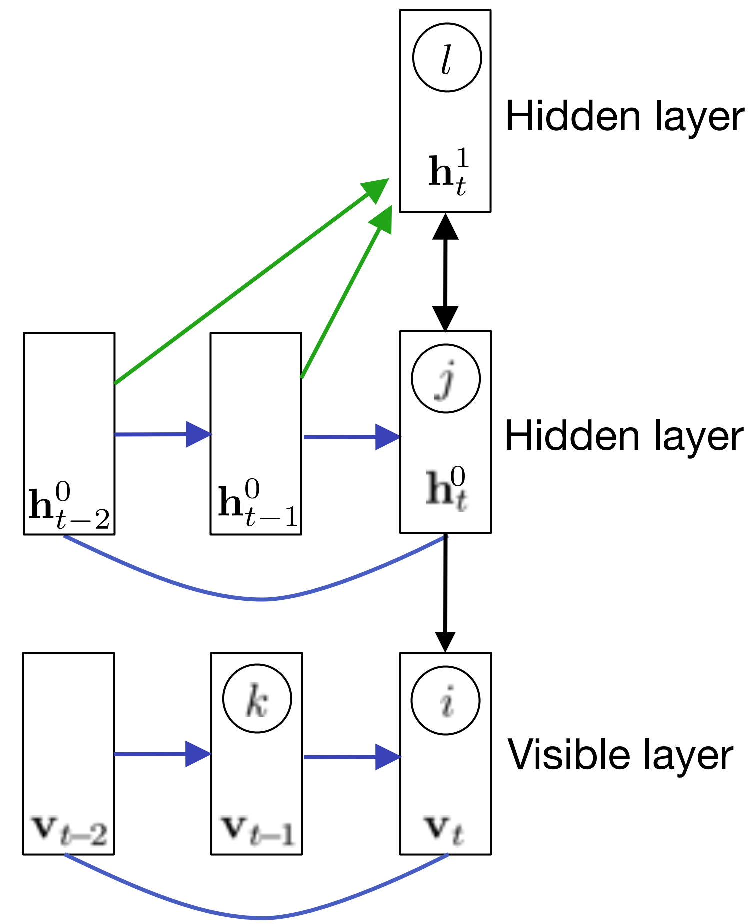
MODELING CONTEXT

- A single model was trained on 10 "styled" walks from CMU subject 137



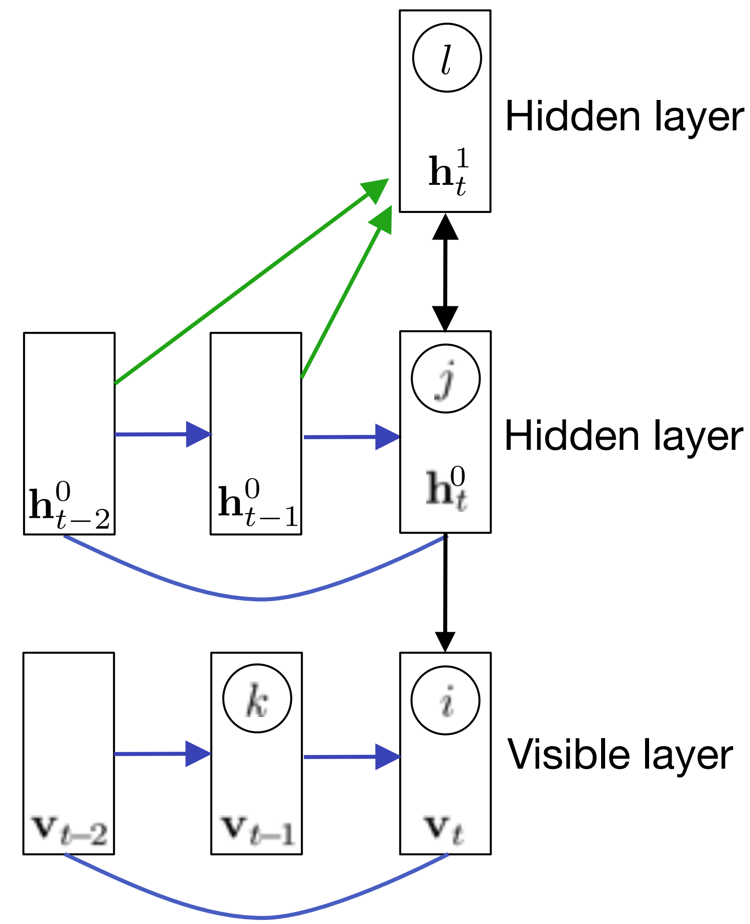
MODELING CONTEXT

- A single model was trained on 10 "styled" walks from CMU subject 137
- The model can generate each style based on initialization



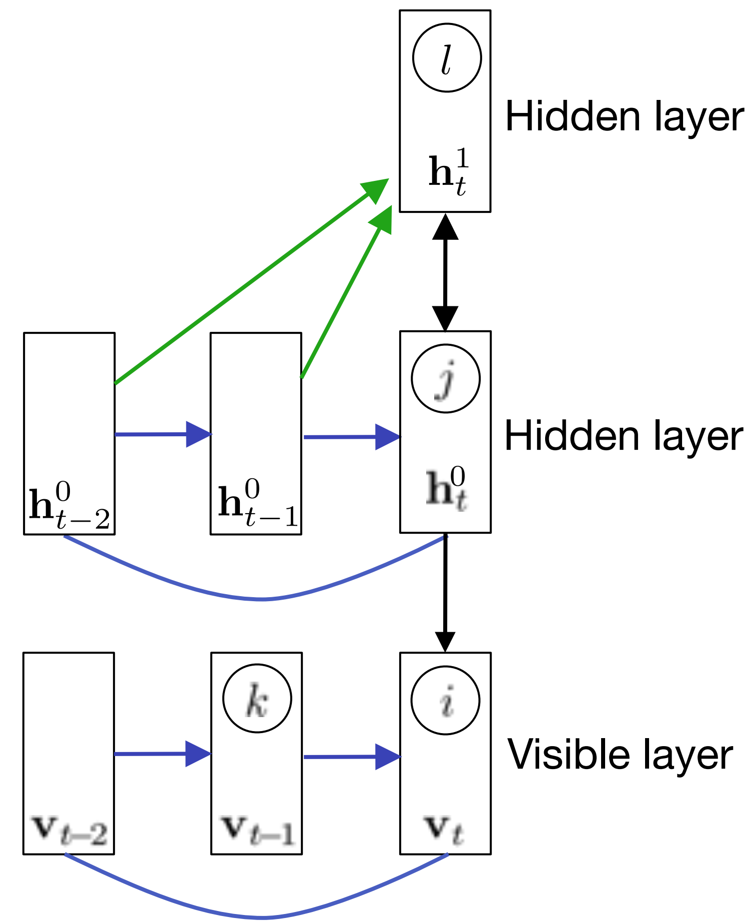
MODELING CONTEXT

- A single model was trained on 10 “styled” walks from CMU subject 137
- The model can generate each style based on initialization
- We cannot prevent nor control transitioning



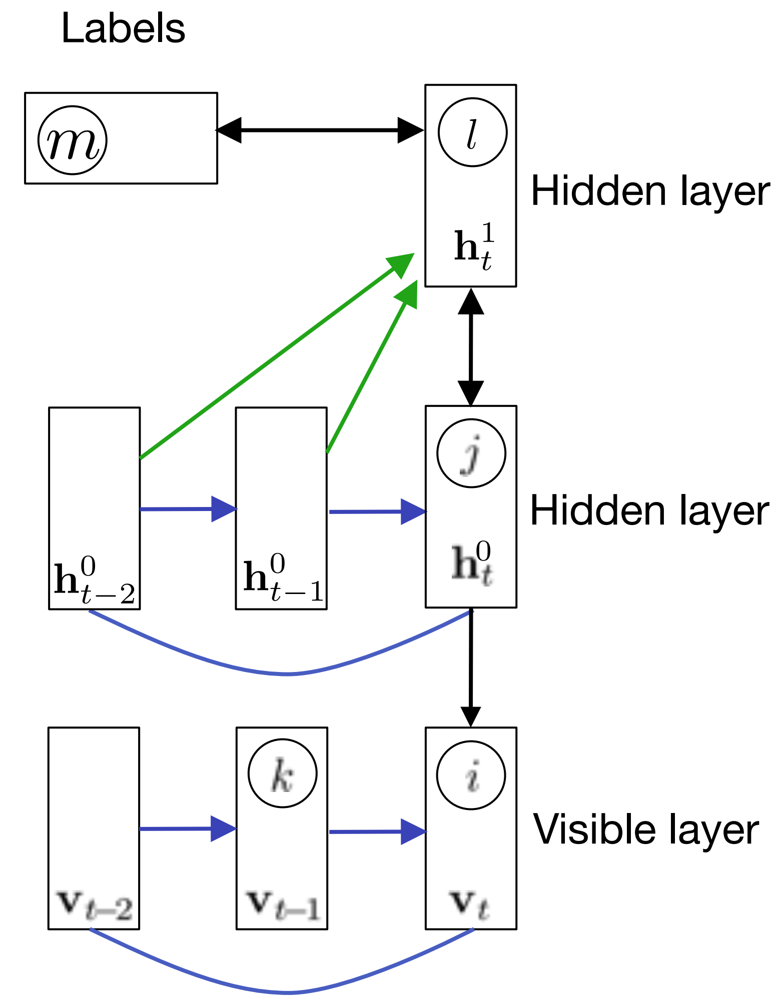
MODELING CONTEXT

- A single model was trained on 10 “styled” walks from CMU subject 137
- The model can generate each style based on initialization
- We cannot prevent nor control transitioning
- How to blend styles?

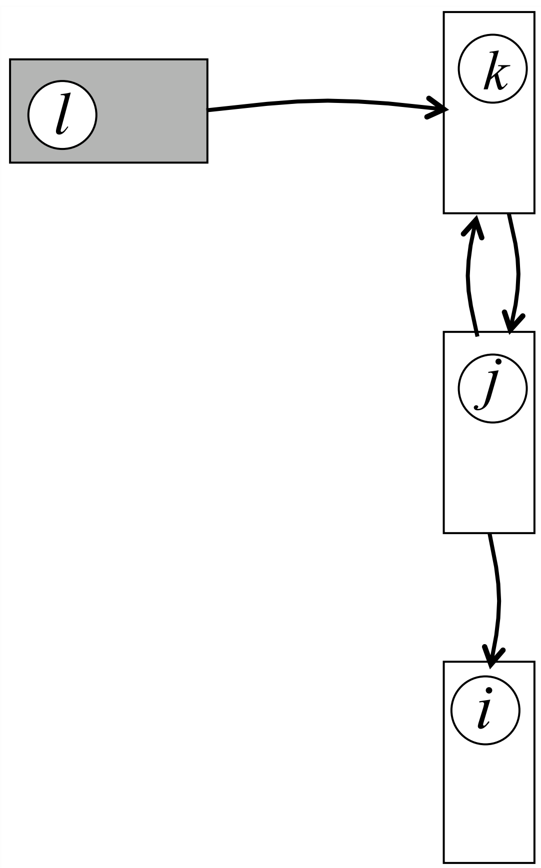


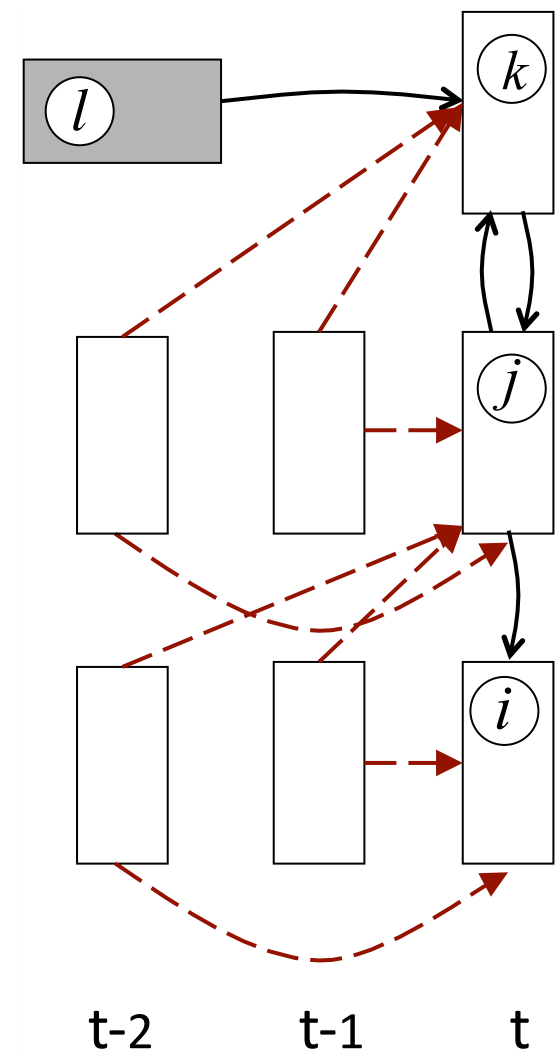
MODELING CONTEXT

- A single model was trained on 10 “styled” walks from CMU subject 137
- The model can generate each style based on initialization
- We cannot prevent nor control transitioning
- How to blend styles?
- Style or person labels can be provided as part of the input to the top layer



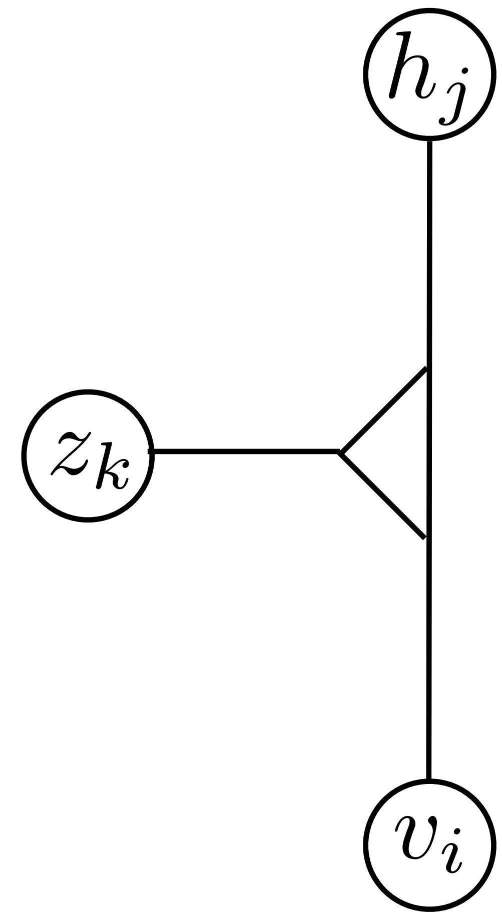
HOW TO MAKE CONTEXT INFLUENCE DYNAMICS?





MULTIPLICATIVE INTERACTIONS

- Let latent variables act like *gates*, that dynamically change the connections between other variables
- This amounts to letting variables multiply connections between other variables: *three-way multiplicative interactions*
- Recently used in the context of learning *correspondence* between images (Memisevic & Hinton 2007, 2010) but long history before that

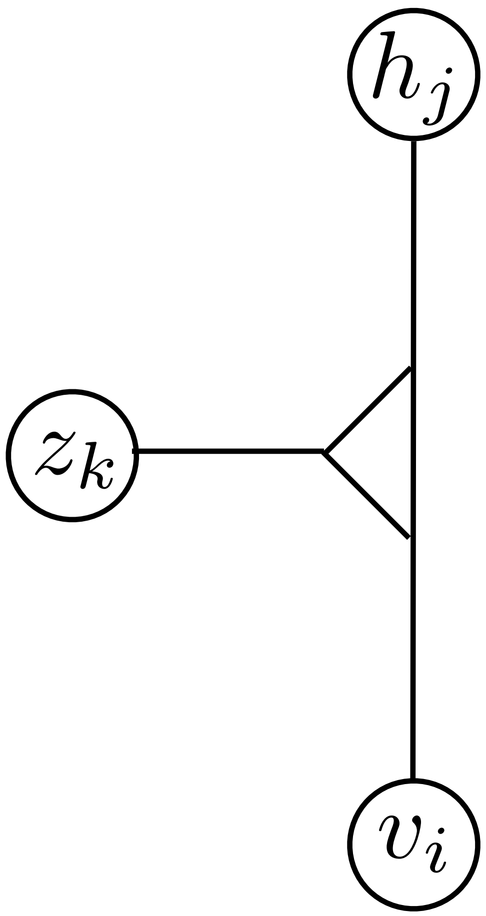


MULTIPLICATIVE INTERACTIONS

- Let latent variables act like *gates*, that dynamically change the connections between other variables

- This amounts to letting variables multiply connections between other variables: *three-way multiplicative interactions*

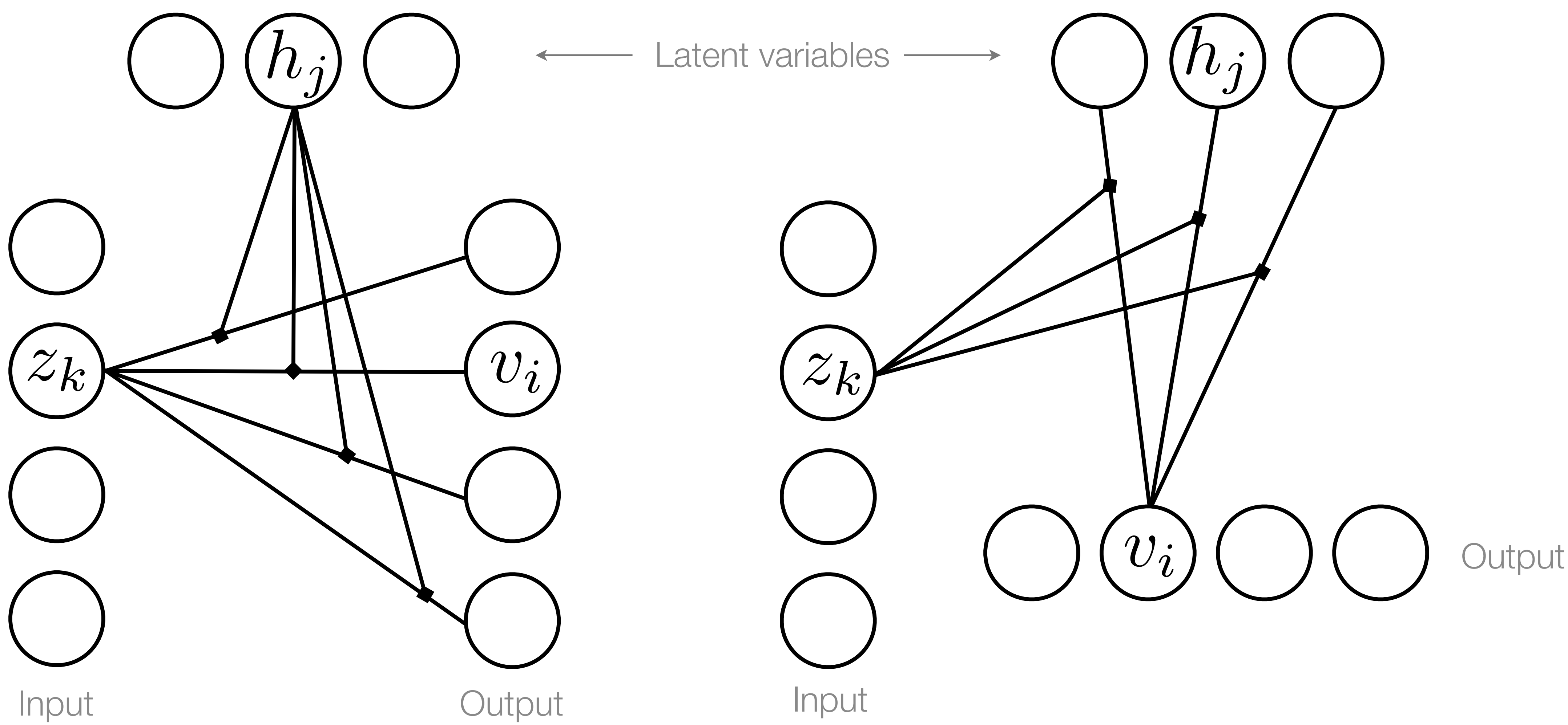
- Recently used in the context of learning *correspondence* between images (Memisevic & Hinton 2007, 2010) but long history before that



Roland Memisevic has a nice Tutorial and review paper on the subject:
http://www.cs.toronto.edu/~rfm/multiview-feature-learning-cvpr/

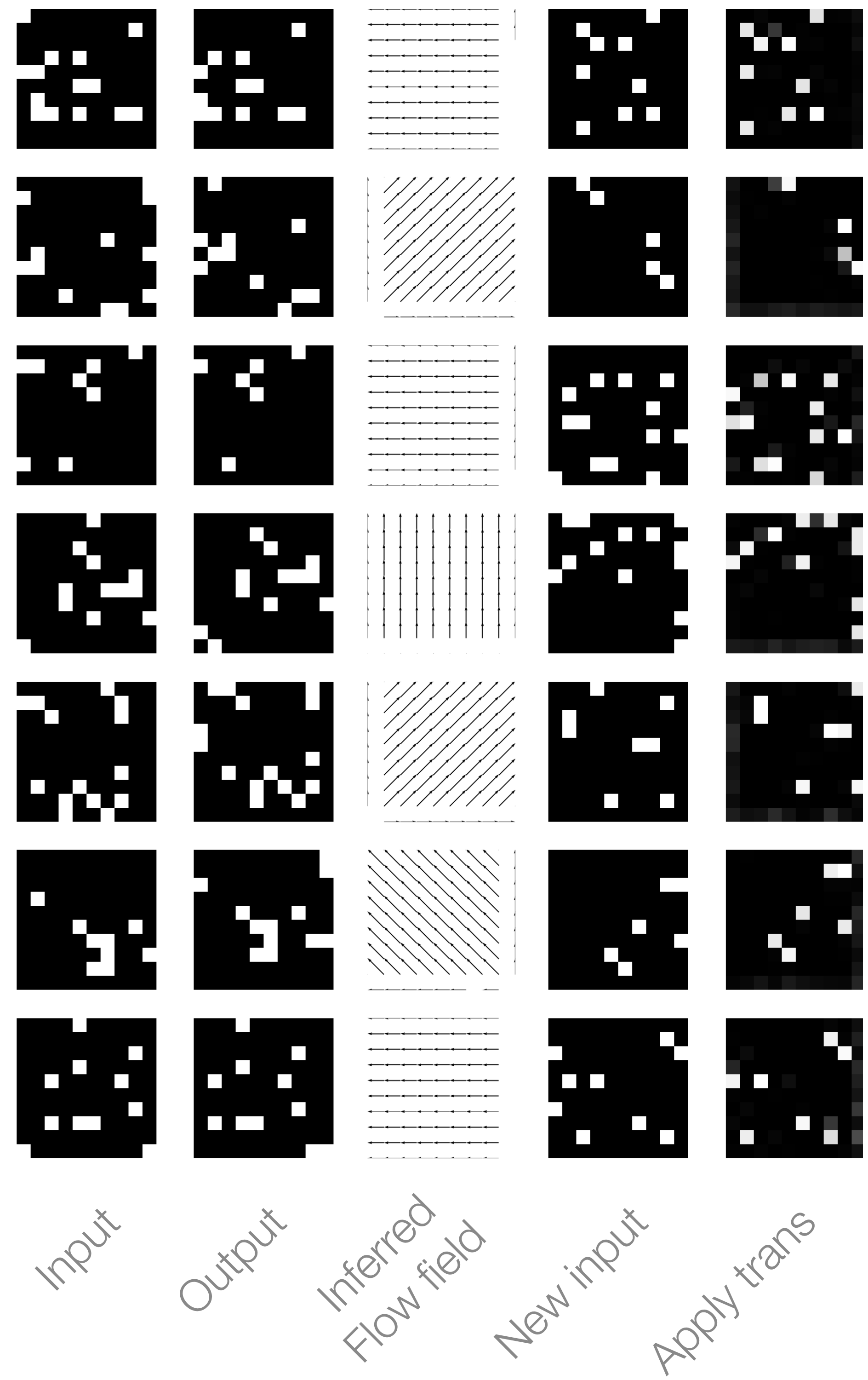
GATED RESTRICTED BOLTZMANN MACHINES (GRBM)

Two views: Memisevic & Hinton (2007)



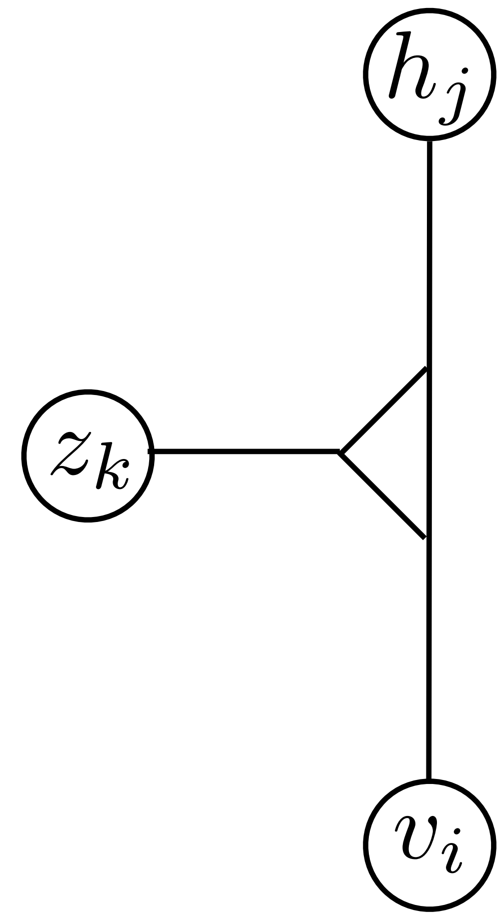
INFERRING OPTICAL FLOW: IMAGE "ANALOGIES"

- Toy images (Memisevic & Hinton 2006)
- No structure in these images, only *how they change*
- Can infer optical flow from a pair of images and apply it to a random image



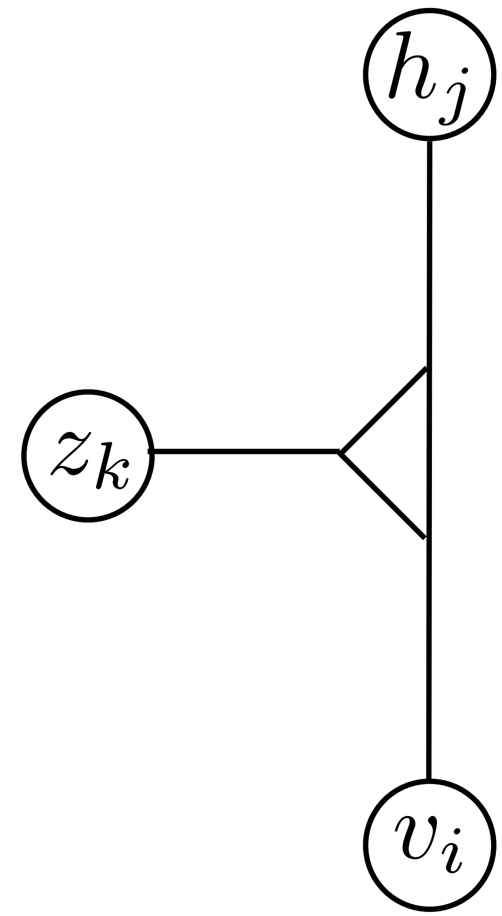
BACK TO MOTION STYLE

- Introduce a set of latent "context" variables whose value is known at training time
- In our example, these represent "motion style" but could also represent height, weight, gender, etc.
- The contextual variables gate every existing pairwise connection in our model



LEARNING AND INFERENCE

- Learning and inference remain almost the same as in the standard CRBM
- We can think of the context or style variables as "blending in" a whole "sub-network"
- This allows us to share parameters across styles but selectively adapt dynamics

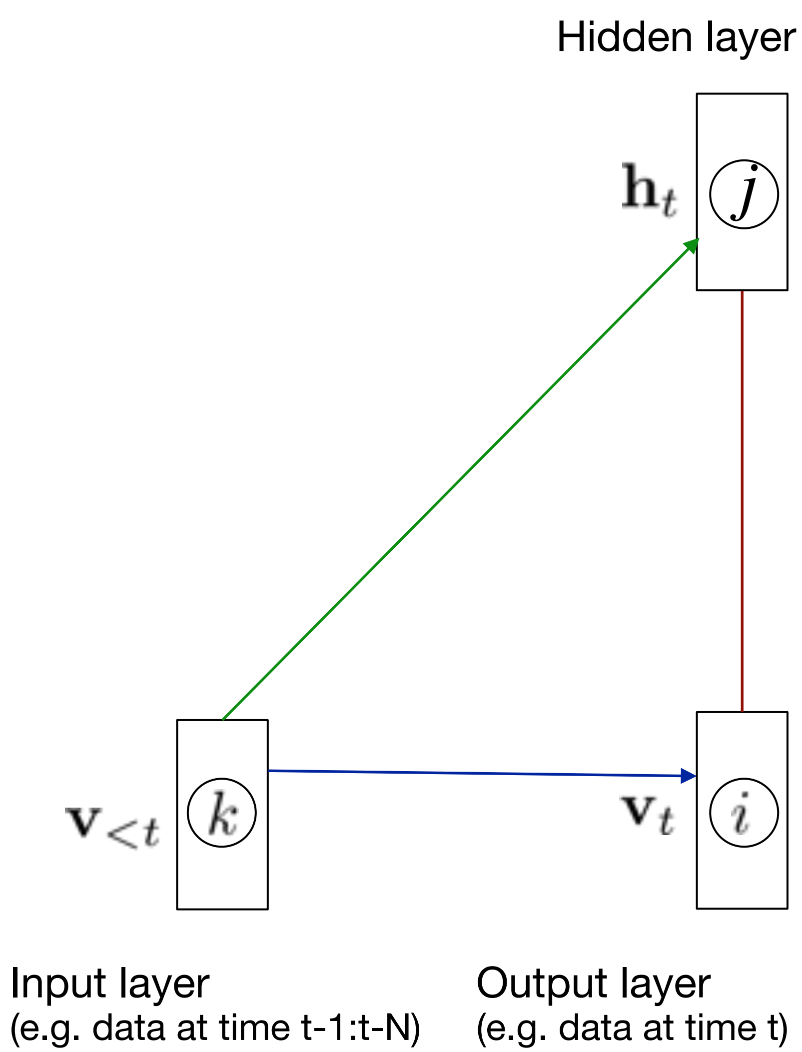


SUPERVISED MODELING OF STYLE

(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)

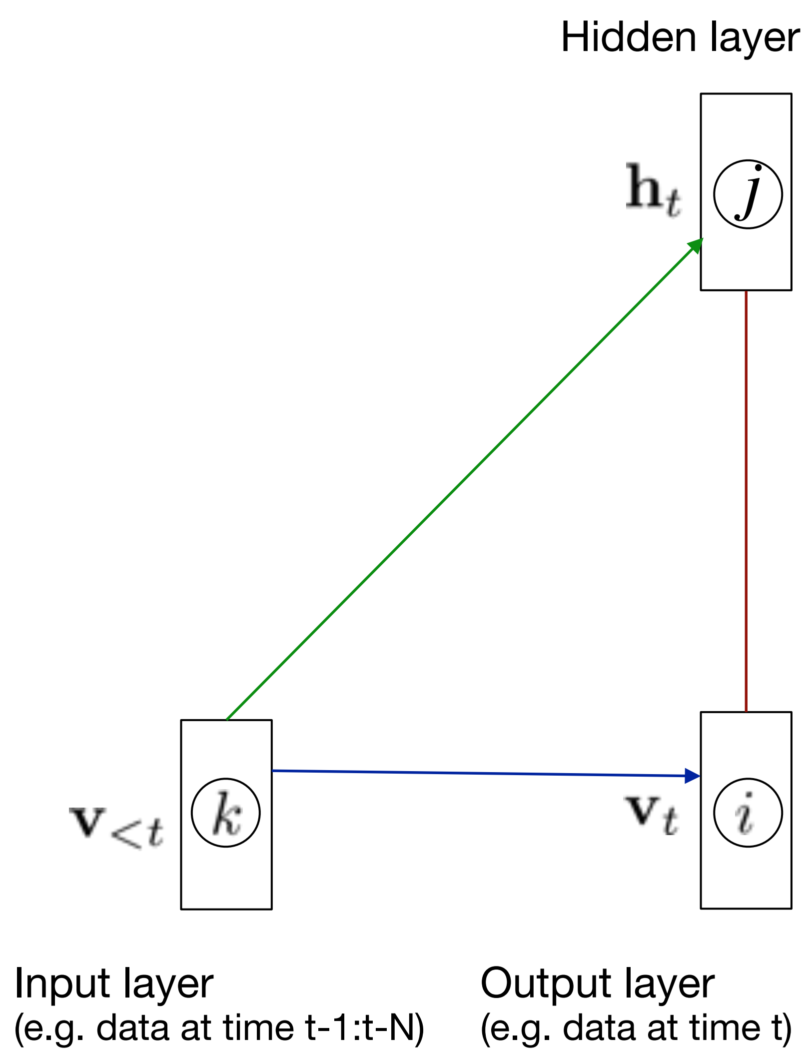
SUPERVISED MODELING OF STYLE

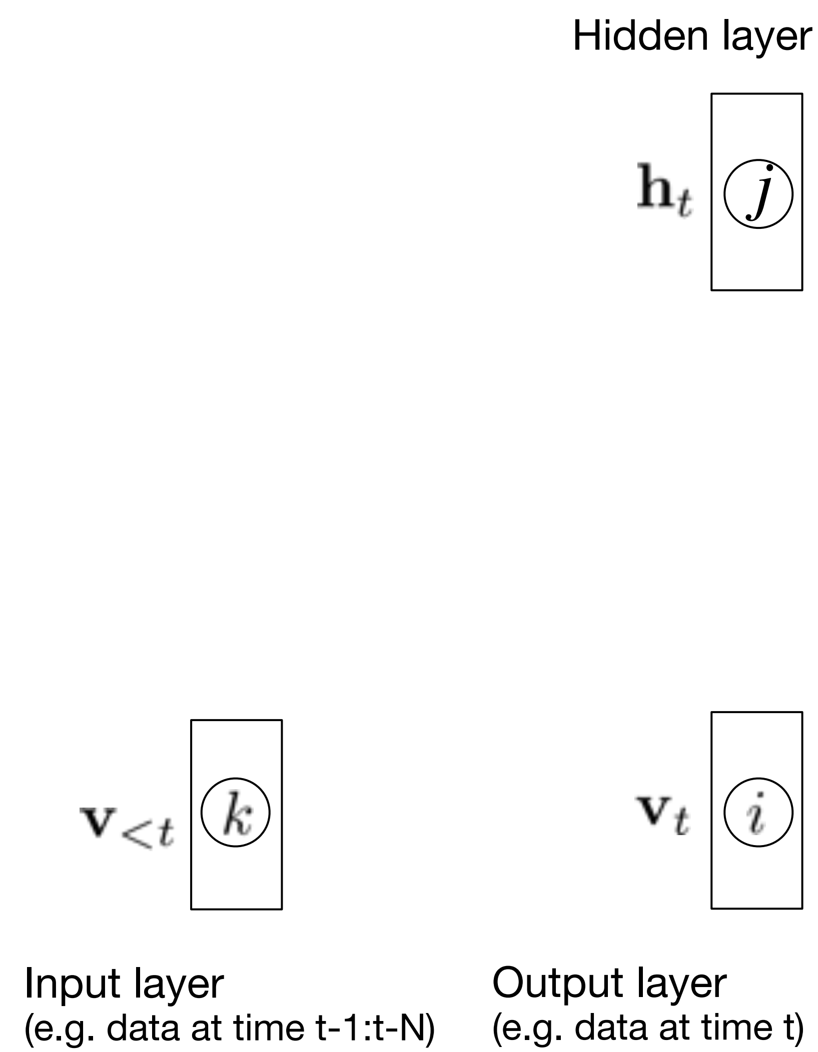
(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)



SUPERVISED MODELING OF STYLE

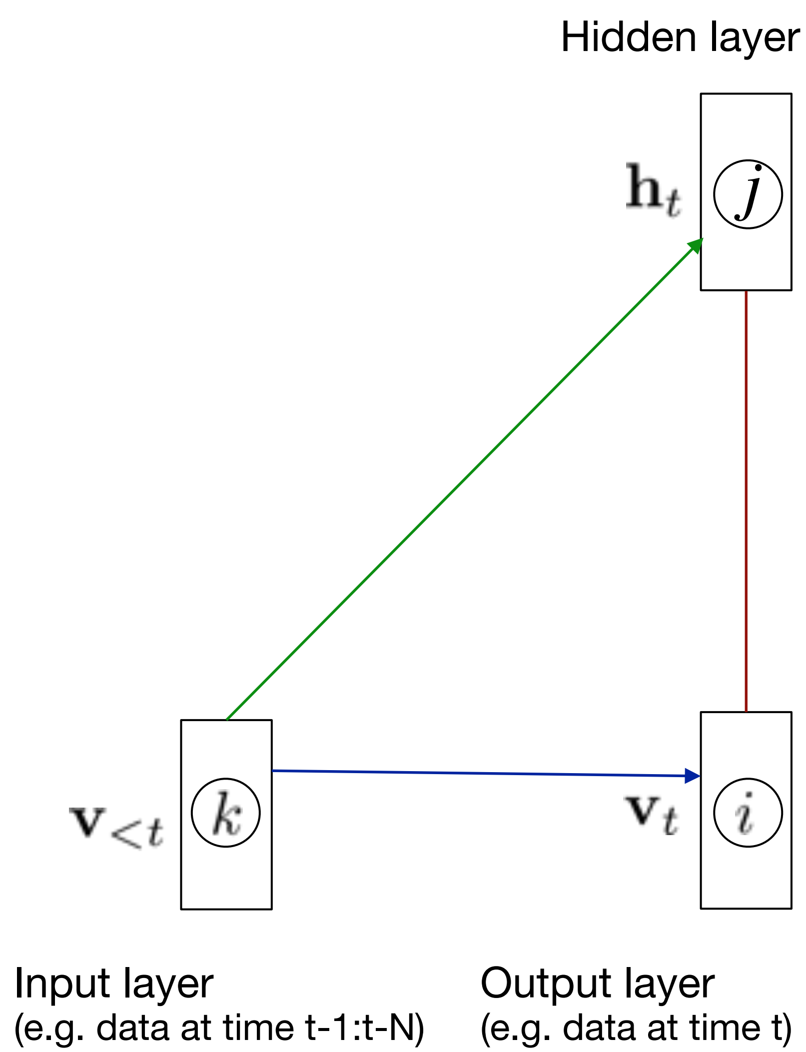
(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)

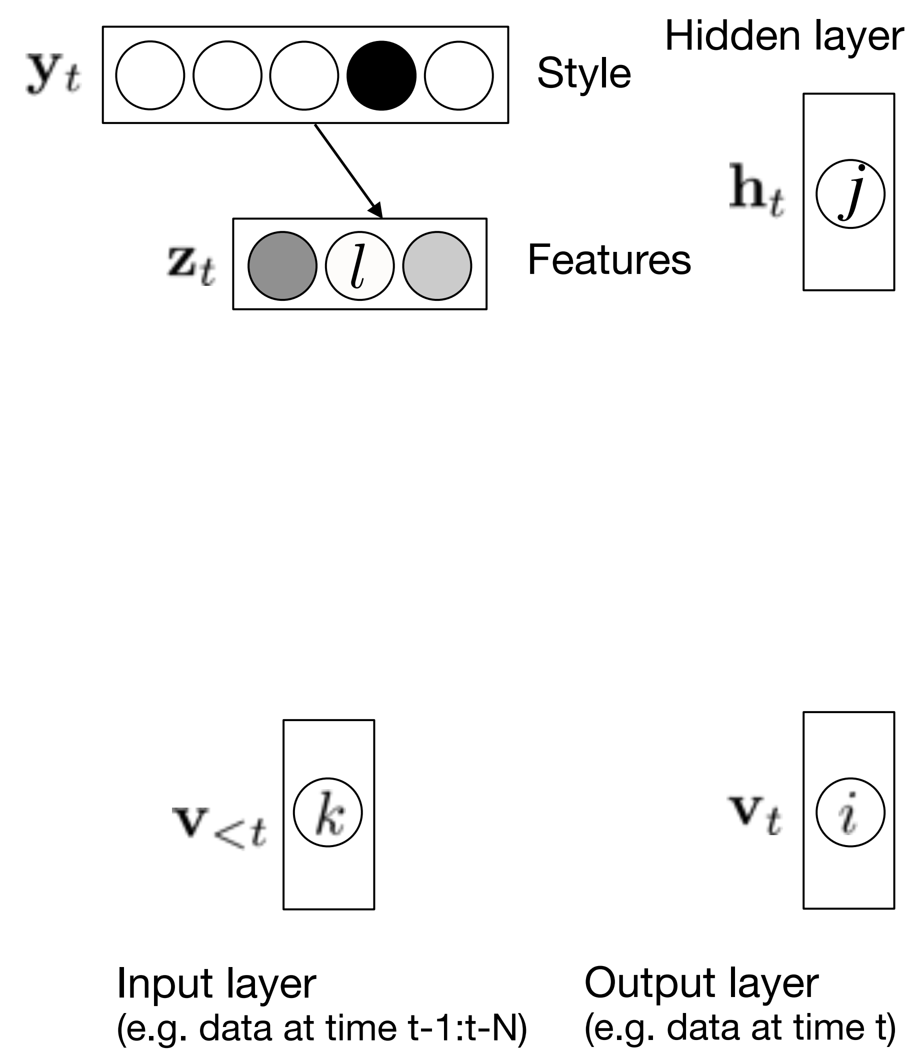




SUPERVISED MODELING OF STYLE

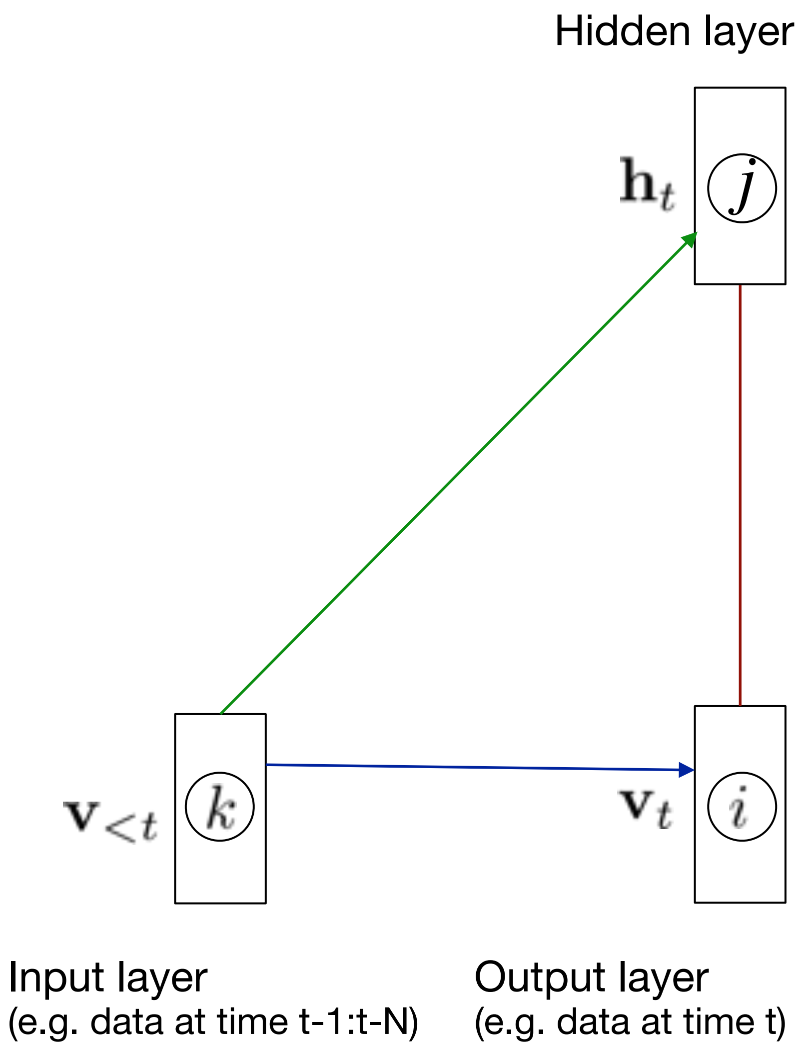
(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)

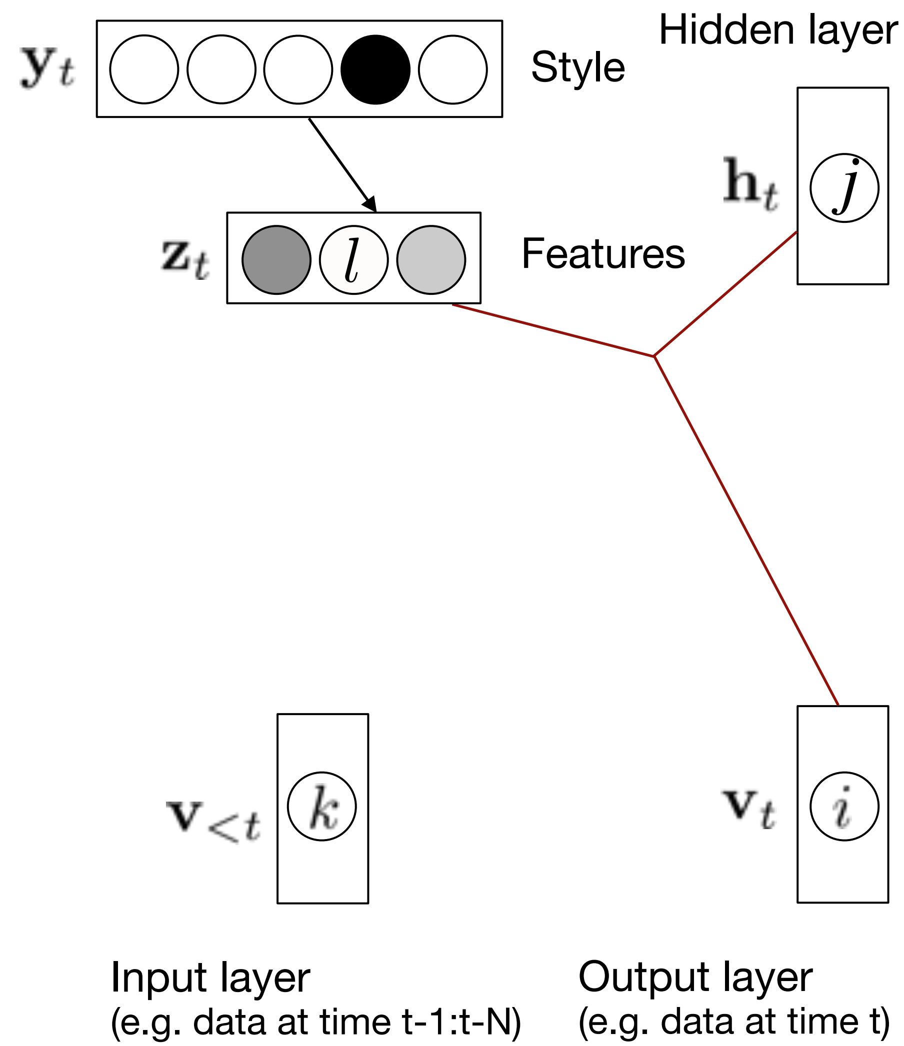




SUPERVISED MODELING OF STYLE

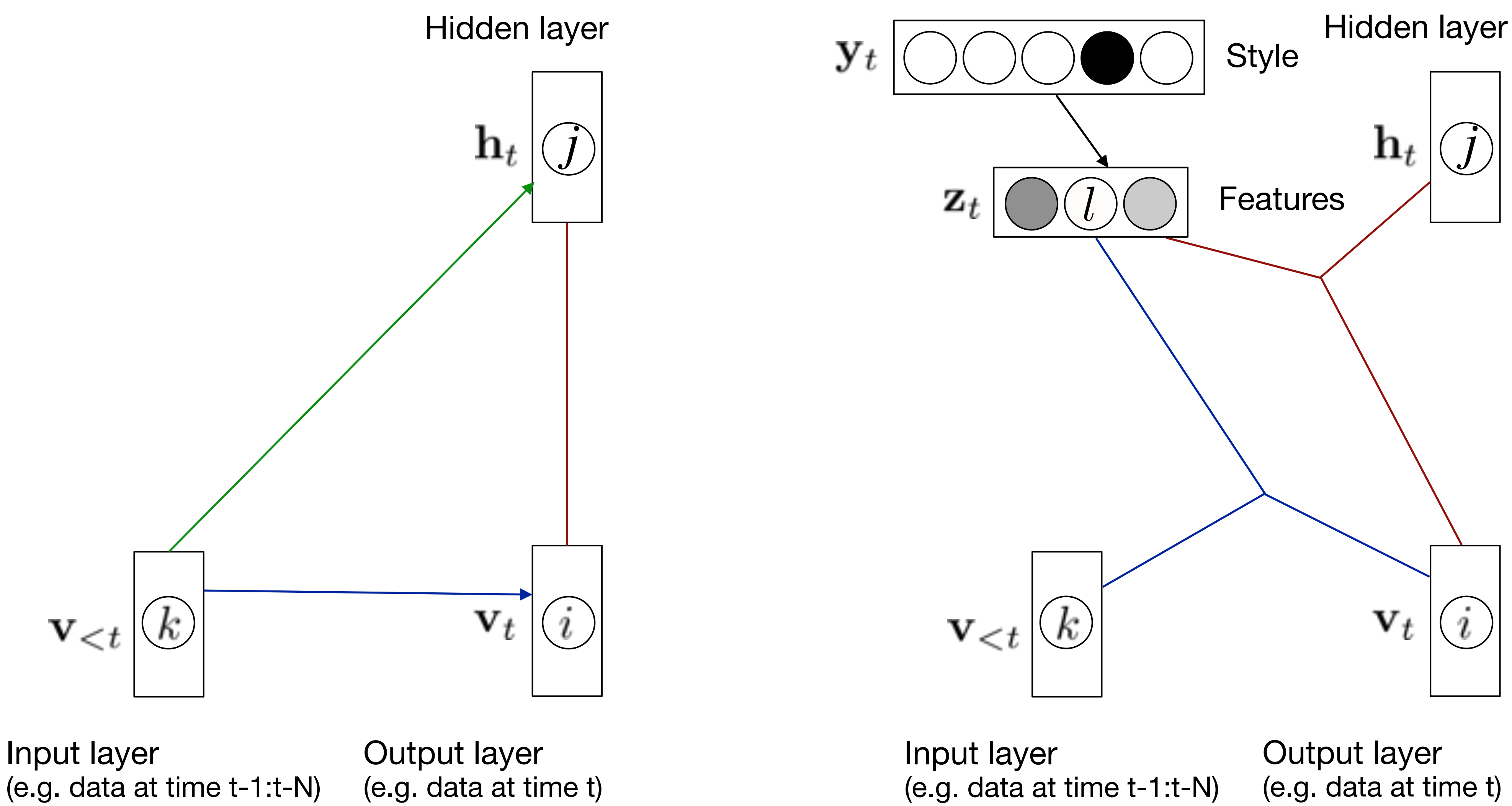
(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)





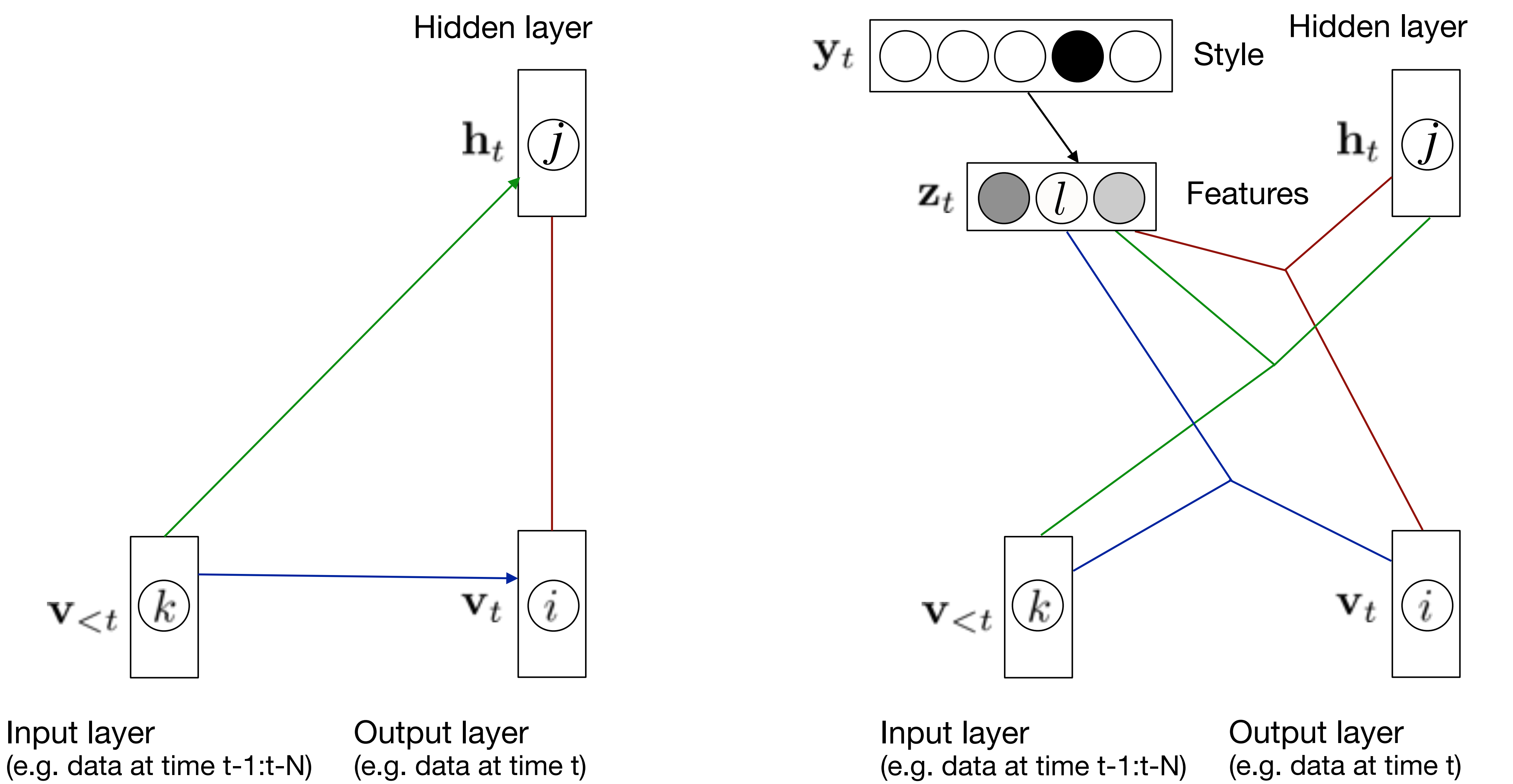
SUPERVISED MODELING OF STYLE

(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)



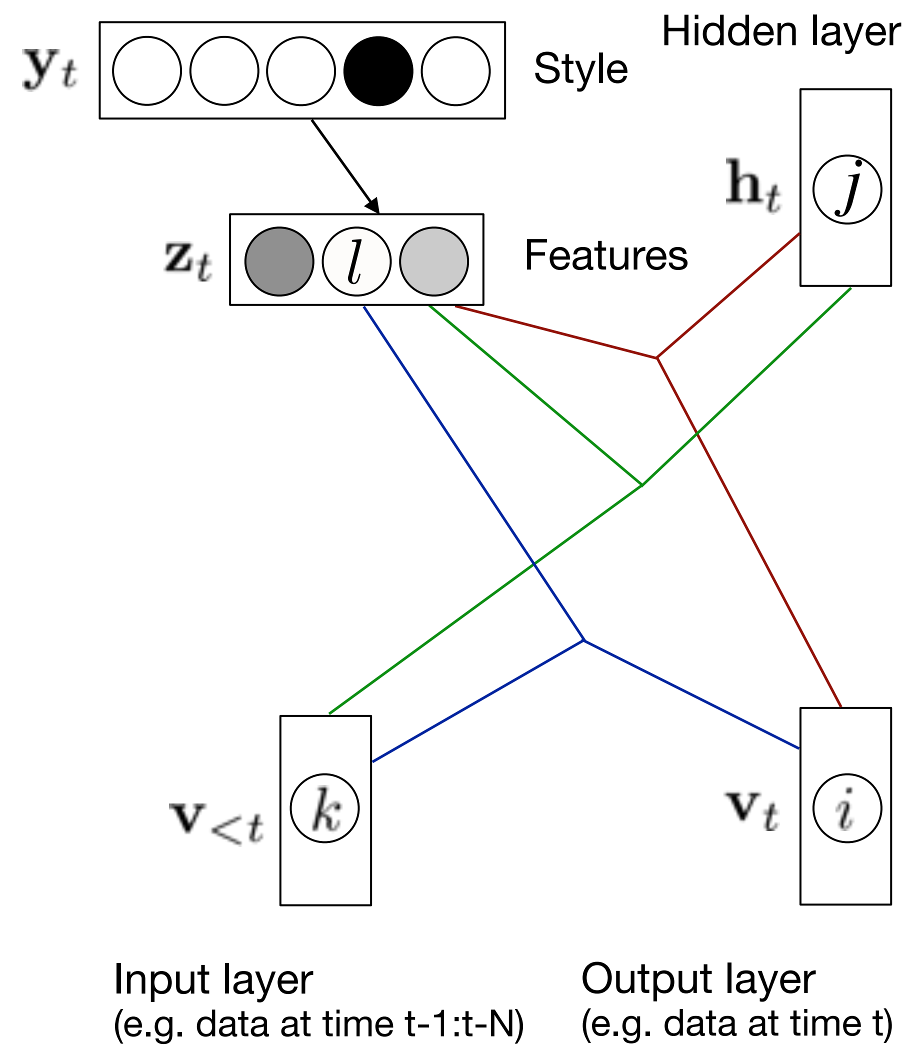
SUPERVISED MODELING OF STYLE

(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)

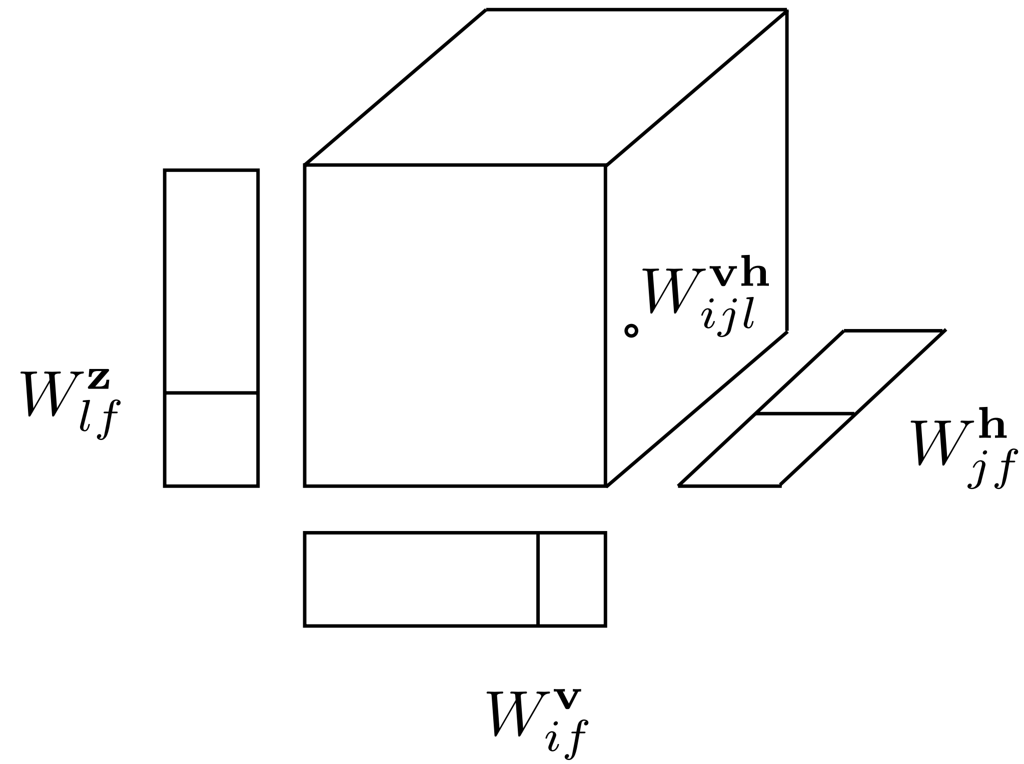


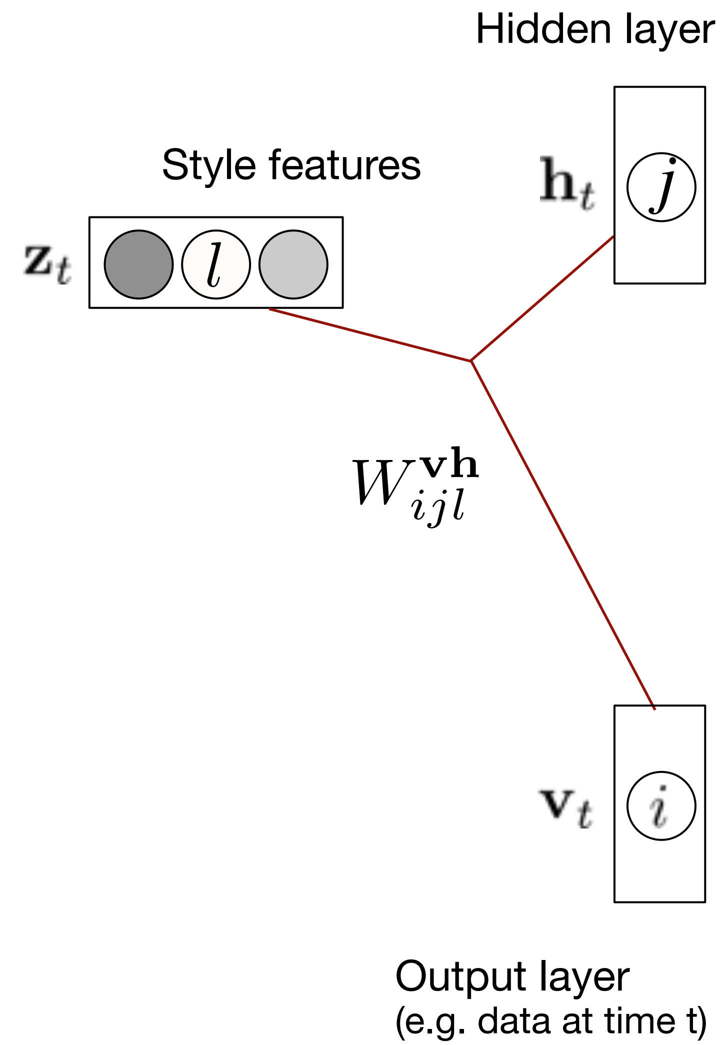
OVERPARAMETERIZATION

- Note: weight Matrix $W^{\mathbf{v},\mathbf{h}}$ has been replaced by a tensor $W^{\mathbf{v},\mathbf{h},\mathbf{z}}$! (Likewise for other weights)
- The number of parameters is $O(N^3)$ - per group of weights
- More, if we want sparse, overcomplete hiddens
- However, there is a simple yet powerful solution!



FACTORING





$$W^{\mathbf{vh}}_{ijl} = \sum_f W^{\mathbf{v}}_{if} W^{\mathbf{h}}_{jf} W^{\mathbf{z}}_{lf}$$

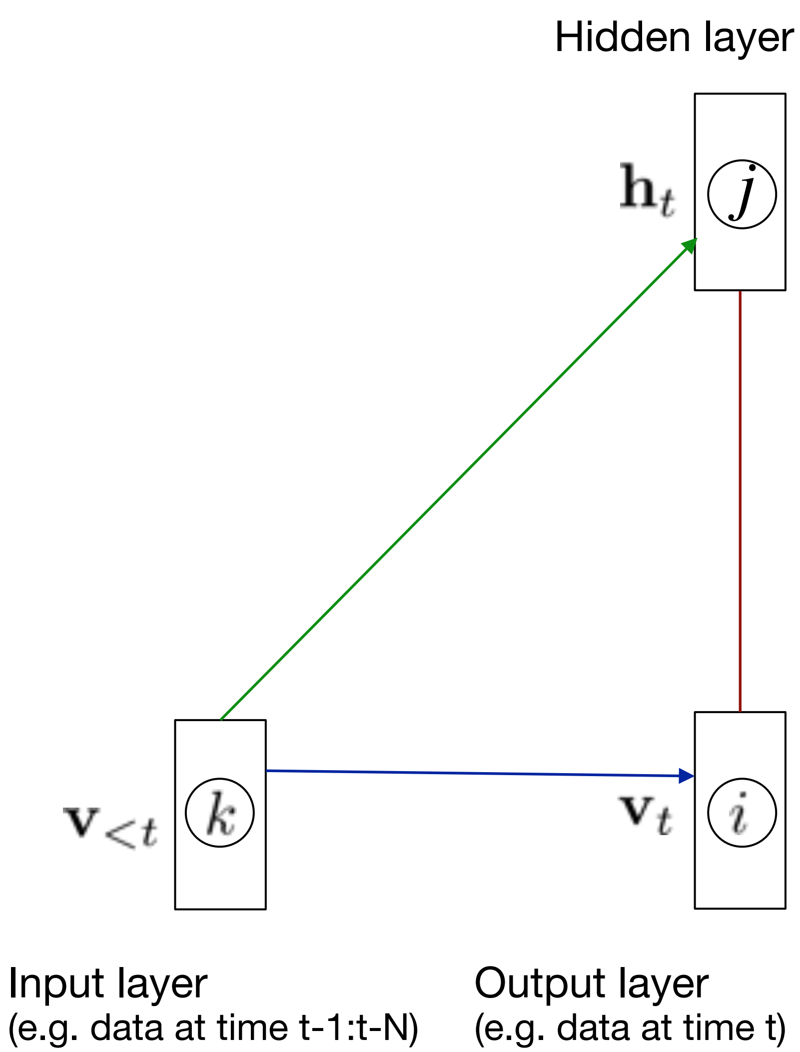
(Figure adapted from Roland Memisevic)

SUPERVISED MODELING OF STYLE

(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)

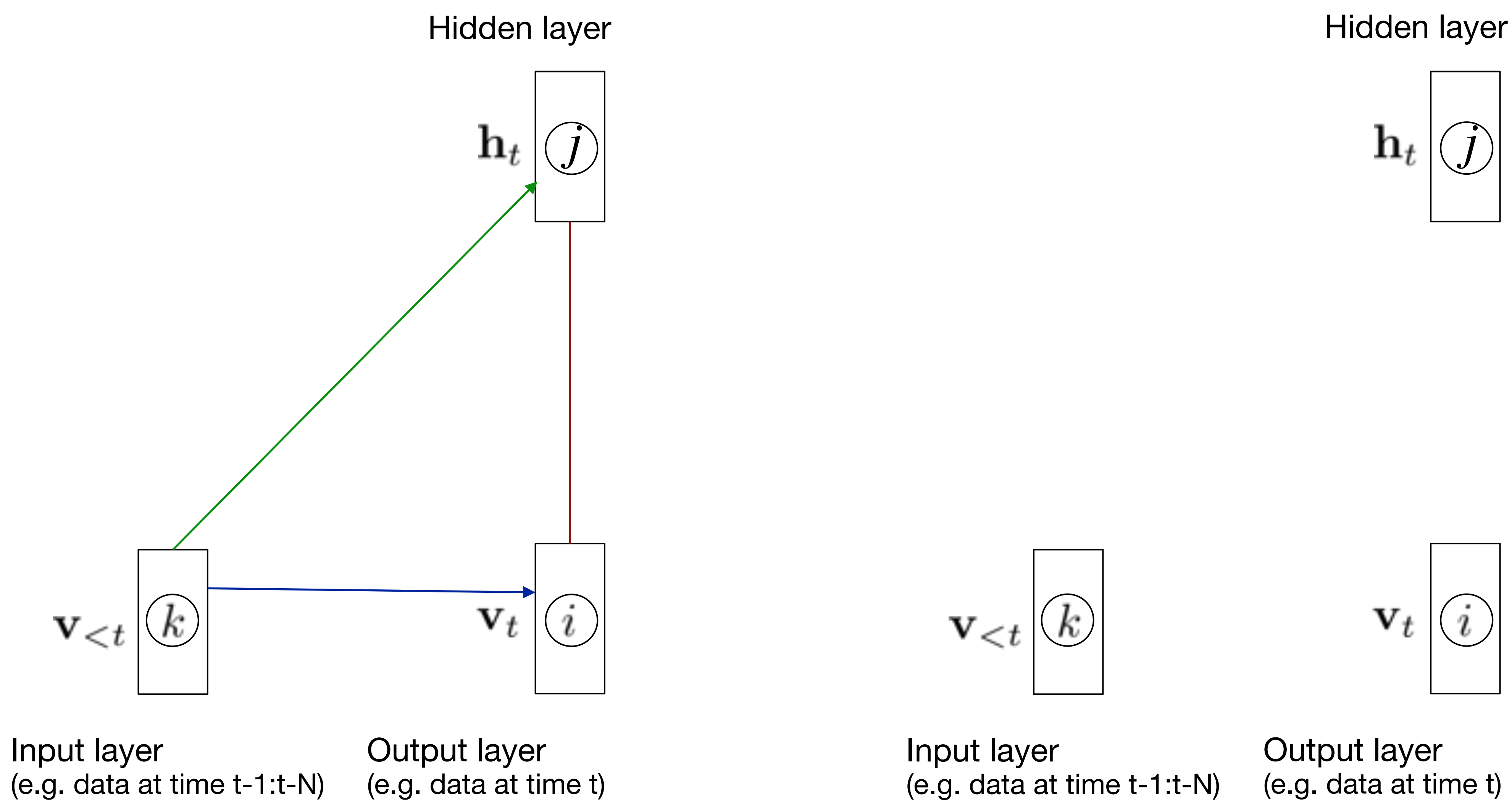
SUPERVISED MODELING OF STYLE

(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)



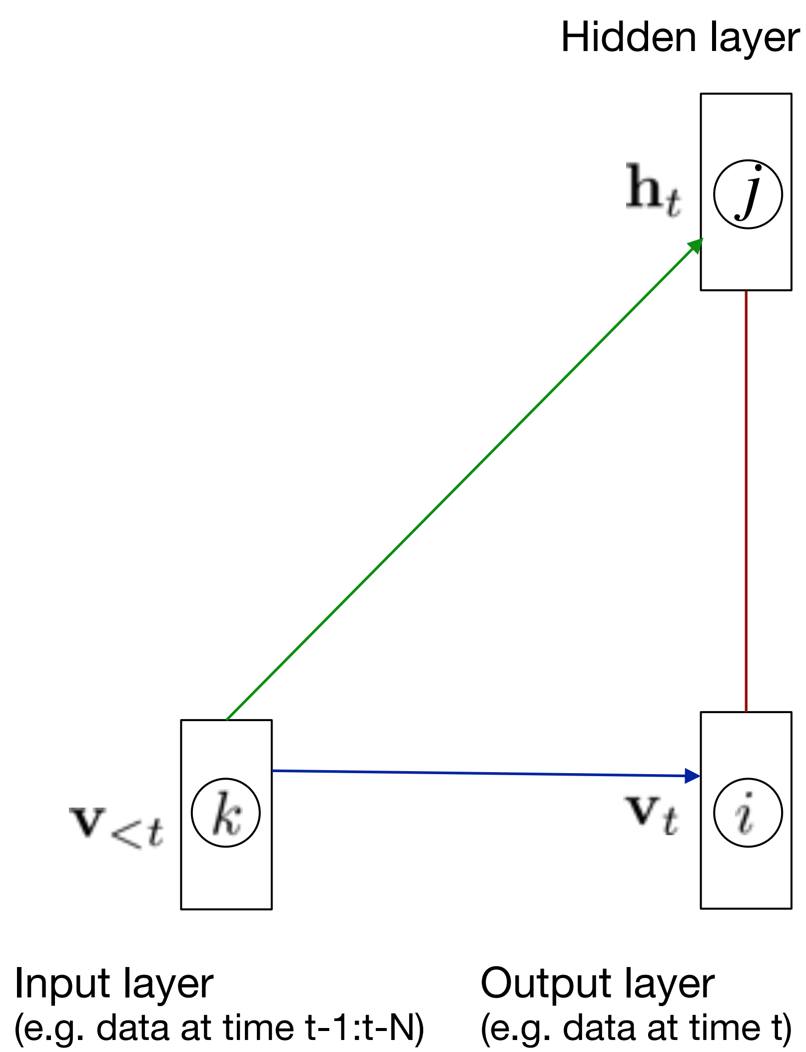
SUPERVISED MODELING OF STYLE

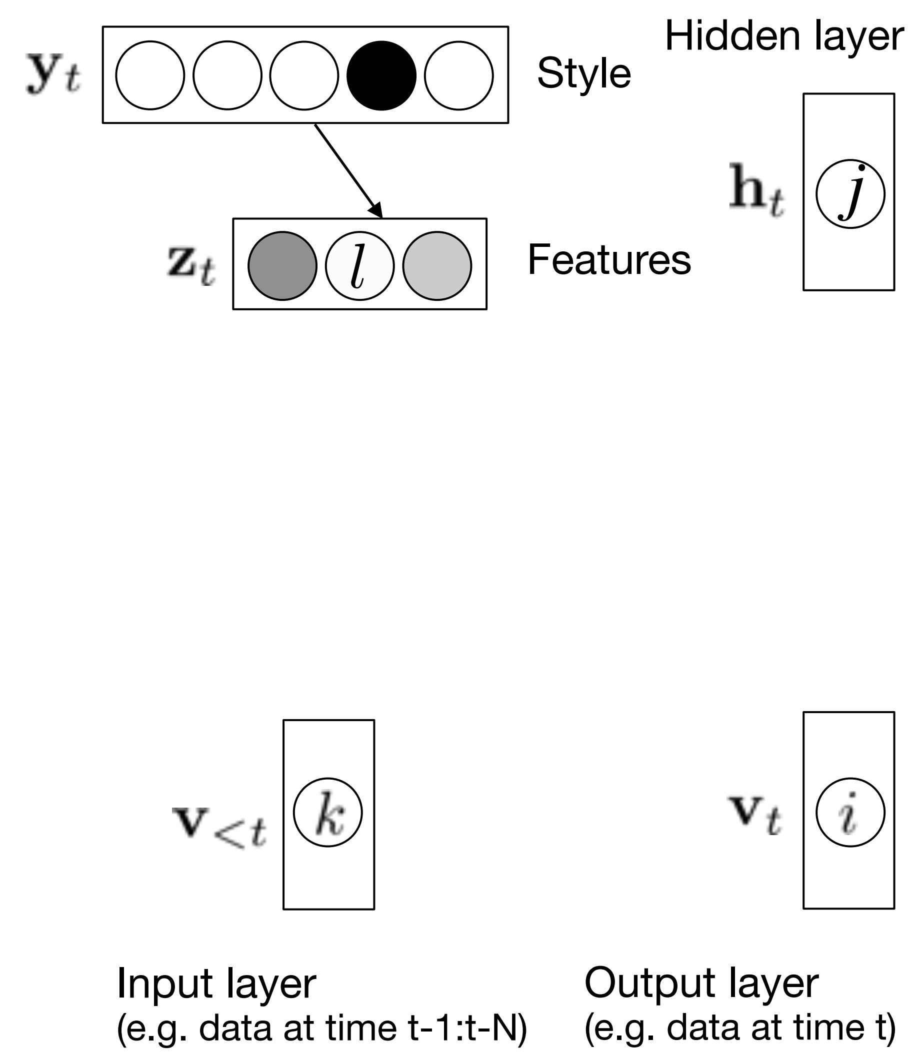
(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)



SUPERVISED MODELING OF STYLE

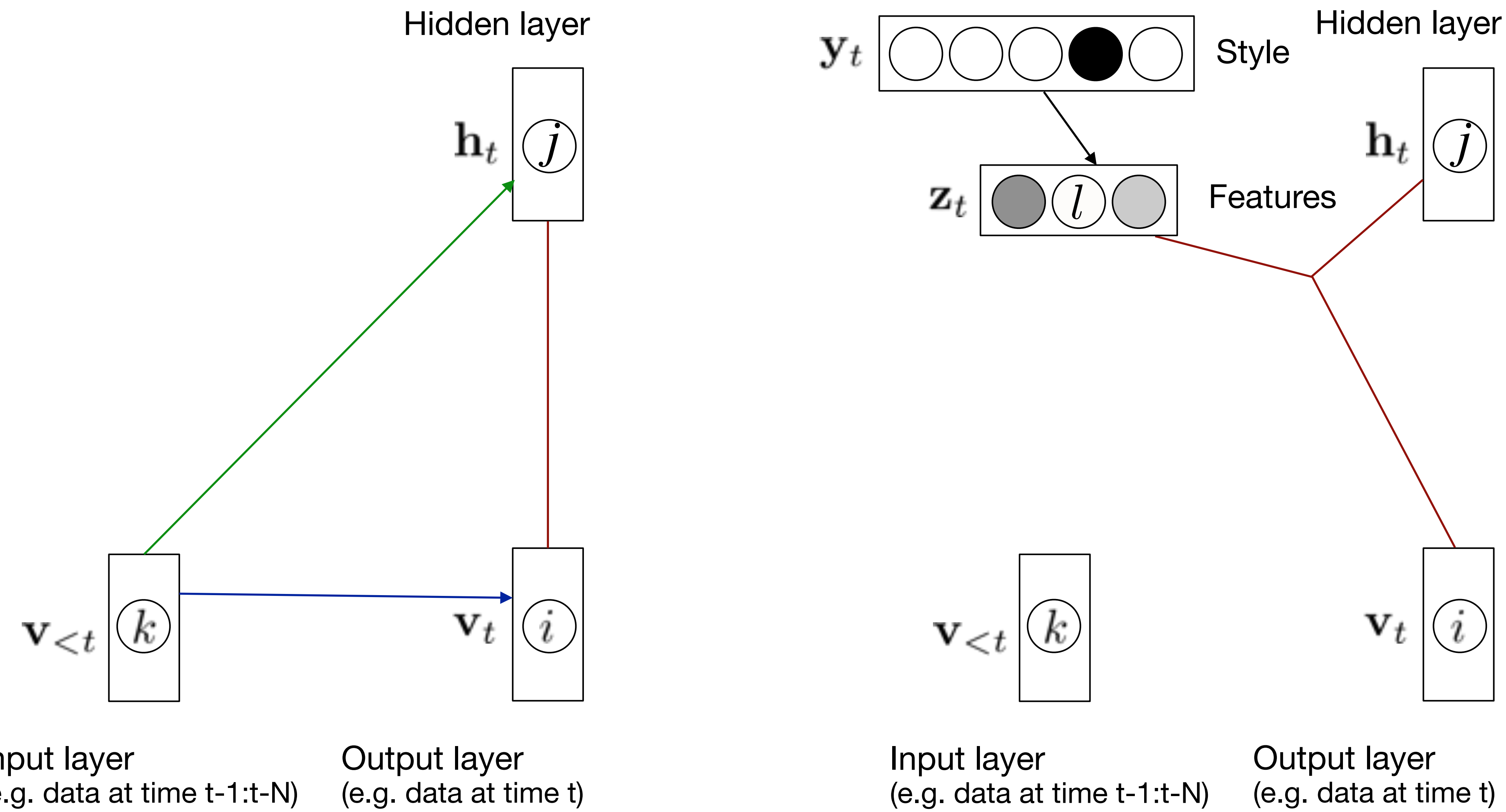
(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)





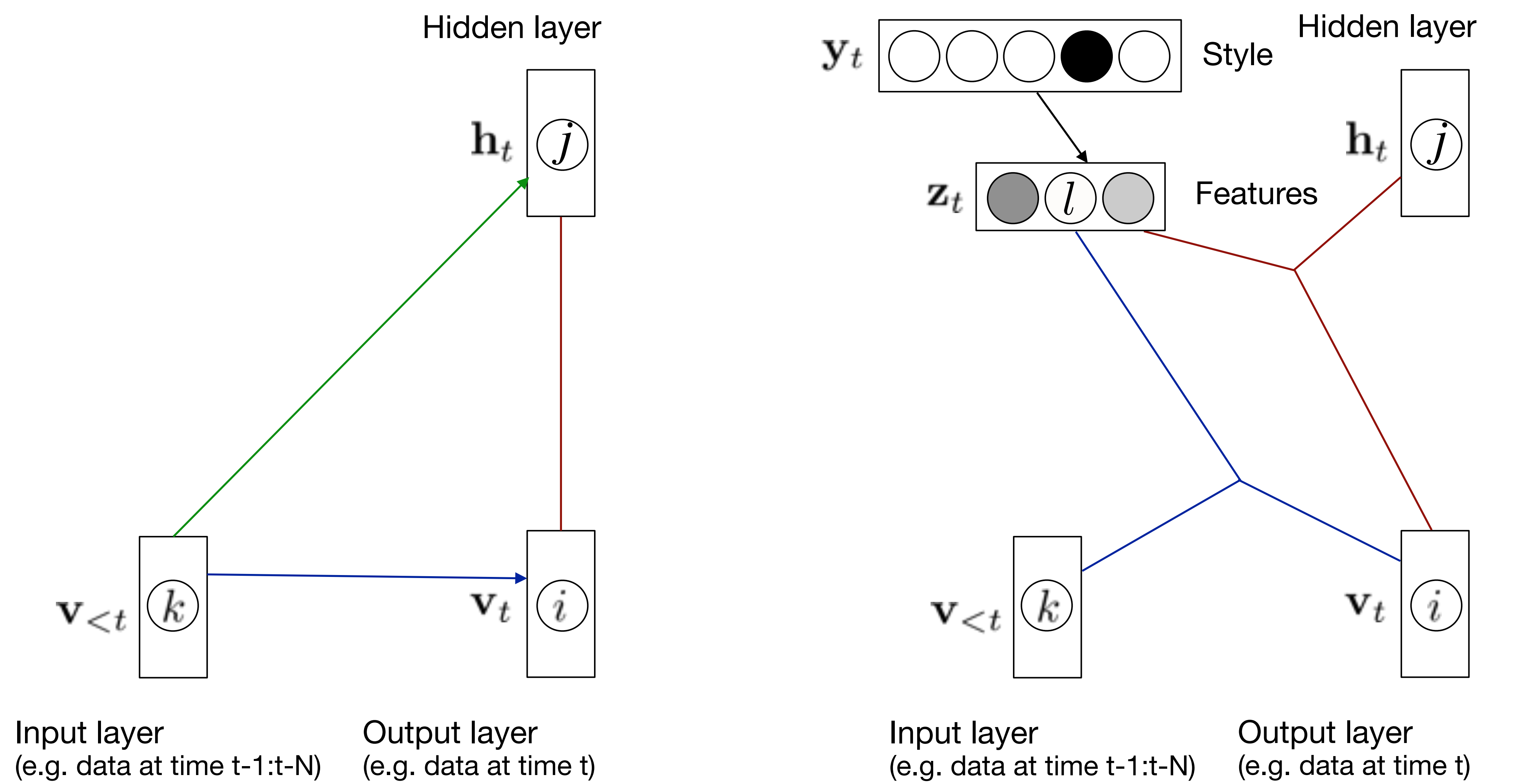
SUPERVISED MODELING OF STYLE

(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)



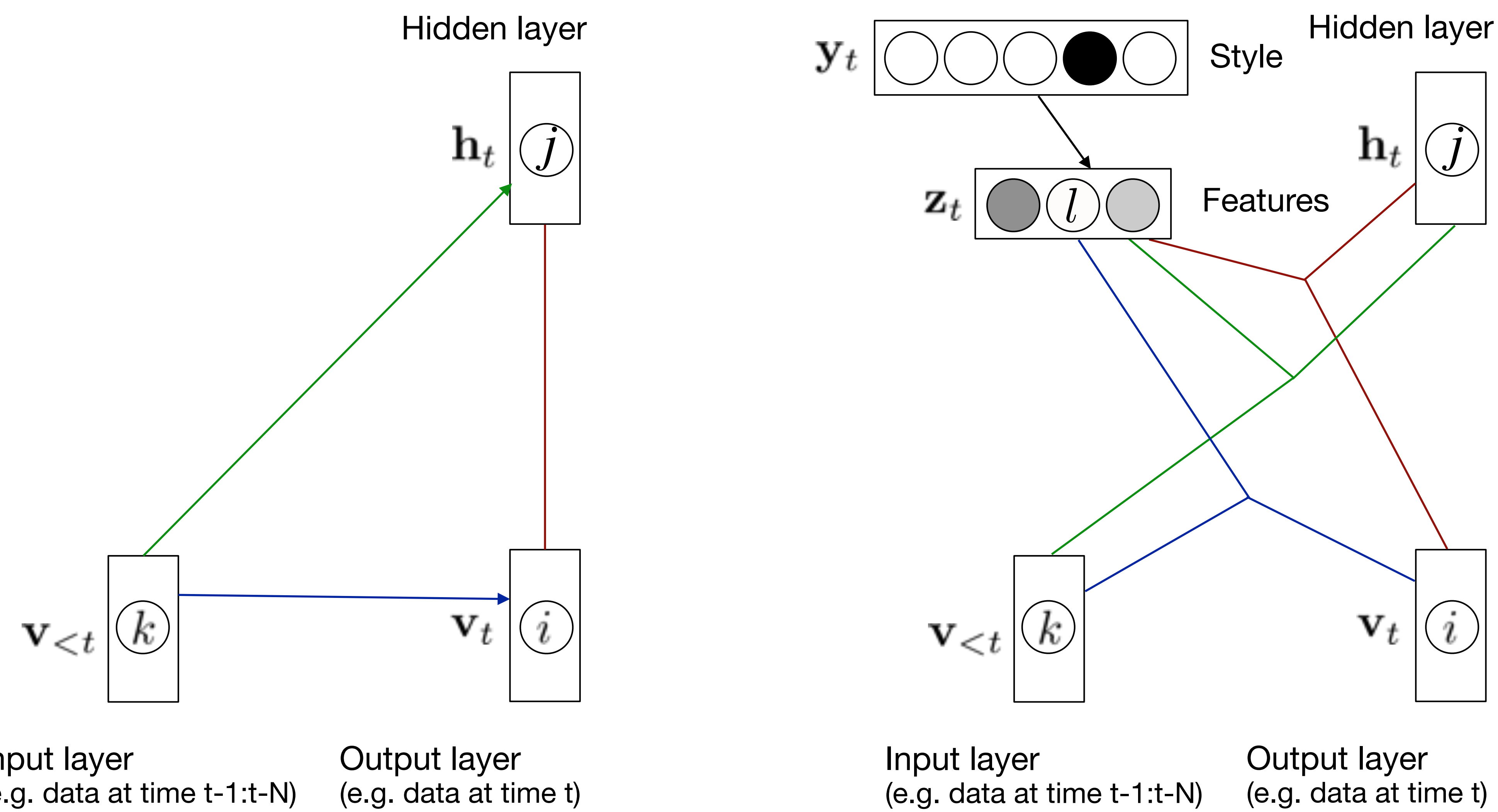
SUPERVISED MODELING OF STYLE

(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)



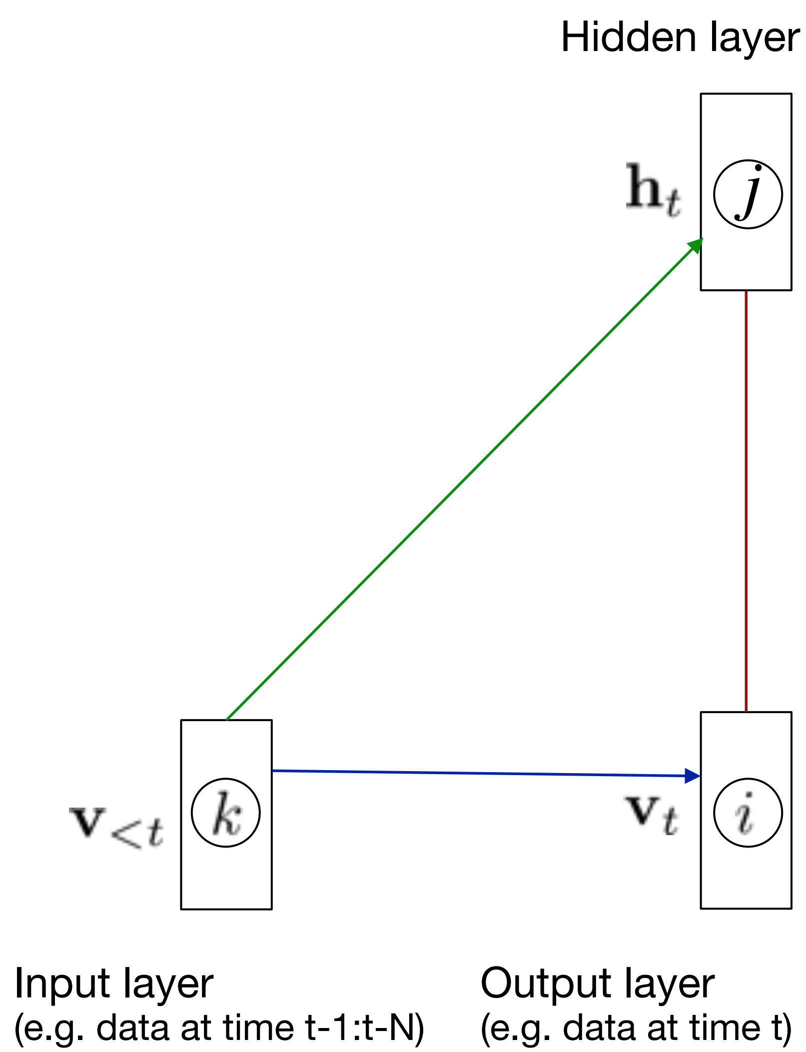
SUPERVISED MODELING OF STYLE

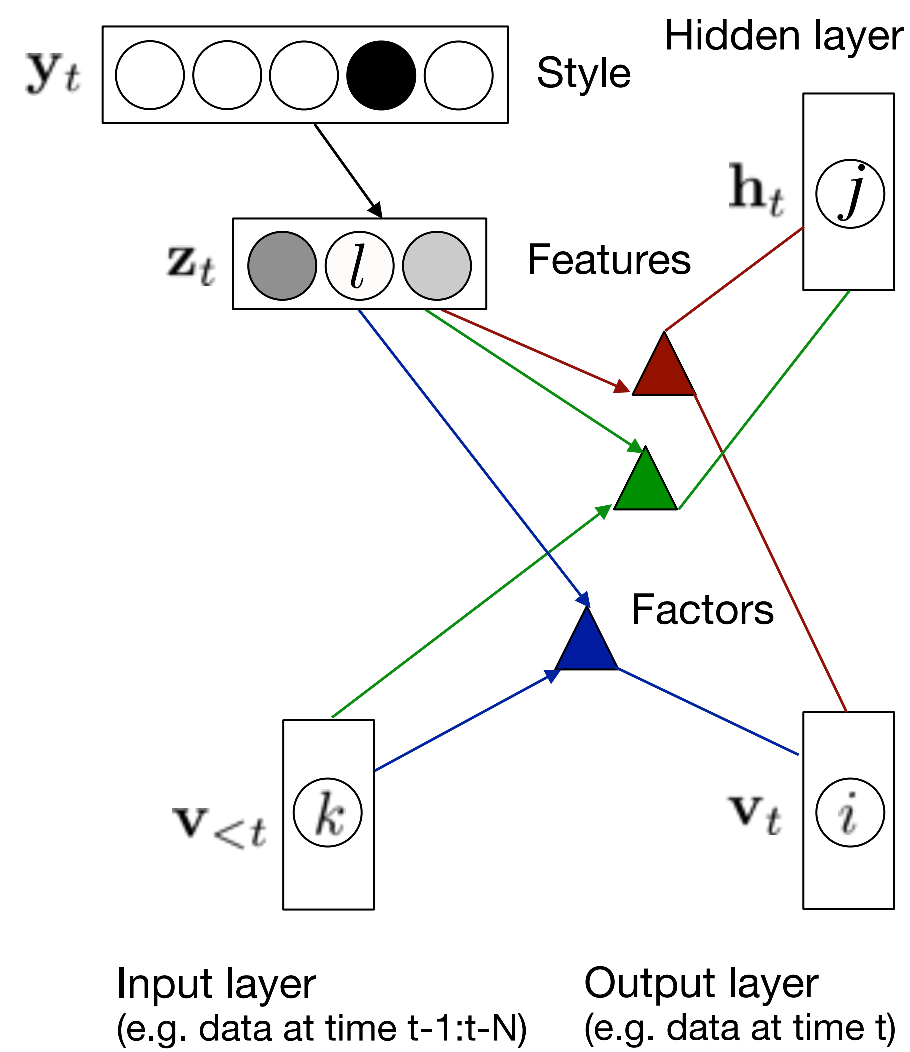
(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)



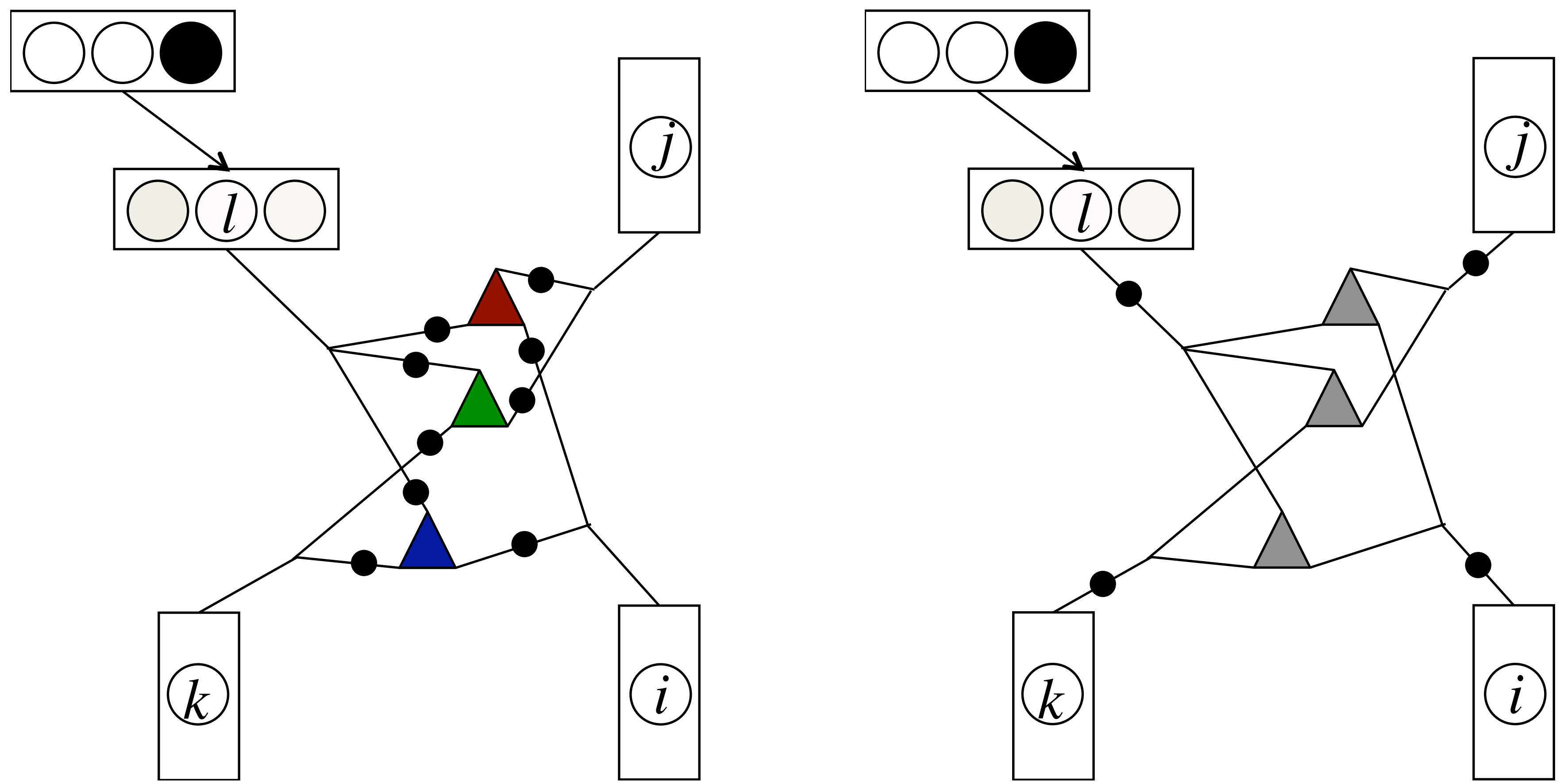
SUPERVISED MODELING OF STYLE

(Taylor, Hinton and Roweis ICML 2009, JMLR 2011)

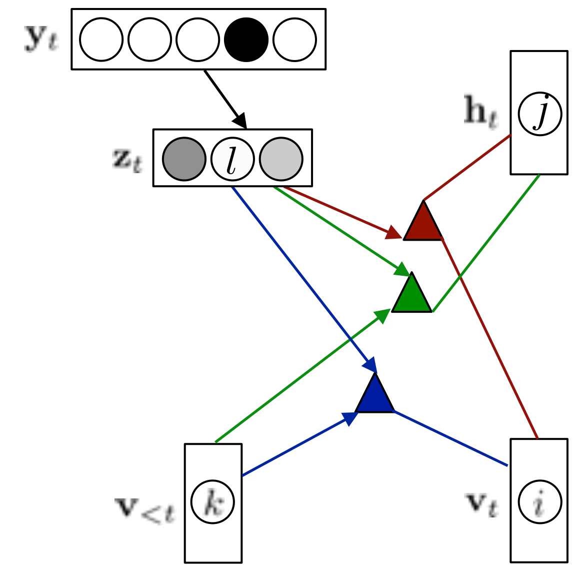




PARAMETER SHARING

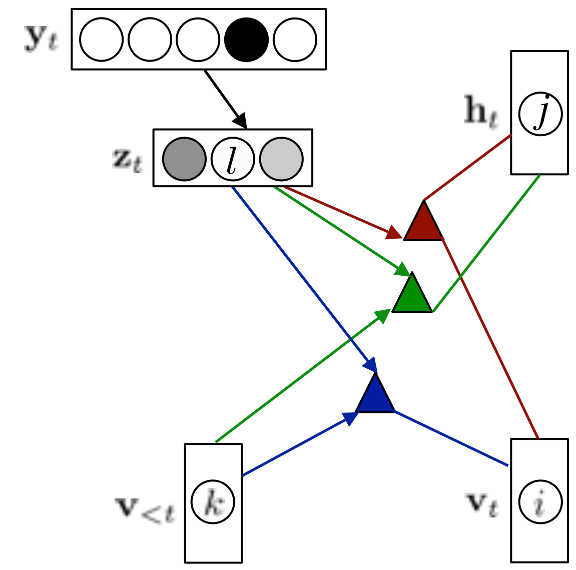


MOTION SYNTHESIS: FACTORED 3RD-ORDER CRBM

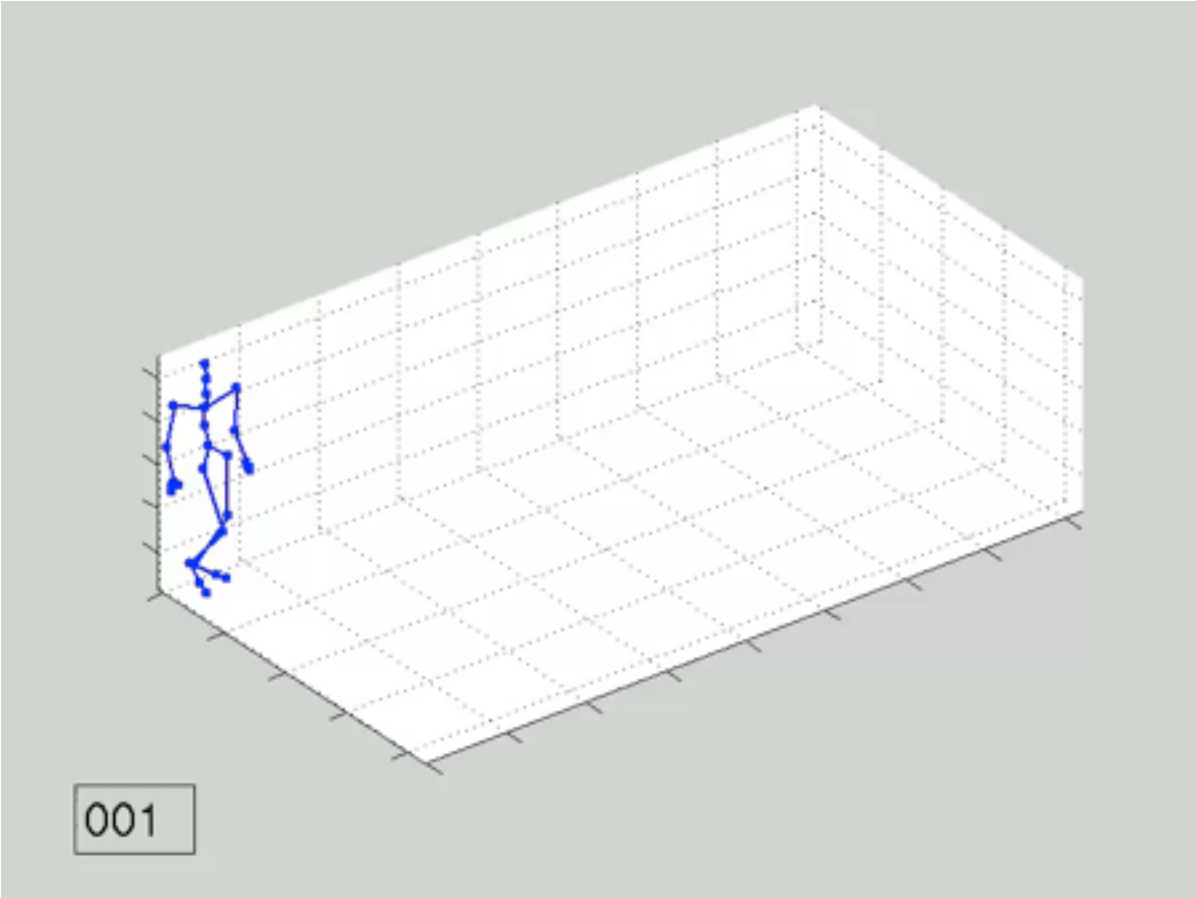


- Same 10-styles dataset
- 600 binary hidden units
- 3×200 deterministic factors
- 100 real-valued style features
- < 1 hour training on a modern workstation
- Synthesis is real-time

MOTION SYNTHESIS: FACTORED 3RD-ORDER CRBM

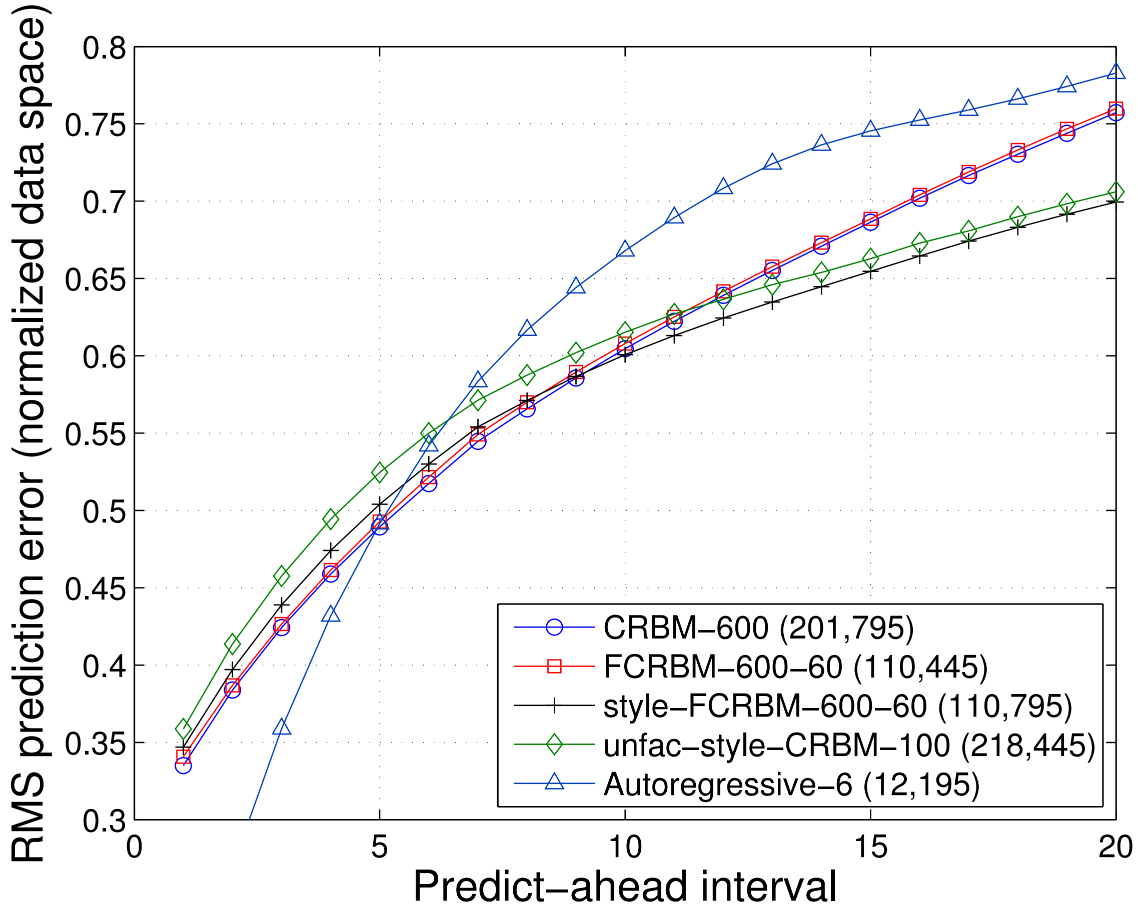


- Same 10-styles dataset
- 600 binary hidden units
- 3×200 deterministic factors
- 100 real-valued style features
- < 1 hour training on a modern workstation
- Synthesis is real-time



QUANTITATIVE EVALUATION

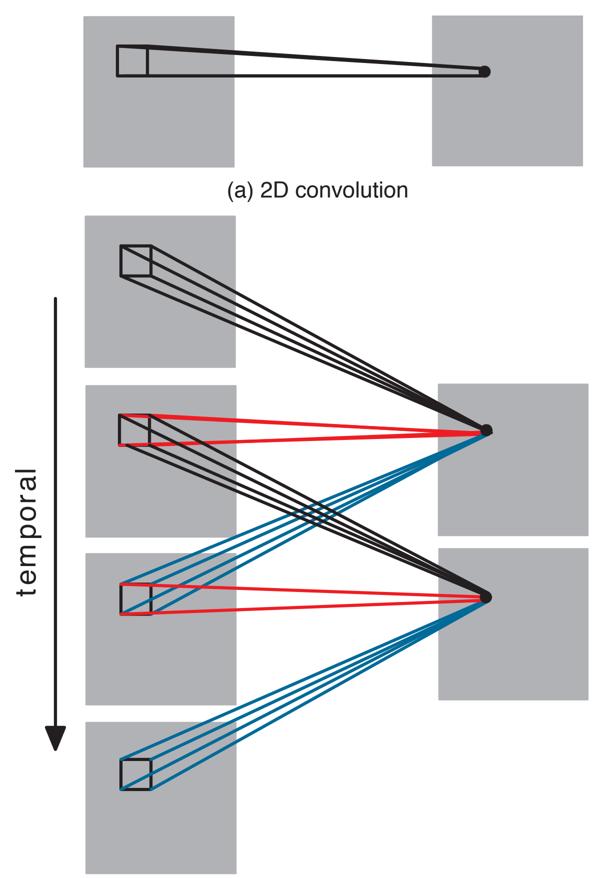
- Not computationally tractable to compute likelihoods
- Annealed Importance Sampling will not work in conditional models (open problem)
- Can evaluate predictive power (even though it has been trained generatively)
- Can also evaluate in denoising tasks



3D CONVNETS FOR ACTIVITY RECOGNITION

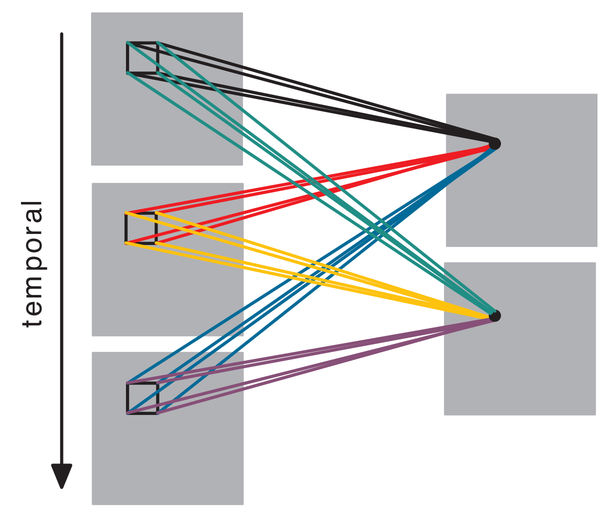
Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu (ICML 2010)

- One approach: treat video frames as still images (LeCun et al. 2005)
- Alternatively, perform 3D convolution so that discriminative features across space and time are captured



(a) 2D convolution

(b) 3D convolution



Multiple convolutions applied to contiguous frames to extract multiple features

Images from Ji et al. 2010

3D CNN ARCHITECTURE

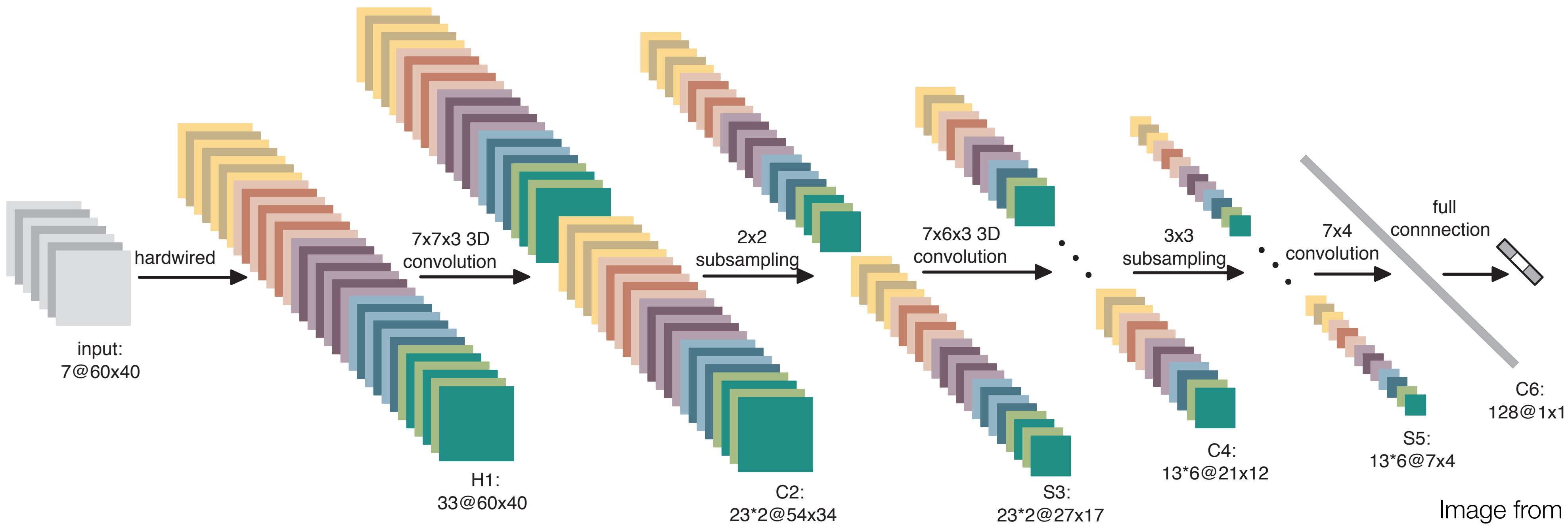


Image from Ji et al. 2010

Hardwired to extract:
1)grayscale
2)grad-x
3)grad-y
4)flow-x
5)flow-y

2 different 3D filters applied to each of 5 blocks independently

Subsample spatially

3 different 3D filters applied to each of 5 channels in 2 blocks

Two fully-connected layers

Action units

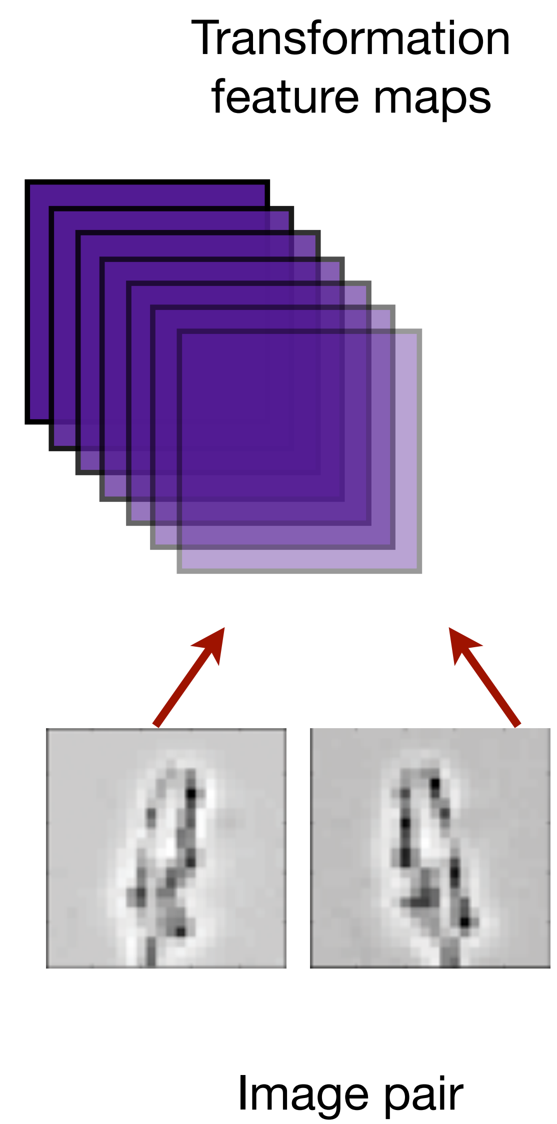
3D CONVNET: DISCUSSION

- Good performance on TRECVID surveillance data (*CellToEar, ObjectPut, Pointing*)
- Good performance on KTH actions (*box, handwave, handclap, jog, run, walk*)
- Still a fair amount of engineering: person detection (TRECVID), foreground extraction (KTH), hard-coded first layer

Image from Ji et al. 2010

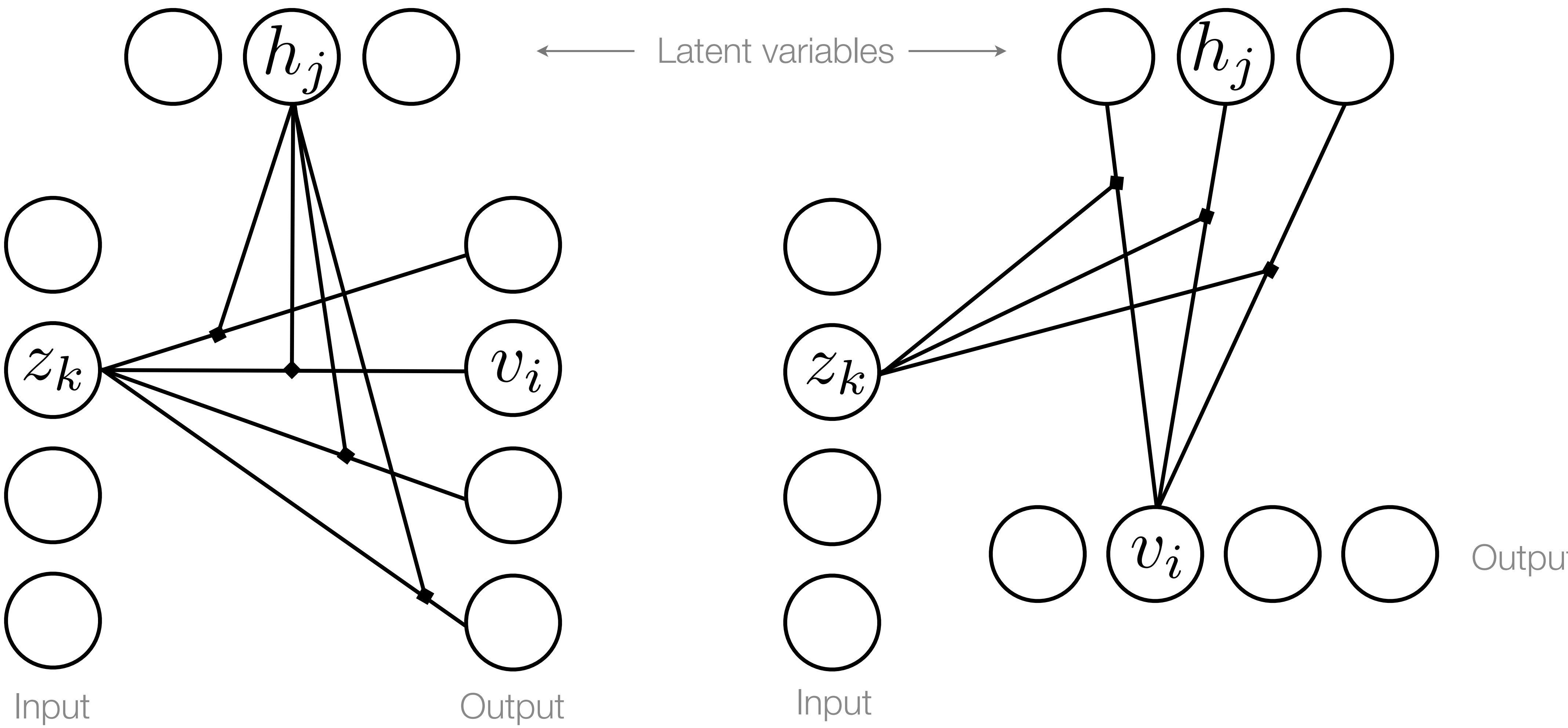
LEARNING FEATURES FOR VIDEO UNDERSTANDING

- Most work on unsupervised feature extraction has concentrated on static images
- We propose a model that extracts motion-sensitive features from pairs of images
- Existing attempts (e.g. Memisevic & Hinton 2007, Cadieu & Olshausen 2009) ignore the *pictorial* structure of the input
- Thus limited to modeling small image patches



GATED RESTRICTED BOLTZMANN MACHINES (GRBM)

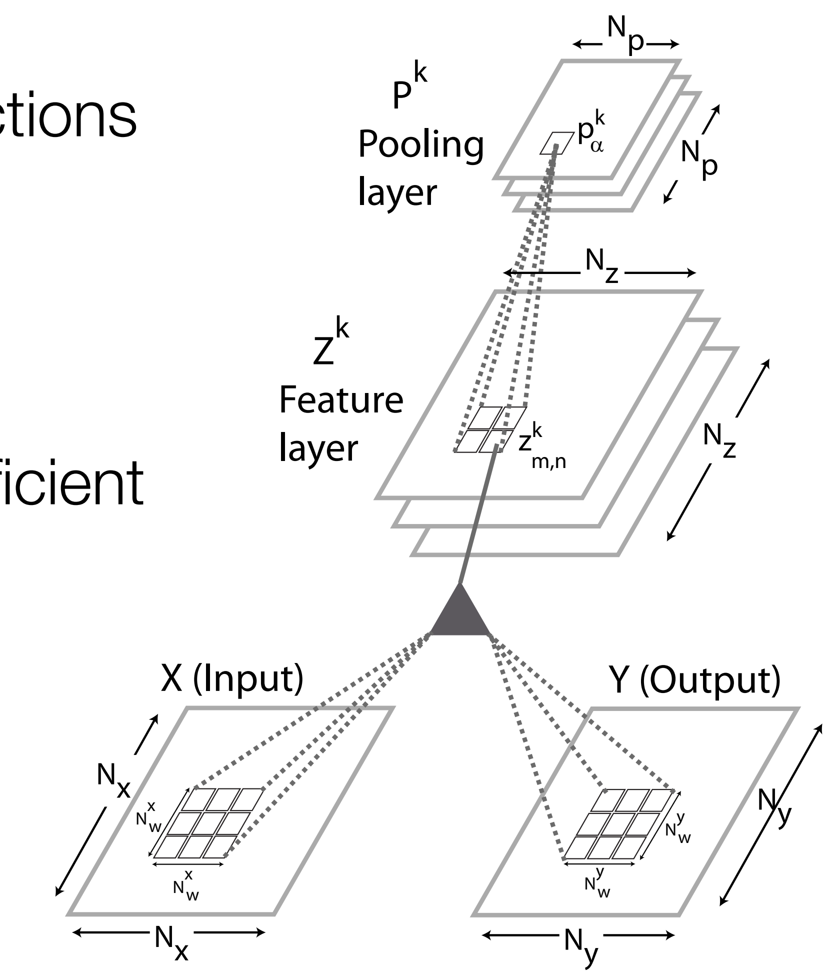
Two views: Memisevic & Hinton (2007)



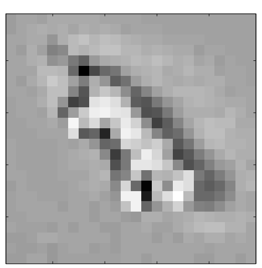
CONVOLUTIONAL GRBM

Graham Taylor, Rob Fergus, Yann LeCun, and Chris Bregler (ECCV 2010)

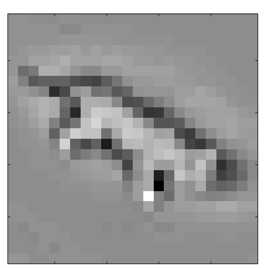
- Like the GRBM, captures third-order interactions
- Shares weights at all locations in an image
- As in a standard RBM, exact inference is efficient
- Inference and reconstruction are performed through convolution operations



MORE COMPLEX EXAMPLE OF "ANALOGIES"



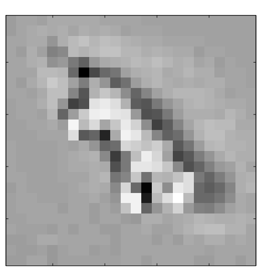
Input



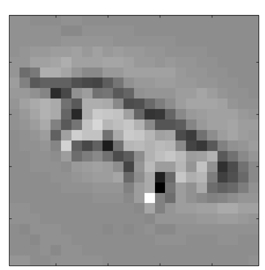
Output

MORE COMPLEX EXAMPLE OF “ANALOGIES”

Feature maps

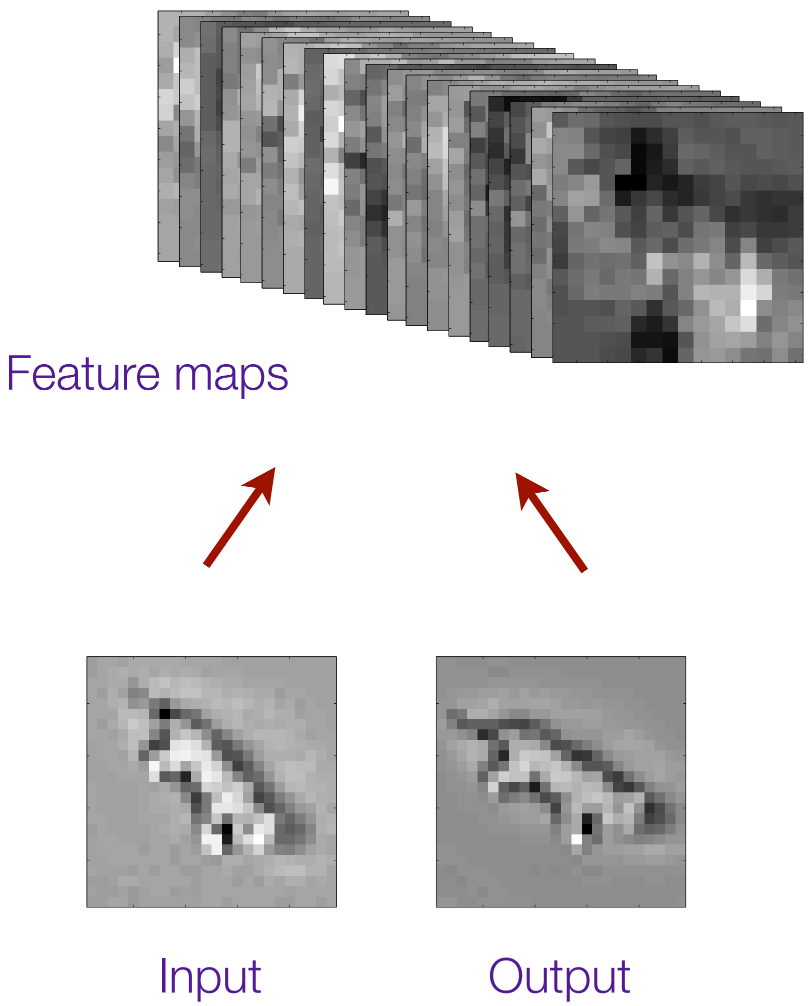


Input

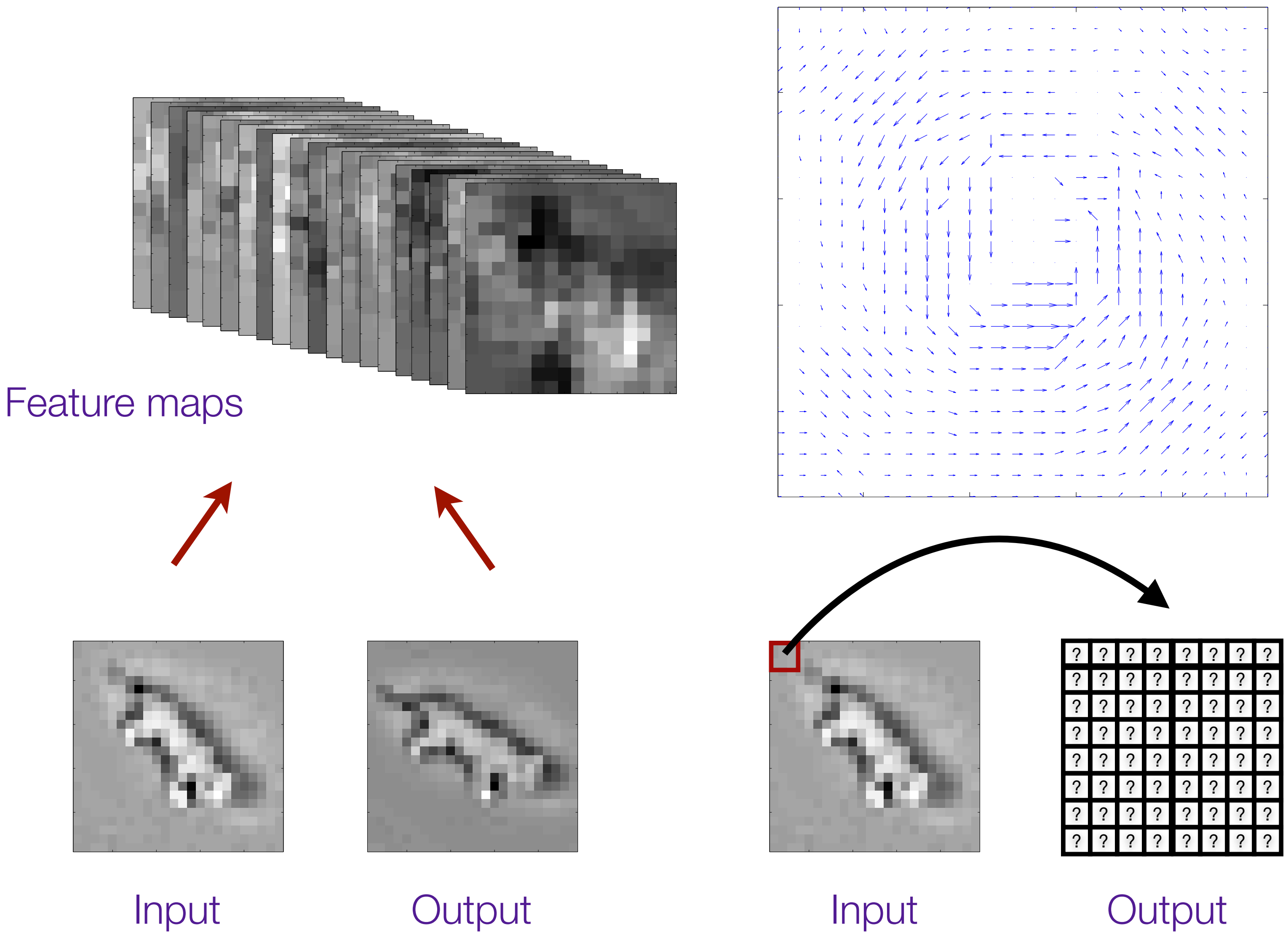


Output

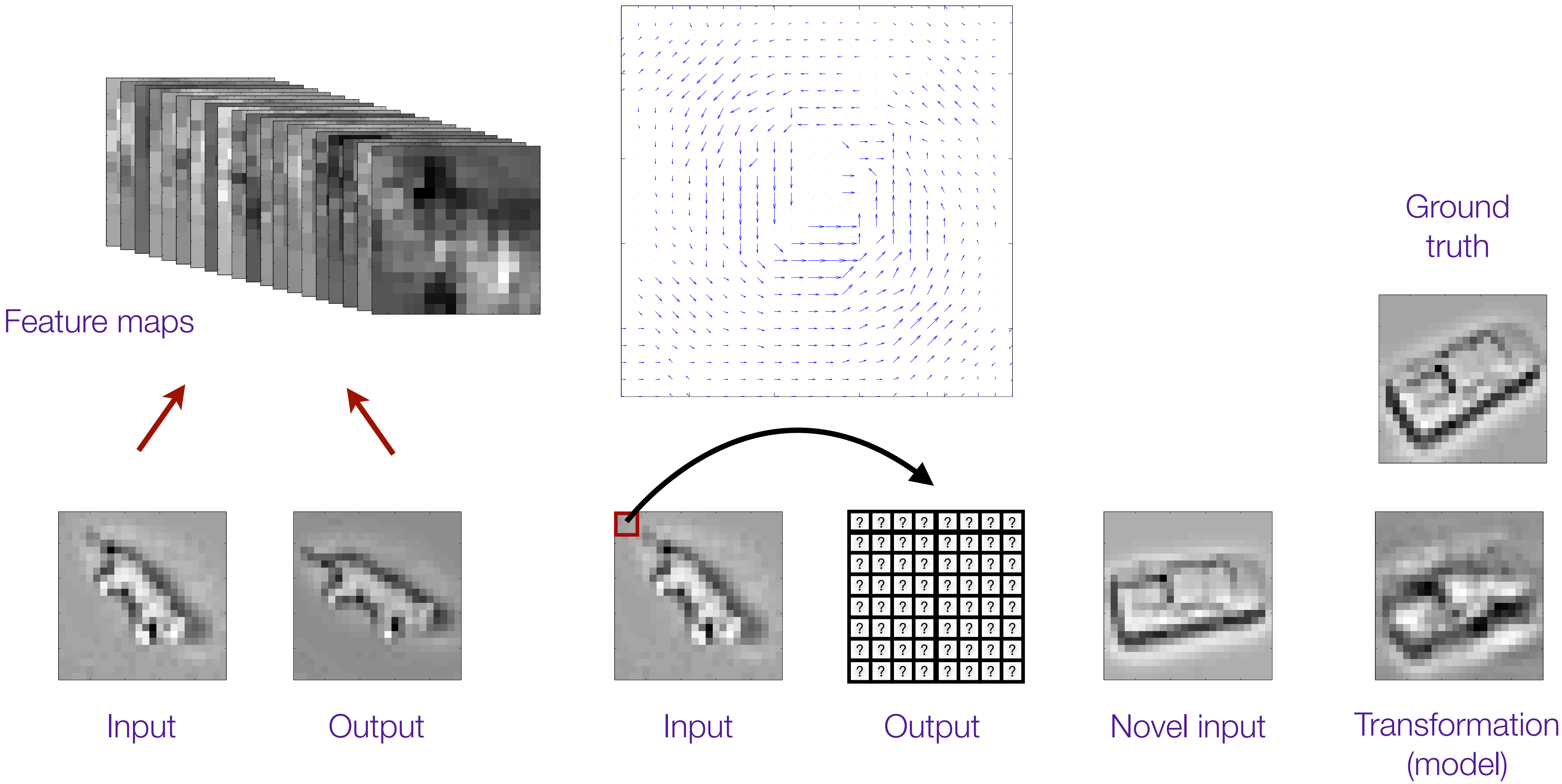
MORE COMPLEX EXAMPLE OF "ANALOGIES"



MORE COMPLEX EXAMPLE OF "ANALOGIES"

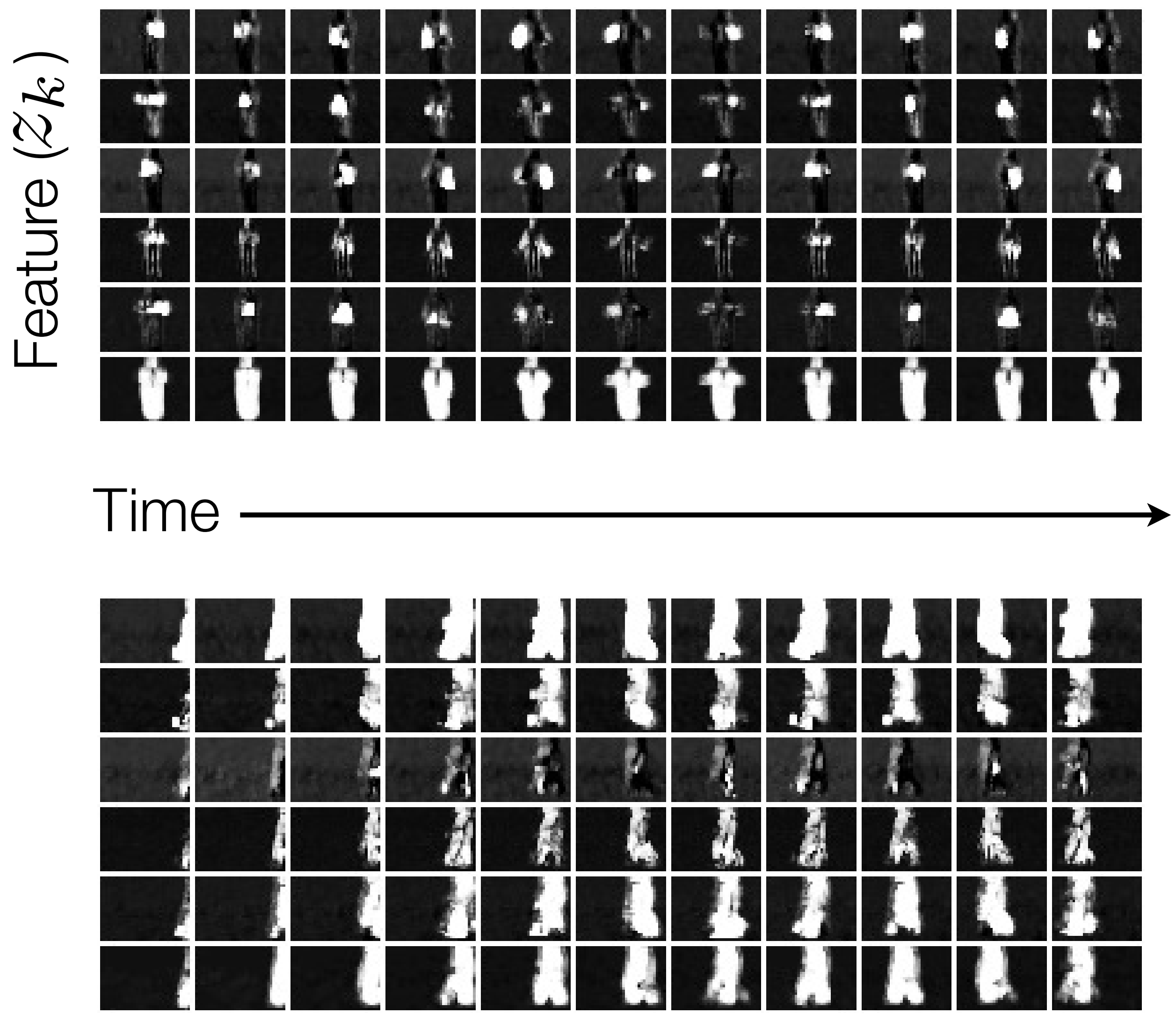


MORE COMPLEX EXAMPLE OF “ANALOGIES”



HUMAN ACTIVITY: KTH ACTIONS DATASET

- We learn 32 feature maps
- 6 are shown here
- KTH contains 25 subjects performing 6 actions under 4 conditions
- Only preprocessing is local contrast normalization
- Motion sensitive features (1,3)
- Edge features (4)
- Segmentation operator (6)



Hand clapping (above); Walking (below)

ACTIVITY RECOGNITION: KTH

Prior Art	Acc (%)	Convolutional architectures	Acc. (%)
HOG3D+KM+SVM	85.3	convGRBM+3D-convnet+logistic reg.	88.9
HOG/HOF+KM+SVM	86.1	convGRBM+3D convnet+MLP	**90.0**
HOG+KM+SVM	79.0	3D convnet+3D convnet+logistic reg.	79.4
HOF+KM+SVM	88.0	3D convnet+3D convnet+MLP	79.5

- Compared to methods that do not use explicit interest point detection
- State of the art: 92.1% (Laptev et al. 2008) 93.9% (Le et al. 2011)
- Other reported result on 3D convnets uses a different evaluation scheme

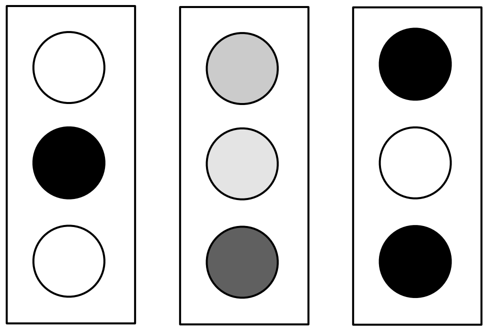
ACTIVITY RECOGNITION: HOLLYWOOD 2

- 12 classes of human action extracted from 69 movies (20 hours)
- Much more realistic and challenging than KTH (changing scenes, zoom, etc.)
- Performance is evaluated by mean average precision over classes

Method	Average Prec.
Prior Art (Wang et al. survey 2009):	
HOG3D+KM+SVM	45.3
HOG/HOF+KM+SVM	**47.4**
HOG+KM+SVM	39.4
HOF+KM+SVM	45.5
Our method:	
GRBM+SC+SVM	**46.8**

SUMMARY

SUMMARY

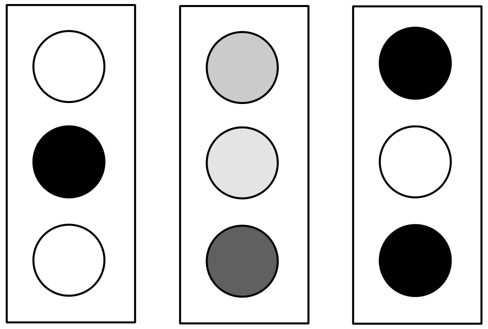


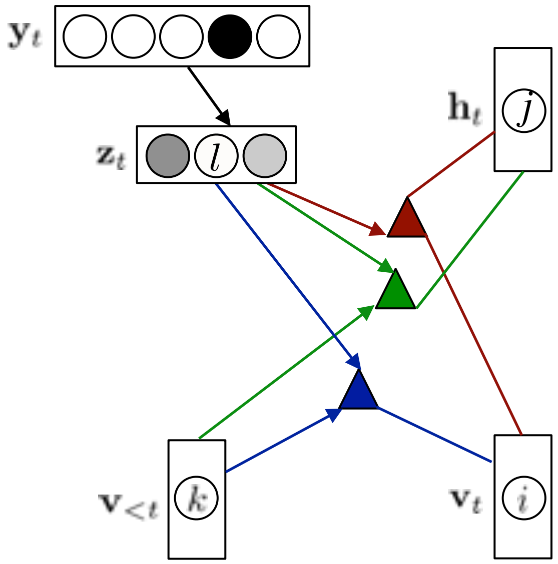
- Learning distributed representations of sequences

SUMMARY

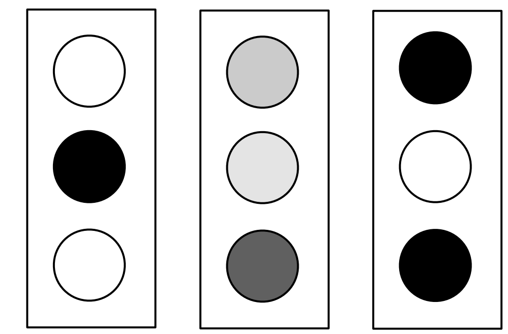
- Learning distributed representations of sequences

- For high-dimensional, multi-modal data: CRBM, FCRBM

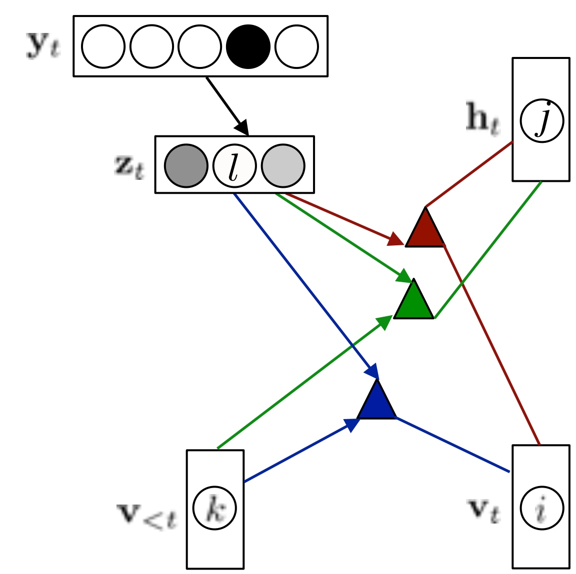




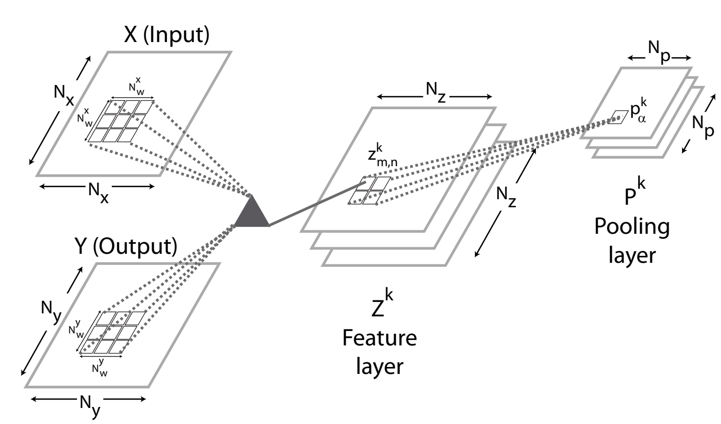
SUMMARY



- Learning distributed representations of sequences



- For high-dimensional, multi-modal data: CRBM, FCRBM



- Activity recognition: 2 methods

The University of Guelph is not in Belgium!