Data Mining & Machine Learning

CS37300
Purdue University

November 10, 2017

Kaggle Competition Update (extra credit)

Students with > 0.6 accuracy in **Public Leaderboard***

21 students so far

*Extra credit based on ***Private* Leaderboard**

Public Leaderboard | **Private Leaderboard**

This leaderboard is calculated with approximately 30% of the test data.
The final results will be based on the other 70%, so the final standings may be different.

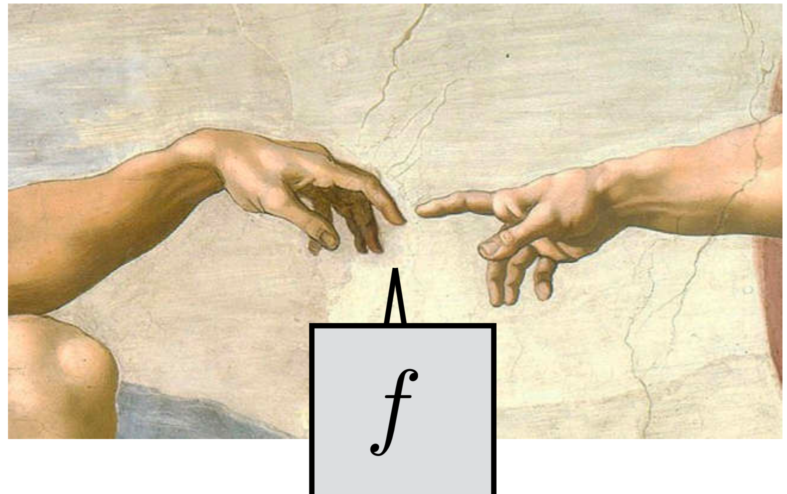
Raw Data | Refresh

#	△1w	Team Name	Kernel	Team Members	Score	Entries	Last
1	—	**General Grievous**			0.87980	6	8d
2	▲2	**Luke Skywalker**			0.85916	15	21h
3	▼1	**Captain Rex**			0.83447	4	8d
4	▼1	**Revan**			0.83407	12	13d
5	▲3	**Yoda**			0.82921	23	8h
6	▼1	**Cad Bane**			0.81991	18	17d
7	▲9	**Count Dooku**			0.81667	5	9h
8	▼2	**Dengar**			0.81222	4	6d
9	▼2	**Darth Vader**			0.81019	13	5h
10	▼1	**Bossk**			0.80291	7	5d
11	▲8	**Mace Windu**			0.77498	3	4d
12	new	**Anakin Solo**			0.76851	1	5d
13	▼3	**Ki-Adi-Mundi**			0.76811	2	18d
14	▼3	**Kyp Durron**			0.73613	4	1mo
15	▼3	**Shaak Ti**			0.70133	2	1mo
16	▼3	**Admiral Thrawn**			0.65520	2	18d
17	new	**Clone Commander Cody**			0.63901	8	3d

Neural Networks - Generative Models

Restricted Boltzmann Machines

Generative Task

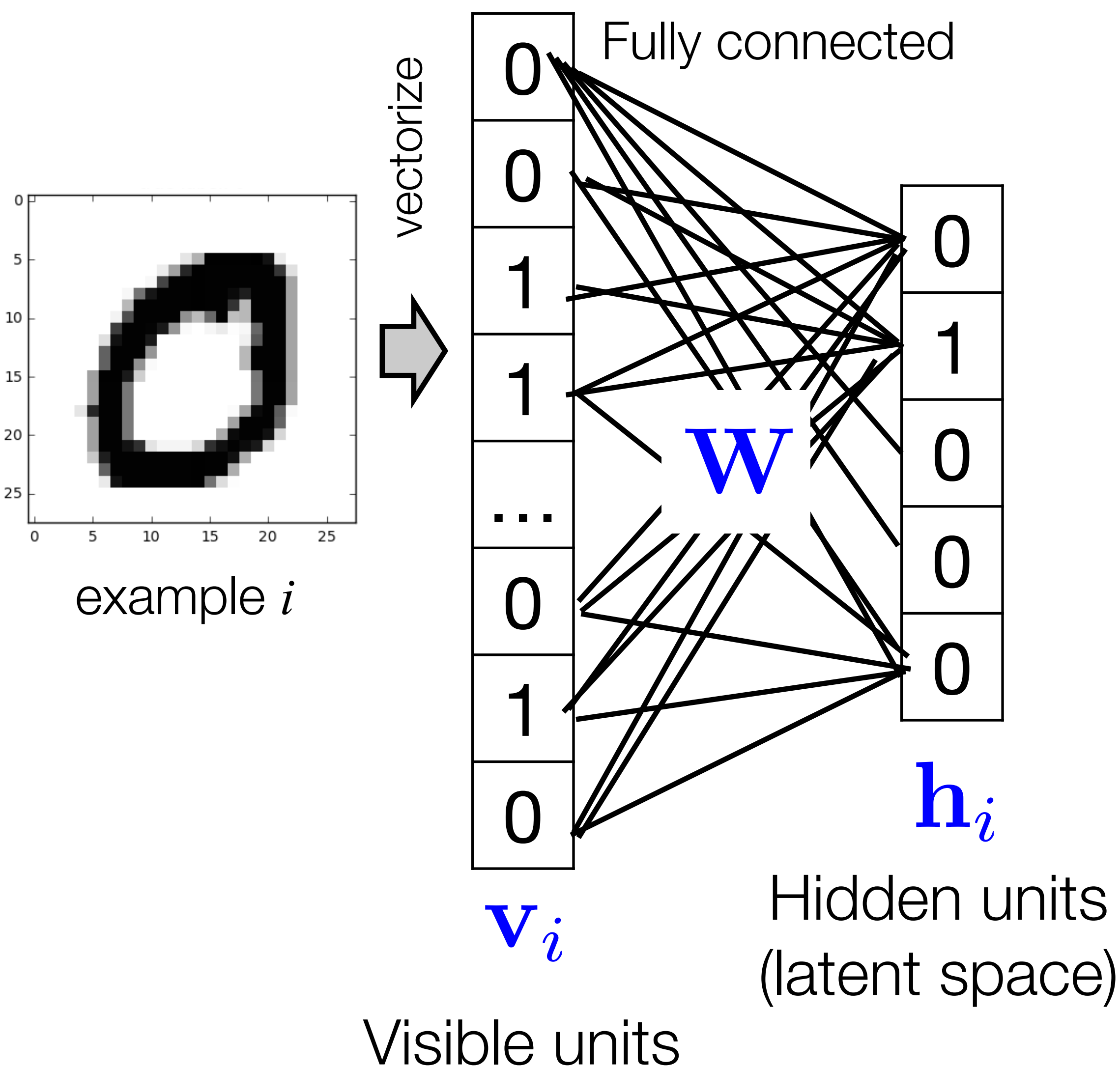


by fiat
(out of the blue)

Learns to generate examples: (x, y)

$$\left(\begin{array}{c} \text{is tanned} \\ \text{employs a captain} \\ \text{pays high property taxes} \\ \text{employs a cook} \end{array} , \text{ is rich} \right)$$

Restricted Boltzmann Machines

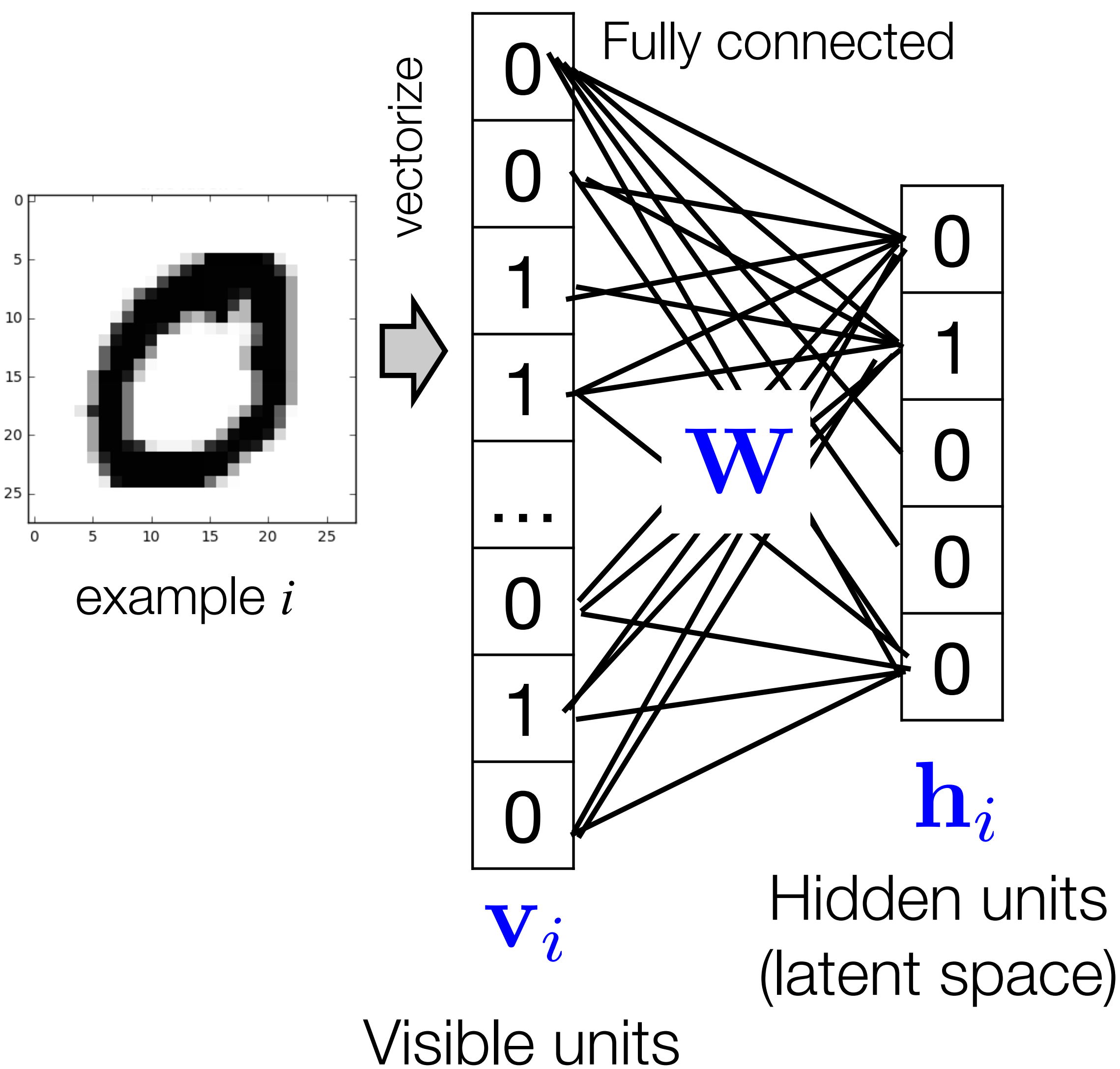


- Need to learn a good joint probability $p(\mathbf{v}_i, \mathbf{h}_i)$
- We will define the joint probability as

$$p(\mathbf{v}_i, \mathbf{h}_i; \mathbf{W}) = \frac{\exp\left(\mathbf{v}_i \mathbf{W} \mathbf{h}_i\right)}{\sum_{\forall \mathbf{v}, \mathbf{h}} \exp\left(\mathbf{v} \mathbf{W} \mathbf{h}\right)}$$

- What is the model space?
  - i.e., what are we searching over?
- What is the score function?
- How can we do the search?

Restricted Boltzmann Machines



- Model space:
  The set of all possible joint probability distributions given by all possible weights $\mathbf{W}$

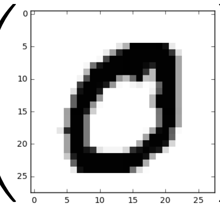
$$p(\mathbf{v}_i, \mathbf{h}_i; \mathbf{W}) = \frac{\exp\left(\mathbf{v}_i \mathbf{W} \mathbf{h}_i\right)}{\sum_{\forall \mathbf{v}, \mathbf{h}} \exp\left(\mathbf{v} \mathbf{W} \mathbf{h}\right)}$$

- What is the score function?
- How can we do the search?

What is the score function

Score Function of Restricted Boltzmann Machines (RBM)

- Training data $\{\mathbf{v}_i\}_{i=1}^N$

$\mathbf{v}_i = \mathrm{vec}\Big($  $\Big)$

column vectors

- Probability distribution of RBM (visible units **v'** , hidden units **h'**):

$$p(\mathbf{v}', \mathbf{h}'; \mathbf{W}) = \frac{\exp\left(\mathbf{v}'^T \mathbf{W} \mathbf{h}'\right)}{Z}, \qquad \text{where } Z = \sum_{\forall \mathbf{v}, \mathbf{h}} \exp\left(\mathbf{v}^T \mathbf{W} \mathbf{h}\right)$$

- Score function is the likelihood over training data

$$\prod_{i=1}^{N} p(\mathbf{v}_i; \mathbf{W}) = \prod_{i=1}^{N} \sum_{\mathbf{h}} p(\mathbf{v}_i, \mathbf{h}; \mathbf{W}) = \prod_{i=1}^{N} \frac{\sum_{\mathbf{h}} \exp\left(\mathbf{v}_i^T \mathbf{W} \mathbf{h}\right)}{\sum_{\forall \mathbf{v}, \mathbf{h}} \exp\left(\mathbf{v}^T \mathbf{W} \mathbf{h}\right)}$$

- How do we search?
    - Maximize the likelihood: $\mathbf{W}^\star = \arg\max_{\mathbf{W}} \prod_{i=1}^{N} p(\mathbf{v}_i; \mathbf{W})$

Searching for a good RBM model $\mathbf{W}$

Searching for Good Restricted Boltzmann Machine Models

- Score function is the likelihood over training data

$$\prod_{i=1}^{N} p(\mathbf{v}_i; \mathbf{W}) = \prod_{i=1}^{N} \sum_{\mathbf{h}} p(\mathbf{v}_i, \mathbf{h}; \mathbf{W}) = \prod_{i=1}^{N} \frac{\sum_{\mathbf{h}'} \exp\left(\mathbf{v}_i \mathbf{W} \mathbf{h}'\right)}{\sum_{\forall \mathbf{v}, \mathbf{h}} \exp\left(\mathbf{v} \mathbf{W} \mathbf{h}\right)}$$

- Maximize the likelihood

$$\mathbf{W}^{\star} = \arg\max_{\mathbf{W}} \log \prod_{i=1}^{N} p(\mathbf{v}_i; \mathbf{W}) = \arg\max_{\mathbf{W}} \sum_{i=1}^{N} \log p(\mathbf{v}_i; \mathbf{W})$$

- Gradient ascent to find W^{*}:

$$\frac{\partial \log p(\mathbf{v}_i; \mathbf{W})}{\partial \mathbf{W}} = \frac{\partial}{\partial \mathbf{W}} \log \frac{\sum_{\mathbf{h}'} \exp\left(\mathbf{v}_i^T \mathbf{W} \mathbf{h}'\right)}{\sum_{\forall \mathbf{v}, \mathbf{h}} \exp\left(\mathbf{v}^T \mathbf{W} \mathbf{h}\right)}$$

$$= \frac{\partial}{\partial \mathbf{W}} \log \sum_{\mathbf{h}'} \exp\left(\mathbf{v}_i^T \mathbf{W} \mathbf{h}'\right) - \frac{\partial}{\partial \mathbf{W}} \log \sum_{\forall \mathbf{v}, \mathbf{h}} \exp\left(\mathbf{v}^T \mathbf{W} \mathbf{h}\right)$$

$$= \frac{\sum_{\mathbf{h}'} \frac{\partial}{\partial \mathbf{W}} \exp\left(\mathbf{v}_i^T \mathbf{W} \mathbf{h}'\right)}{\sum_{\mathbf{h}} \exp\left(\mathbf{v}_i^T \mathbf{W} \mathbf{h}\right)} - \frac{\sum_{\forall \mathbf{v}', \mathbf{h}'} \frac{\partial}{\partial \mathbf{W}} \exp\left(\mathbf{v}'^T \mathbf{W} \mathbf{h}'\right)}{\sum_{\forall \mathbf{v}, \mathbf{h}} \exp\left(\mathbf{v}^T \mathbf{W} \mathbf{h}\right)}$$

Computing the log-likelihood derivative (cont)

$$\frac{\partial \log p(\mathbf{v}_i; \mathbf{W})}{\partial \mathbf{W}} =$$

$$= \frac{\sum_{\mathbf{h}'} \frac{\partial}{\partial \mathbf{W}} \exp\left(\mathbf{v}_i^T \mathbf{W} \mathbf{h}'\right)}{\sum_{\mathbf{h}} \exp\left(\mathbf{v}_i^T \mathbf{W} \mathbf{h}\right)} - \frac{\sum_{\forall \mathbf{v}', \mathbf{h}'} \frac{\partial}{\partial \mathbf{W}} \exp\left({\mathbf{v}'}^T \mathbf{W} \mathbf{h}'\right)}{\sum_{\forall \mathbf{v}, \mathbf{h}} \exp\left(\mathbf{v}^T \mathbf{W} \mathbf{h}\right)}$$

$$= \sum_{\mathbf{h}'} {\mathbf{v}_i}^T \mathbf{h}' \frac{\exp\left(\mathbf{v}_i^T \mathbf{W} \mathbf{h}'\right)}{\sum_{\mathbf{h}} \exp\left(\mathbf{v}_i^T \mathbf{W} \mathbf{h}\right)} - \sum_{\forall \mathbf{v}', \mathbf{h}'} {\mathbf{v}'}^T \mathbf{h}' \frac{\exp\left({\mathbf{v}'}^T \mathbf{W} \mathbf{h}'\right)}{\sum_{\forall \mathbf{v}, \mathbf{h}} \exp\left(\mathbf{v}^T \mathbf{W} \mathbf{h}\right)}$$

Probability of h' given v_i

Somewhat easy to compute exactly
(there are some mathematical tricks)

Joint probability of v' and h'

Very hard to compute exactly
(no trick)

Hard to Compute Gradient?

$$\frac{\partial \log p(\mathbf{v}_i; \mathbf{W})}{\partial \mathbf{W}} =$$

$$= \sum_{\mathbf{h}'} {\mathbf{v}_i}^T \mathbf{h}' \frac{\exp\left(\mathbf{v}_i^T \mathbf{W} \mathbf{h}'\right)}{\sum_{\mathbf{h}} \exp\left(\mathbf{v}_i^T \mathbf{W} \mathbf{h}\right)} - \sum_{\forall \mathbf{v}', \mathbf{h}'} {\mathbf{v}'}^T \mathbf{h}' \frac{\exp\left({\mathbf{v}'}^T \mathbf{W} \mathbf{h}'\right)}{\sum_{\forall \mathbf{v}, \mathbf{h}} \exp\left(\mathbf{v}^T \mathbf{W} \mathbf{h}\right)}$$

(CONDITIONAL DATA AVERAGE)
Somewhat easy to compute exactly
(there are some mathematical tricks)

(TOTAL AVERAGE)
Very hard to compute exactly
(no trick)

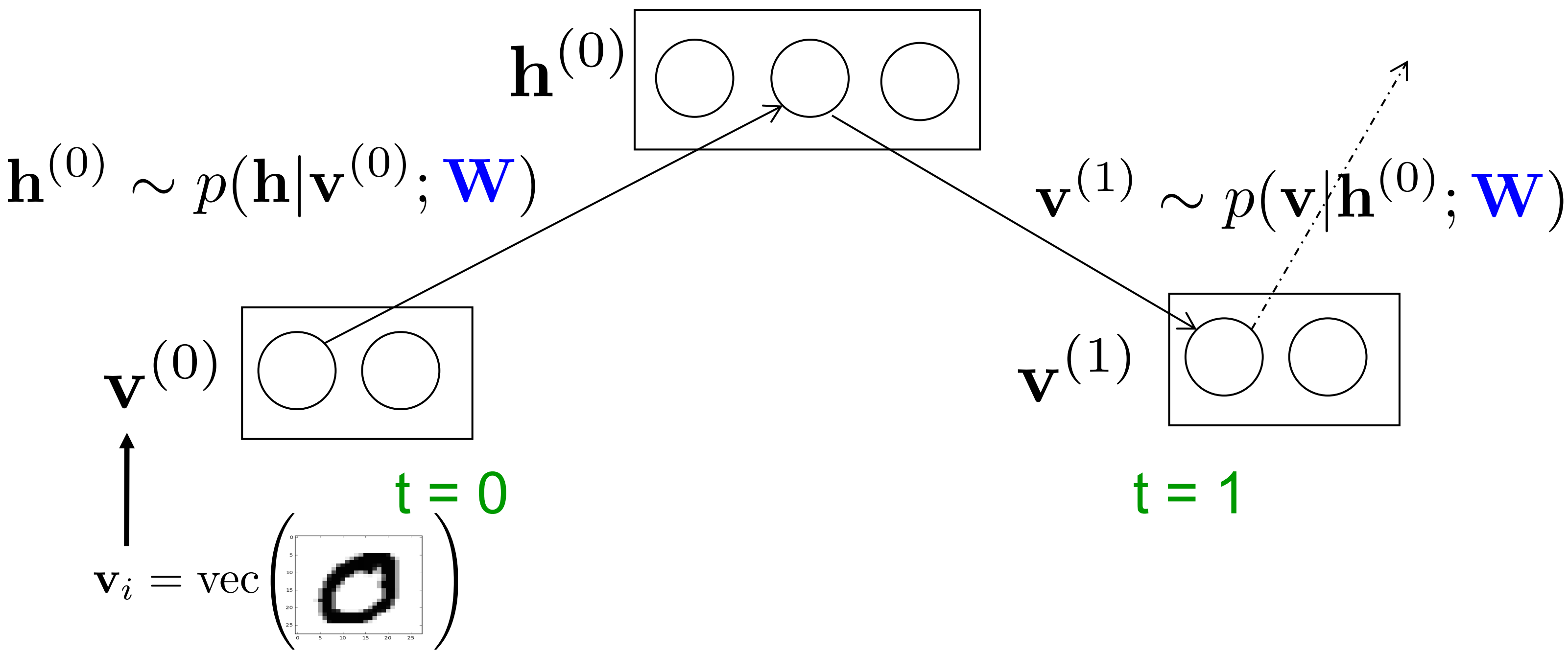
- Rather than computing the right-hand side exactly, we will approximate it
  - The RHS is just an average: $$E[{\mathbf{v}'}^T \mathbf{h}'] = \sum_{\forall \mathbf{v}', \mathbf{h}'} {\mathbf{v}'}^T \mathbf{h}' \frac{\exp\left({\mathbf{v}'}^T \mathbf{W} \mathbf{h}'\right)}{\sum_{\forall \mathbf{v}, \mathbf{h}} \exp\left(\mathbf{v}^T \mathbf{W} \mathbf{h}\right)}$$
  - We will compute this average using a Markov Chain Monte Carlo method called *Contrastive Divergence*
    - We will see how it works, not why it works

Estimating the average $E[\mathbf{v}'^T\mathbf{h}']$

Note that we need to do this every time we want to compute the gradient of the log-likelihood function (e.g., at every gradient step)

Estimating Model Average: Contrastive Divergence

- Follow this procedure to estimate $E[\mathbf{v}'^T \mathbf{h}']$

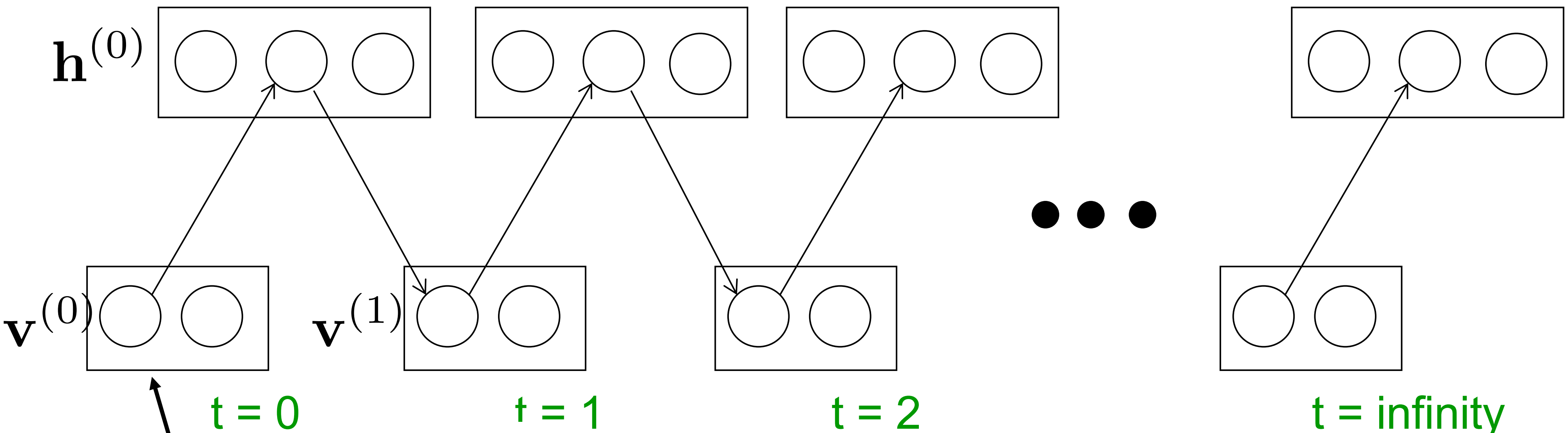


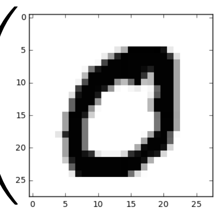
Algorithm

1. Start with a training example as $v^{(0)}$, set t = 0
2. At step t, sample $\mathbf{h}^{(t)}$ from the conditional distribution $p(\mathbf{h}|\mathbf{v}^{(t)}; \mathbf{W})$
3. t = t +1
4. Sample $\mathbf{v}^{(t)}$ from the conditional distribution $p(\mathbf{v}|\mathbf{h}^{(t)}; \mathbf{W})$
5. Repeat 2 # note the infinite loop

Contrastive Divergence

- This is a way to compute $E[\mathbf{v}'^T \mathbf{h}']$



$\mathbf{v}_i = \mathrm{vec}\Big($  $\Big)$

After the infinite loop is “over”, when the universe ends, output $(\mathbf{v}^{(\infty)})^T \mathbf{h}^{(\infty)}$ as our estimate of $E[\mathbf{v}'^T \mathbf{h}']$

In practice we will do just **K** steps… and hope for the best

- **K = 1** works surprisingly well

RMBs in real life

- 784 pixels: 28x28 digit image
- Only 32 hidden neurons
- Top data examples
- Bottom are generated examples

data

generated