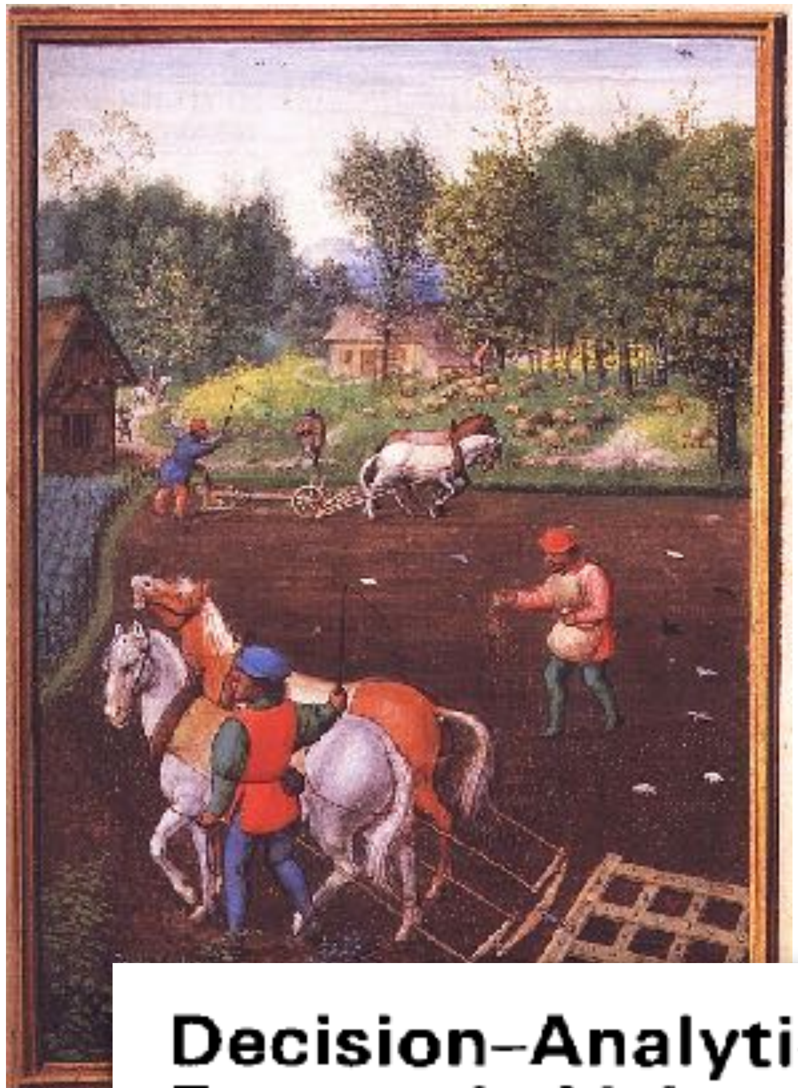
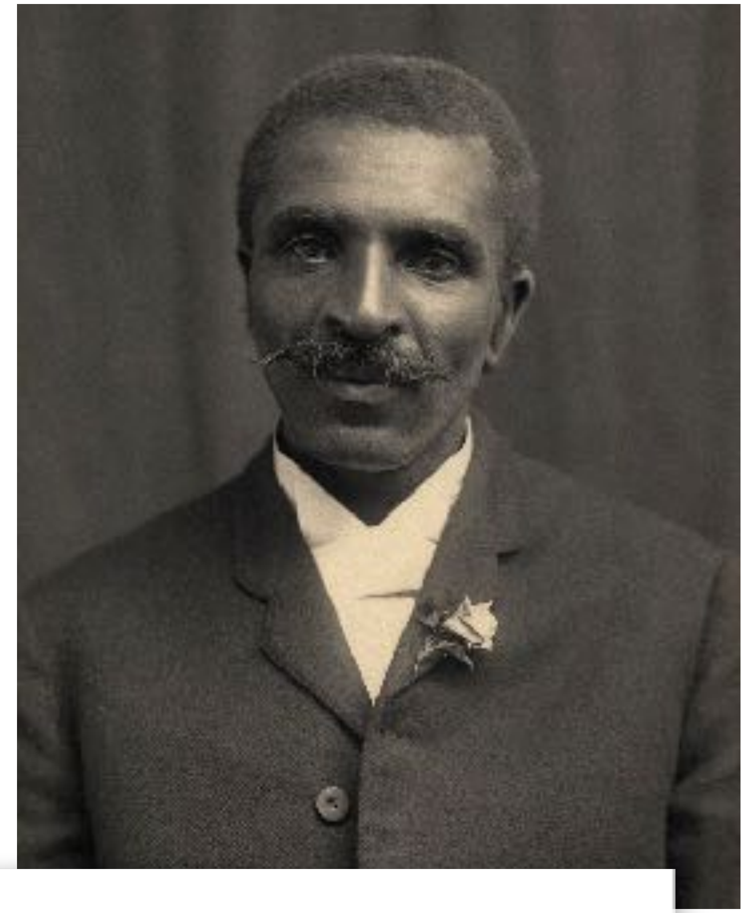
Recurrent Neural Networks

Prof. Tamara Broderick

Edited From 6.036 Fall21 Offering





Decision–Analytic Assessment of the Economic Value of Weather Forecasts: The Fallowing/Planting Problem

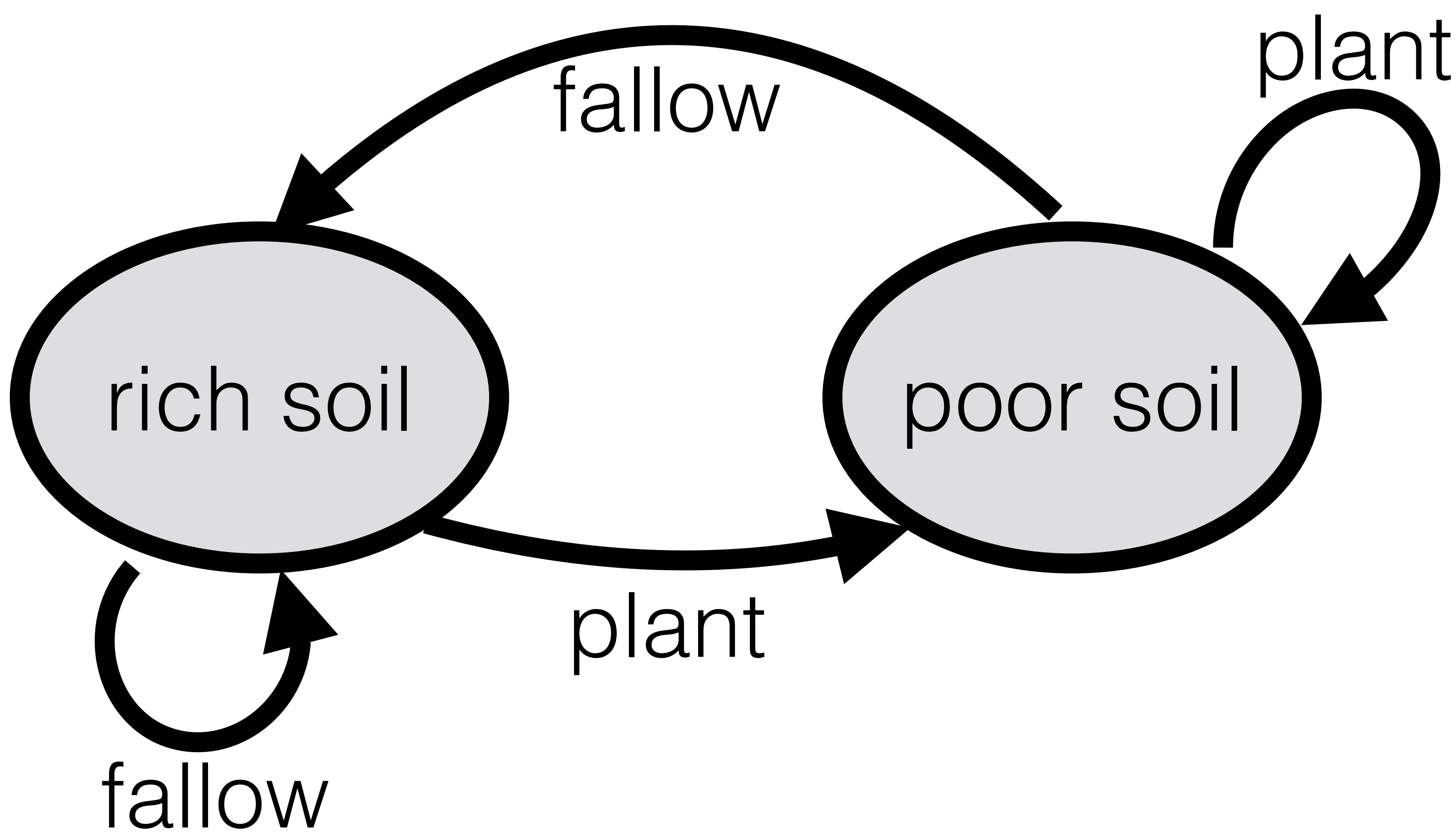
RICHARD W. KATZ
National Center for Atmospheric Research, U.S.A.

and

BARBARA G. BROWN* and ALLAN H. MURPHY
Oregon State University, U.S.A.

State Machine

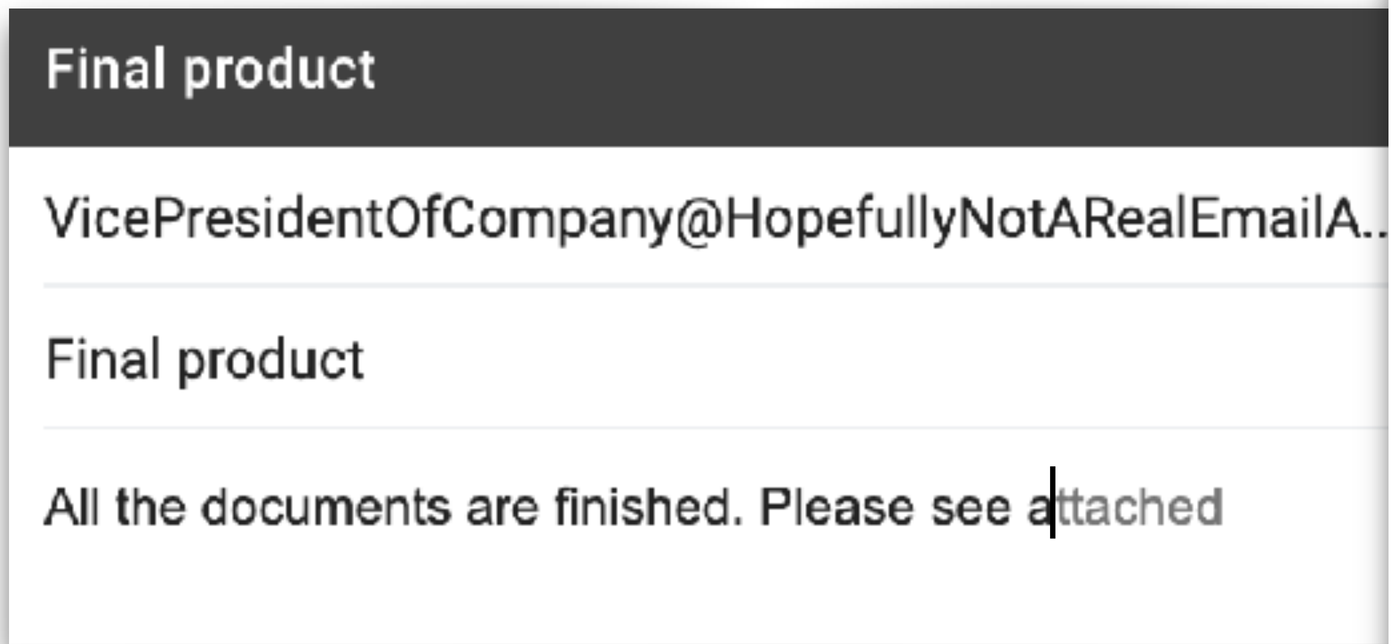
- $\mathcal{S}$ = set of possible states
- $\mathcal{X}$ = set of possible inputs
- $s_0 \in \mathcal{S}$: initial state
- $f : \mathcal{S} \times \mathcal{X} \to \mathcal{S}$: transition function
- $\mathcal{Y}$: set of possible outputs
- $g : \mathcal{S} \to \mathcal{Y}$: output function
  - e.g. g(s) = s
  - e.g. g(s) = soil-moisture-sensor(s)

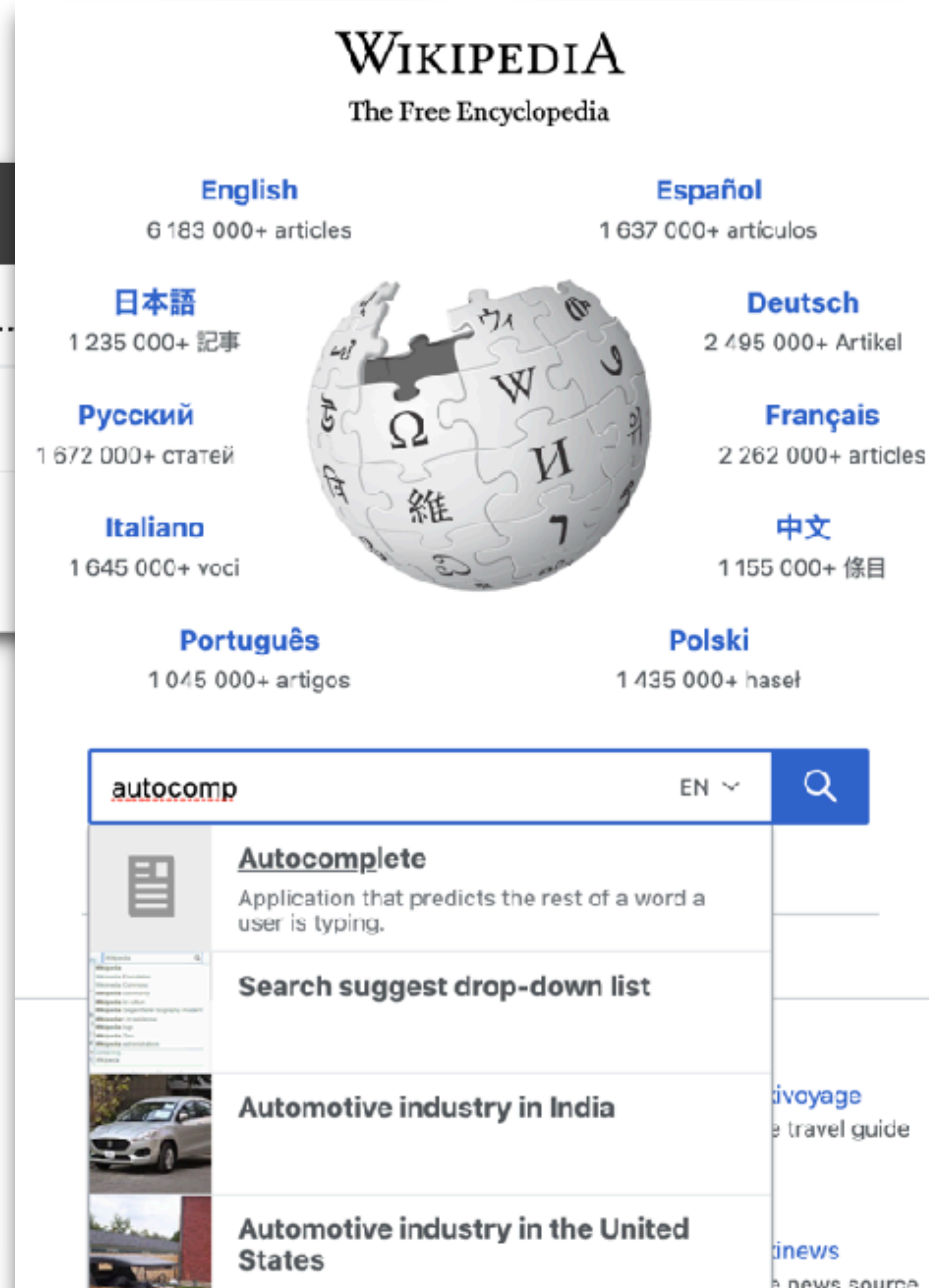


Example

$$s_0 = \text{rich}$$
$$s_1 = f(s_0, \text{plant}) = \text{poor};$$
$$y_1 = g(s_1) = \text{poor}$$
$$s_2 = f(s_1, \text{fallow}) = \text{rich};$$
$$y_2 = g(s_2) = \text{rich}$$

Text prediction





[https://commons.wikimedia.org/wiki/File:Stephen_Hawking.StarChild.jpg, www.itv.com/news/central/story/2014-08-05/locked-in-syndrome-woman-earns-degree-by-blinking]

Text prediction: supervised learning

- Training data: lots of text
  - “what happens to a dream deferred”

features	label
w	h
wh	a
wha	t
what	_
what_	h
what_h	a
what_ha	p
what_hap	p
what_happ	e

- Classification with 27 classes
- How to featurize?
- Idea: use all previous characters. But so far we've said $x^{(i)} \in \mathbb{R}^d$; i.e. fixed dimension
- Idea: just use last character. But lose info
- Idea: use last *m* characters

Text prediction: supervised learning

- Training data: lots of text
  - “what happens to a dream deferred”

features	label
w	h
wh	a
wha	t
what	_
what_	h
what_h	a
what_ha	p
what_hap	p
what_happ	e

- Classification with 27 classes
- How to featurize?
- Idea: use all previous characters. But so far we’ve said $x^{(i)} \in \mathbb{R}^d$; i.e. fixed dimension
- Idea: just use last character. But lose info
- Idea: use last $m = 3$ characters

A state machine: writing & predicting text

- Recall state machines:
  - Set of possible states $\mathcal{S}$
  - Set of possible inputs $\mathcal{X}$
  - Initial state
  - Transition function $f(s,x)$
  - Set of possible outputs
  - Output function $p = g(s)$

previously "y"

- Example:
  - Every sequence of 3 chars
  - Every single character
  - s_0 = ^^^
  - E.g. f(^^w,h) = ^wh
  - Output: 27 prediction probs
  - E.g. p=g(^^^) = [0.08, 0.02, …., 0.001, 0.01]

t	$x_t^{(1)}$	$s_t^{(1)}$
0		^^^
1	w	^^w
2	h	^wh
3	a	wha
4	t	hat
5	_	at_

- $x^{(1)}$: "what happens to a dream deferred"
- $x^{(2)}$: "if you can keep your head when all about you"
- $x^{(3)}$: "you may write me down in history"

A state machine: writing & predicting text

- Recall state machines:
  - Set of possible states $\mathcal{S}$
  - Set of possible inputs $\mathcal{X}$
  - Initial state
  - Transition function $f(s,x)$
  - Set of possible outputs
  - Output function $p = g(s)$
- Example:
  - Every sequence of 3 chars
  - Every single character
  - s_0
  - E.g. $f\left(\begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}, x\right) = \begin{bmatrix} c_2 \\ c_3 \\ x \end{bmatrix}$
  - Output: v prediction probs
  - E.g. $p=g\left(\begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}\right)$
- Example with alphabet {0,1} (v=2); state is last m = 3 chars

$$s_t = f_1\left(\begin{bmatrix} w_1^{sx} \\ w_2^{sx} \\ w_3^{sx} \end{bmatrix} x_t + \begin{bmatrix} w_{11}^{ss} & w_{12}^{ss} & w_{13}^{ss} \\ w_{21}^{ss} & w_{22}^{ss} & w_{23}^{ss} \\ w_{31}^{ss} & w_{32}^{ss} & w_{33}^{ss} \end{bmatrix} s_{t-1} + \begin{bmatrix} w_{0,1}^{ss} \\ w_{0,2}^{ss} \\ w_{0,3}^{ss} \end{bmatrix}\right)$$

$$\begin{aligned} p_t &= g(s_t) \\ &= f_2(W^o s_t + W_0^o) \end{aligned}$$

same as final layer in all of our neural networks so far, except for notation

A state machine: writing & predicting text

- Recall state machines:
  - Set of possible states $\mathcal{S}$
  - Set of possible in... ts $\mathcal{X}$
  - Initial state
  - Transition f...
  - Set of p... ts
  - Outp... $= g(s)$

componentwise; activation functions

- Example:
  - Every sequence of 3 chars
  - Every single character
  - s_0
  - E.g. $f\left(\begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}, x\right) = \begin{bmatrix} c_2 \\ c_3 \\ x \end{bmatrix}$
  - Output: v prediction probs
  - E.g. $p = g\left(\begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}\right)$

- Example ... alphabet {0,1} (v=2); state is last m = 3 chars

$$s_t = f_1\left(\begin{bmatrix} w_1^{sx} \\ w_2^{sx} \\ w_3^{sx} \end{bmatrix} x_t + \begin{bmatrix} w_{11}^{ss} & w_{12}^{ss} & w_{13}^{ss} \\ w_{21}^{ss} & w_{22}^{ss} & w_{23}^{ss} \\ w_{31}^{ss} & w_{32}^{ss} & w_{33}^{ss} \end{bmatrix} s_{t-1} + \begin{bmatrix} w_{0,1}^{ss} \\ w_{0,2}^{ss} \\ w_{0,3}^{ss} \end{bmatrix} \right)$$

$$\begin{aligned} p_t &= g(s_t) \\ &= f_2(W^o s_t + W_0^o) \end{aligned}$$

same as final layer in all of our neural networks so far, except for notation

A state machine: writing & predicting text

- Recall state machines:
  - Set of possible states $\mathcal{S}$
  - Set of possible inputs $\mathcal{X}$
  - Initial state
  - Transition function $f(s,x)$
  - Set of possible outputs
  - Output function $p = g(s)$
- Example:
  - Every sequence of 3 chars
  - Every single character
  - s_0
  - E.g. $f\left(\begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}, x\right) = \begin{bmatrix} c_2 \\ c_3 \\ x \end{bmatrix}$
  - Output: v prediction probs
  - E.g. $p = g\left(\begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}\right)$
- Example with alphabet {0,1} (v=2); state is last m = 3 chars

$$s_t = f_1\left(W^{sx}x_t + W^{ss}s_{t-1} + W_0^{ss}\right)$$

$$\begin{aligned} p_t &= g(s_t) \\ &= f_2(W^o s_t + W_0^o) \end{aligned}$$

$\boxed{\text{same as final layer in all of our neural networks so far, except for notation}}$

A state machine: writing & predicting text

- Recall state machines:
  - Set of possible states $\mathcal{S}$
  - Set of po
  - Initial sta
  - Transition
  - Set of po
  - Output fu
- Example:
  - Every sequence of 3 chars
  - racter
  - $\begin{bmatrix} c_2 \\ c_3 \\ x \end{bmatrix}$
  - tion probs

Recall:

- In 2-layer, fully-connected NNs, we learned the features.
- In CNNs, we learned the filters.
- Analogous idea here.

- Example with alphabet {0, 1} (V=2), state is last m = 3 chars

$$s_t = f_1\left(W^{sx}x_t + W^{ss}s_{t-1} + W_0^{ss}\right)$$

$$\begin{aligned} p_t &= g(s_t) \\ &= f_2(W^o s_t + W_0^o) \end{aligned}$$

Put it all together:

$$p^{(i)} = \mathrm{RNN}(x^{(i)}; W, W_0)$$

"recurrent neural network"

Recurrent Neural Networks

Recall: familiar pattern

1. Choose how to predict label (given features & parameters)
2. Choose a loss (between guess & actual label)
3. Choose parameters by trying to minimize the training loss

In our example application:

1. Use an RNN: $p^{(i)} = \mathrm{RNN}(x^{(i)}; W, W_0)$

$$s_t = f_1\left(W^{sx} x_t + W^{ss} s_{t-1} + W_0^{ss}\right)$$

$$p_t = f_2(W^o s_t + W_0^o)$$

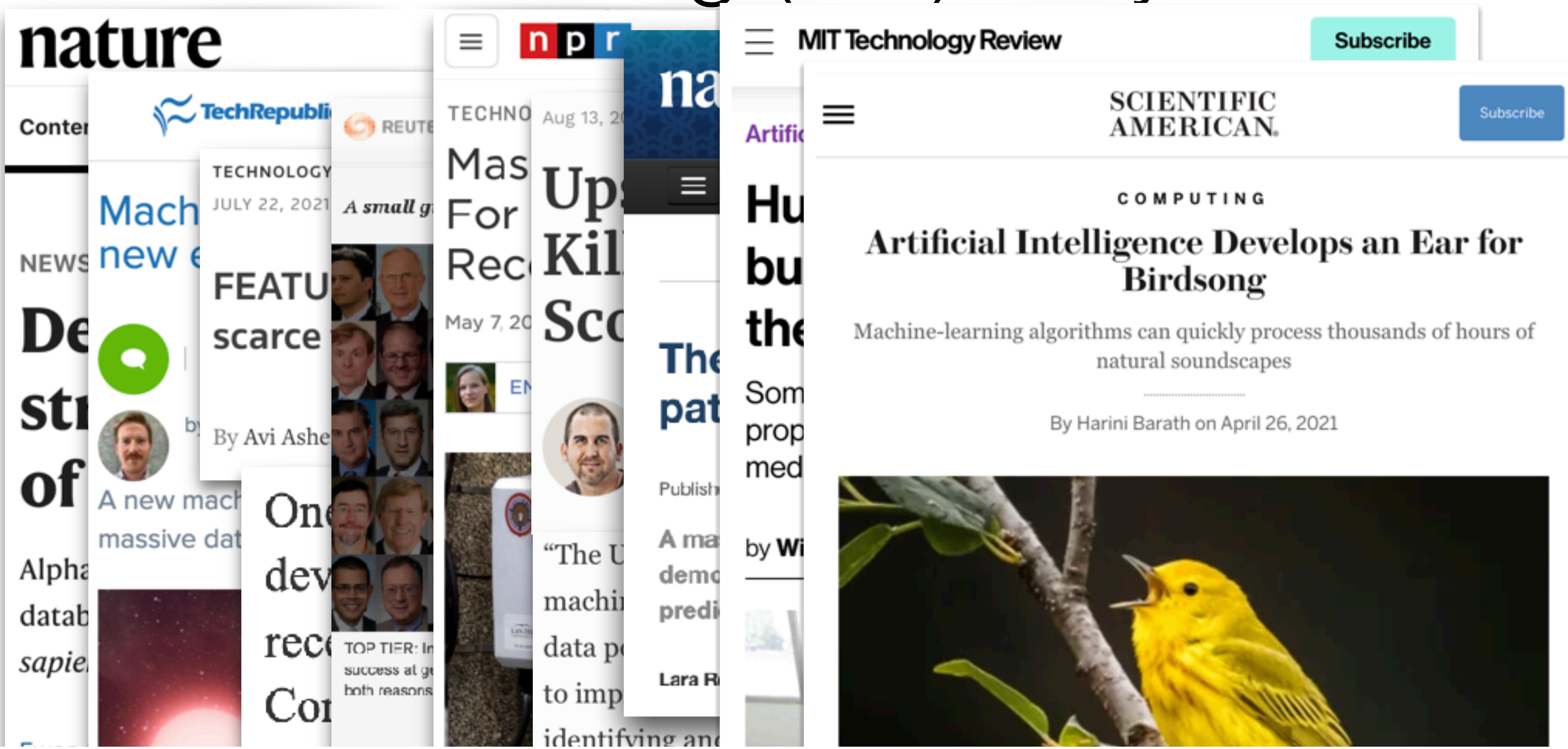
2. Loss: for *q* sequences, where the *i*th has length $n^{(i)}$

sequence $L_{\mathrm{seq}}(p^{(i)}, y^{(i)}) = \sum_{t=1}^{n^{(i)}} L_{\mathrm{elt}}(p_t^{(i)}, y_t^{(i)})$ element

$$J(W, W_0) = \sum_{i=1}^{q} L_{\mathrm{seq}}\left(p^{(i)}, y^{(i)}\right)$$

3. Stochastic gradient descent

Machine learning (ML): why & what



- **What is ML?** A set of methods for making decisions from data. (See the rest of the course!)
- **Why study ML?** To apply; to understand; to evaluate
- **Notes:** ML is a tool with pros & cons. ML is built on math

Thank you!