EECS 395/495: Statistical Language Models

Winter 2017

Introductions

- Professor: Doug Downey
- (part-time) Teaching Assistant: Nishant Subramani
- Course web site:
  - www.cs.northwestern.edu/~ddowney/courses/395_Winter2017
  - (linked off prof. home page)

Logistics

- Grading
  - Participation (50%)
    - Reading papers – two-paragraph summaries (10%)
    - Attending and participating in discussions (10%)
      - Self-reported
    - Presenting a paper (30%)
      - Teams of k
  - Self-directed Project (50%)
    - Proposal (5%)
    - Status update (5%)
    - Final presentation (20%) and report (20%)

Outline for Today

- Statistical Language Models (SLMs): What are they?
- Brief Neural Net (NN) primer
- NN SLMs
- Class structure
  - Class participation
  - Projects

Modeling Sequences of Words

- Statistical language models assign **probabilities** to **sequences of words**

$$P(\text{“the dog barked”}) = 4.203 * 10^{-9}$$

- Applications
  - Speech Recognition
  - Machine Translation
  - Spelling Correction
  - Information Extraction

How to measure LM performance?

- Test model M on held-out text $\overline{w}$ of length N

$$\text{Perplexity}(M, \overline{w}) = 2^{\frac{-\log_2 P_M(\overline{w})}{N}}$$

Models trained on 38 million words from the Wall Street Journal (WSJ) using a 19,979 word vocabulary.

Evaluate on a disjoint set of 1.5 million WSJ words

Perplexity: Trigram < Bigram < Unigram (in this case)

Which N-gram is the best in general?

	Unigram	Bigram	Trigram
Perplexity	962	170	109

http://www.cs.stonybrook.edu/~ychoi/cse628/lecture/02-ngram.pdf

Unigram

Months the my and issue of year foreign new exchange's september were recession exchange new endorsed a acquire to six executives

Bigram

Last December through the way to preserve the Hudson corporation N. B. E. C. Taylor would seem to complete the major central planners one point five percent of U. S. E. has already old M. X. corporation of living on information such as more frequently fishing to keep her

Trigram

They also point to ninety nine point six billion dollars from two hundred four oh six three percent of the rates of interest stores as Mexico and Brazil on market conditions

Outline for Today

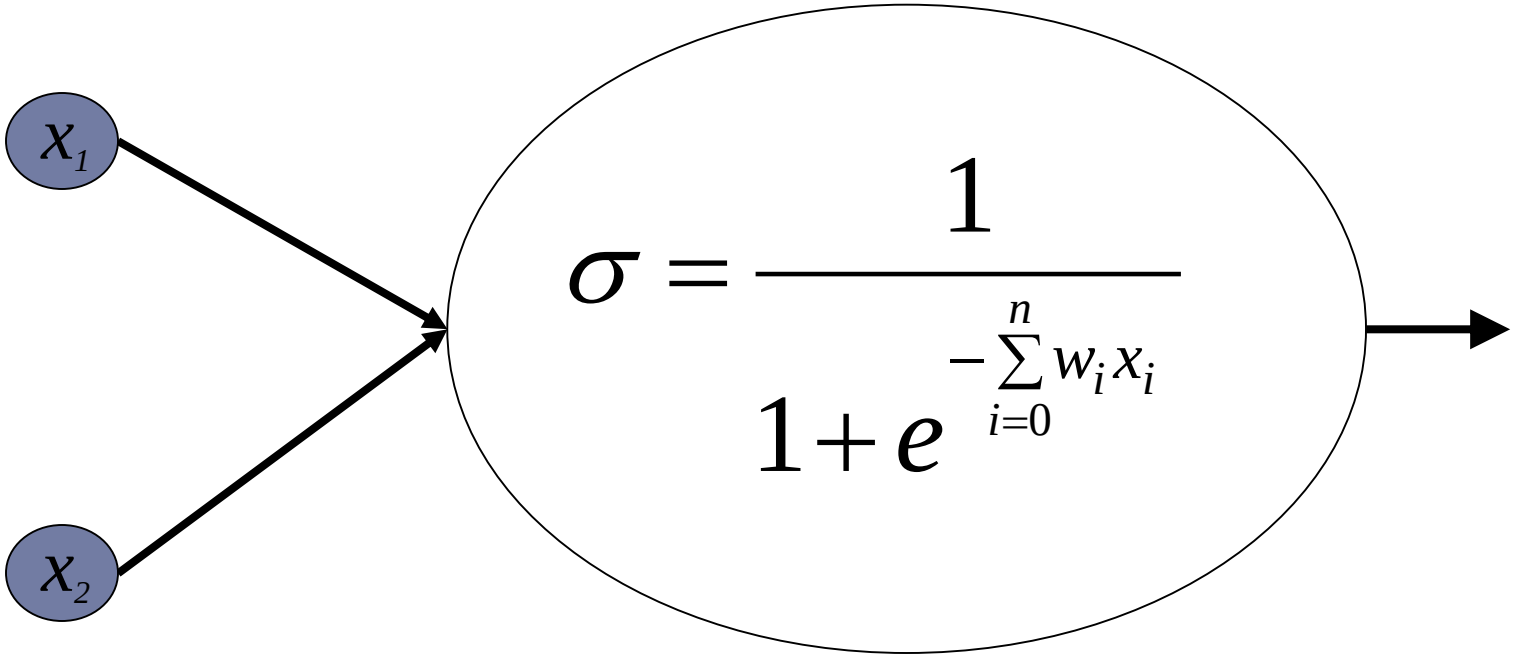
- Statistical Language Models (SLMs): What are they?
- **Brief Neural Net (NN) primer**
- NN SLMs
- Class structure
  - Class participation
  - Projects

Perceptrons

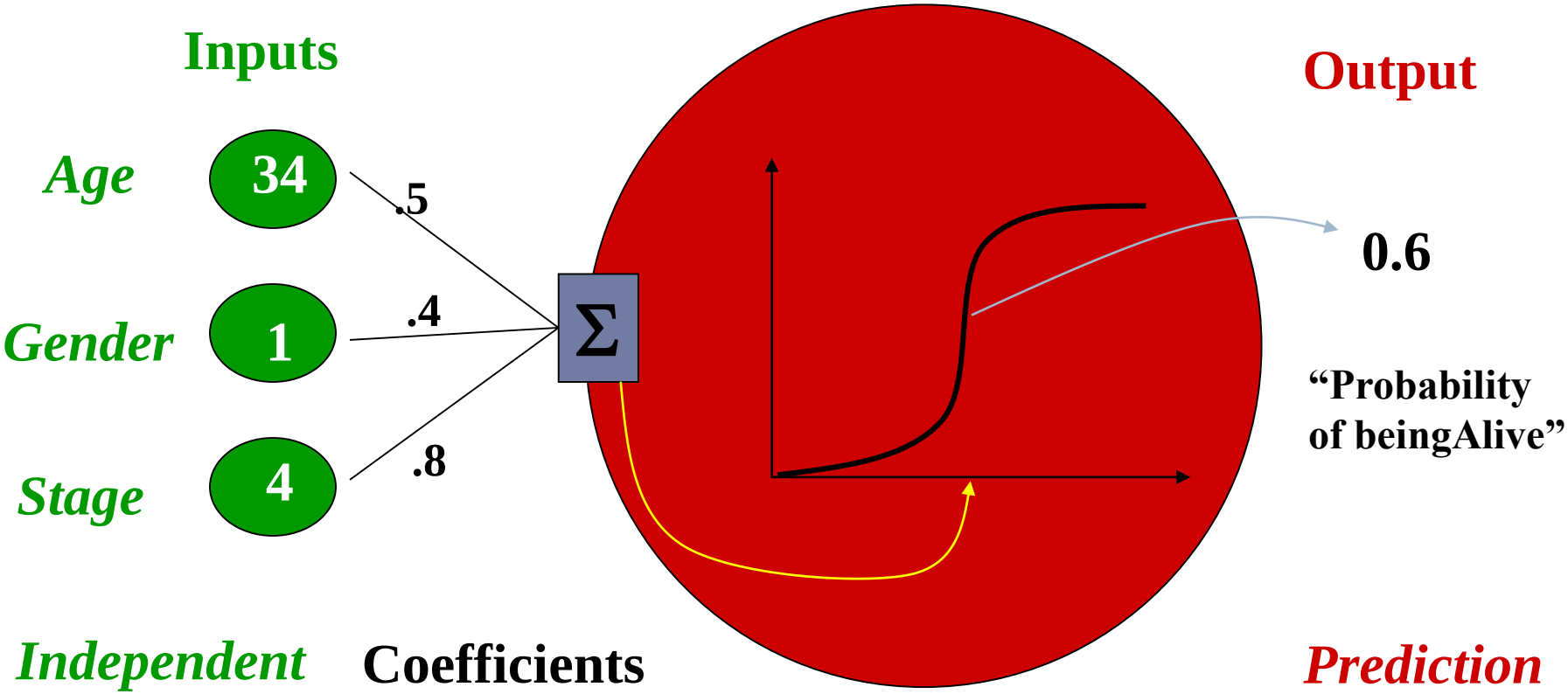
- Problem def:
  - Let f be a target function from
    $X = <x_1, x_2, \ldots>$ where $x_i \in \{0, 1\}$
    to
    $y \in \{0, 1\}$
  - Given training data $\{(X_1, y_1), (X_2, y_2) \ldots\}$
    - Learn $h(X)$, an approximation of $f(X)$

The sigmoid (logistic) unit

- Has differentiable function
  - Allows gradient descent
- Can be used to learn non-linear functions

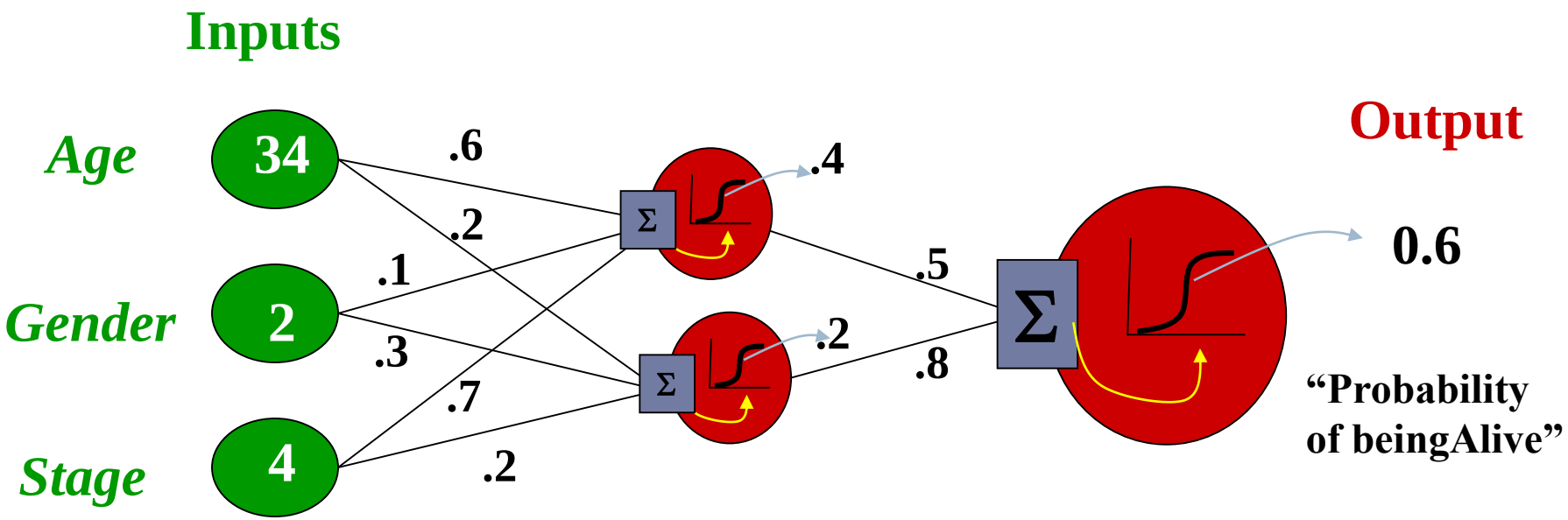


Logistic function



$$\sigma = \frac{1}{1+e^{-\sum_{i=0}^{n} w_i x_i}}$$

Neural Network Model



Independent variables

Weights

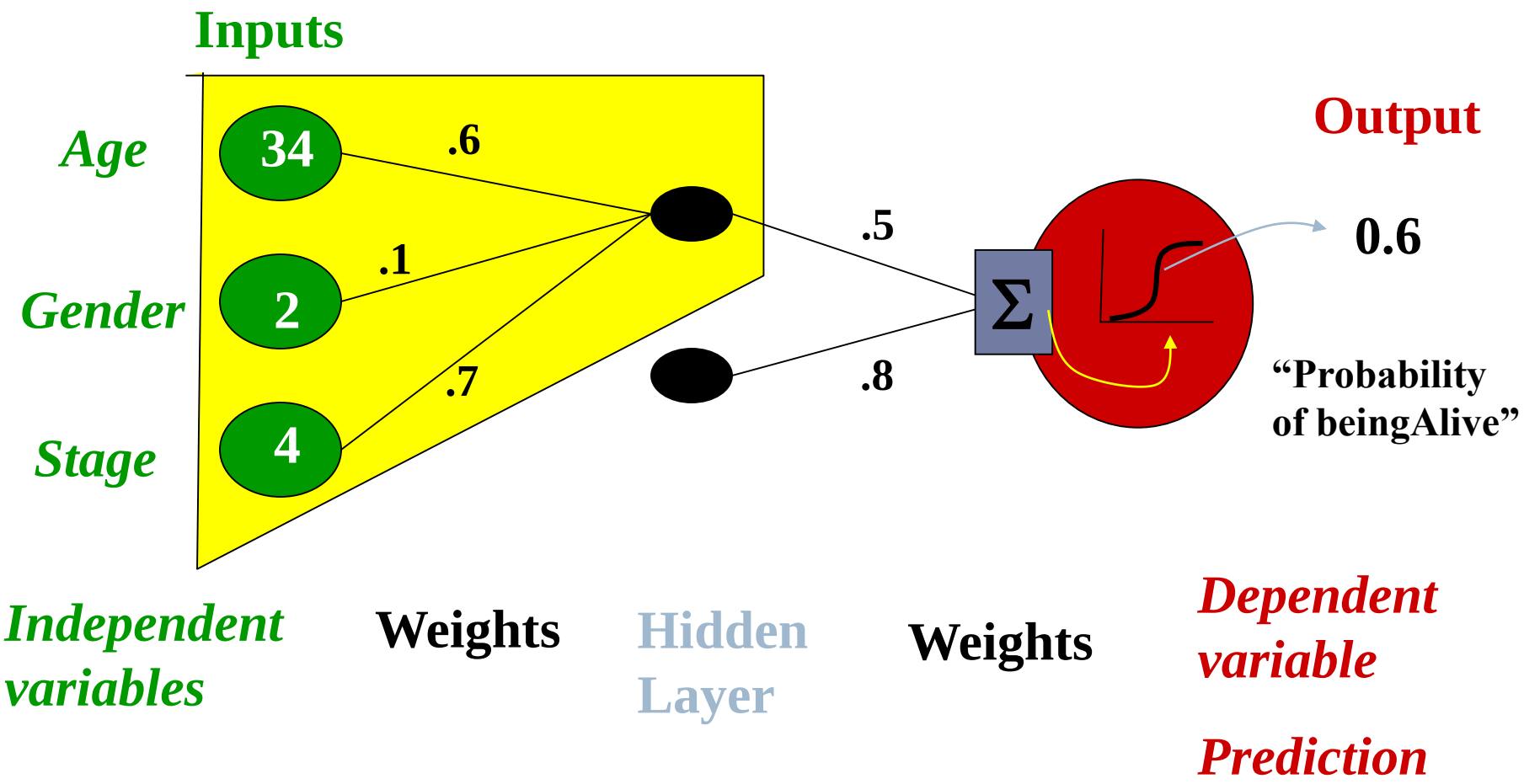
Hidden Layer

Weights

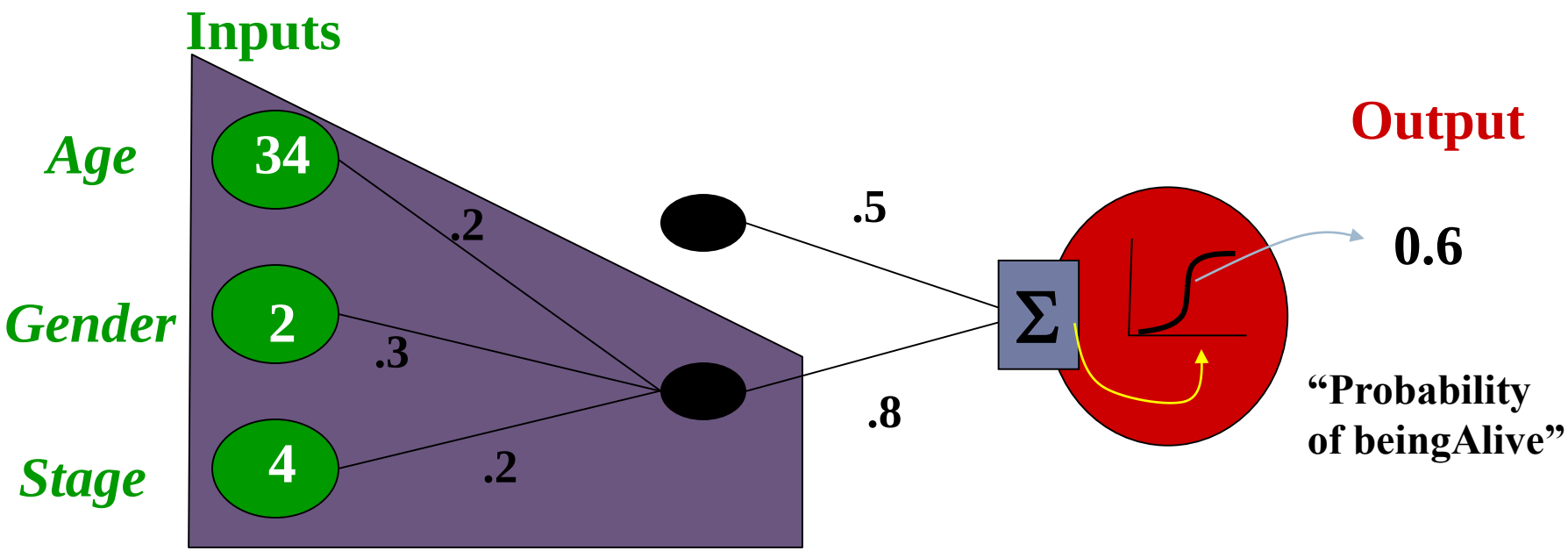
Dependent variable

Prediction

Getting an answer from a NN



Getting an answer from a NN



Independent variables

Weights

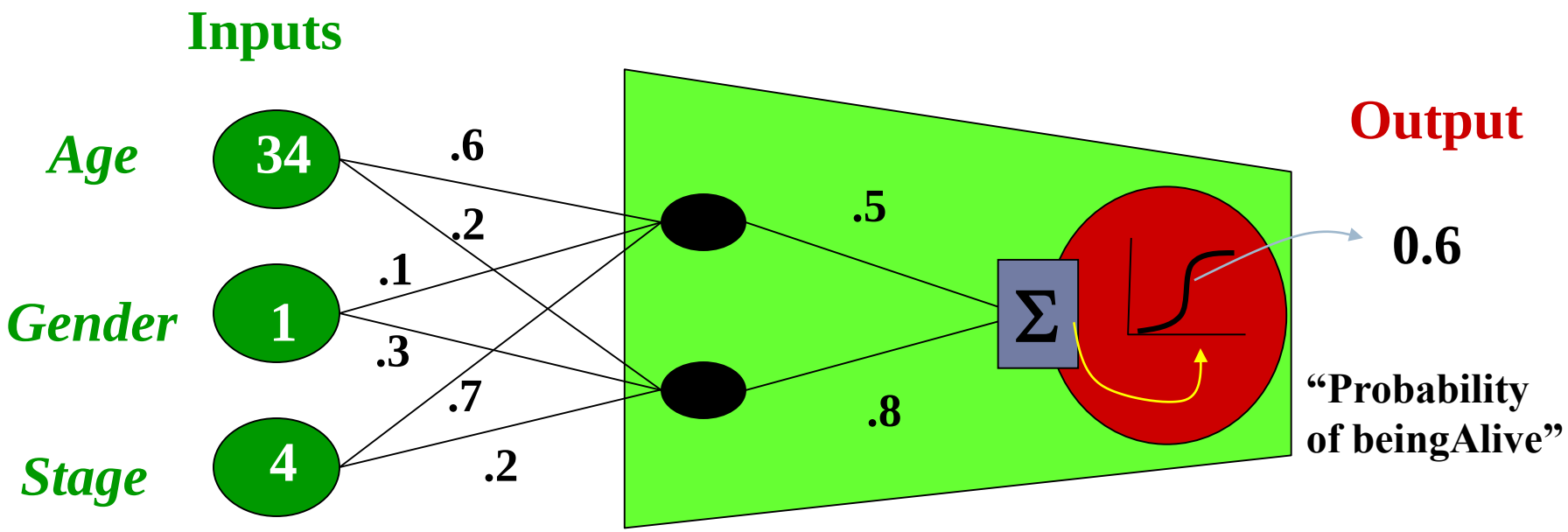
Hidden Layer

Weights

Dependent variable

Prediction

Getting an answer from a NN



Independent variables

Weights

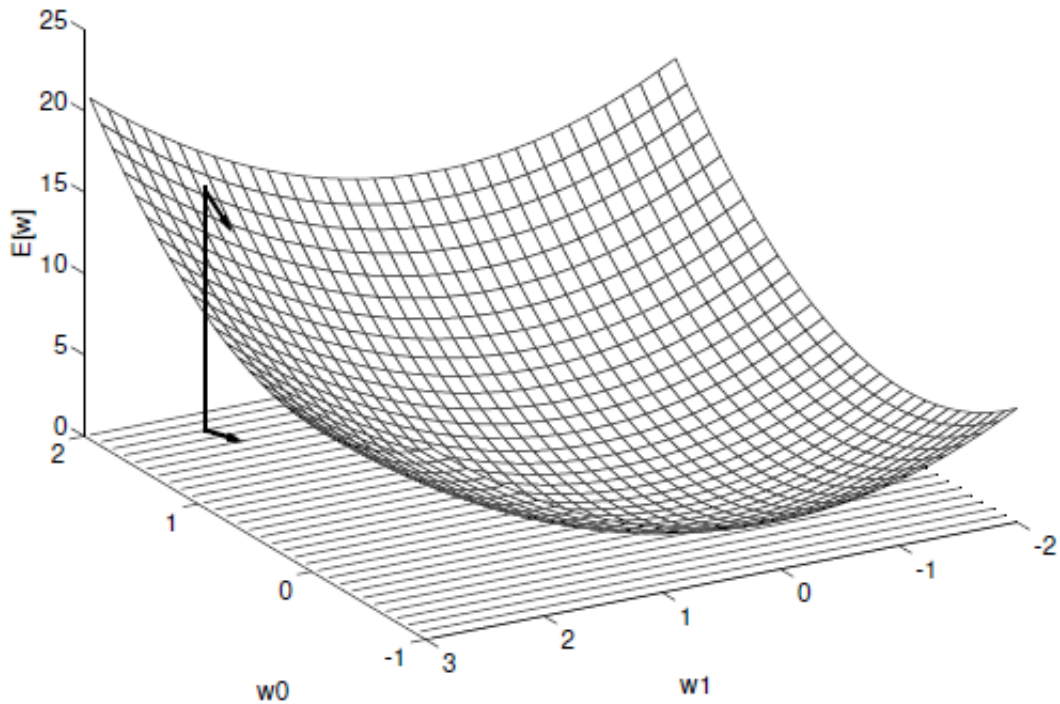
Hidden Layer

Weights

Dependent variable

Prediction

Gradient Descent



Gradient:

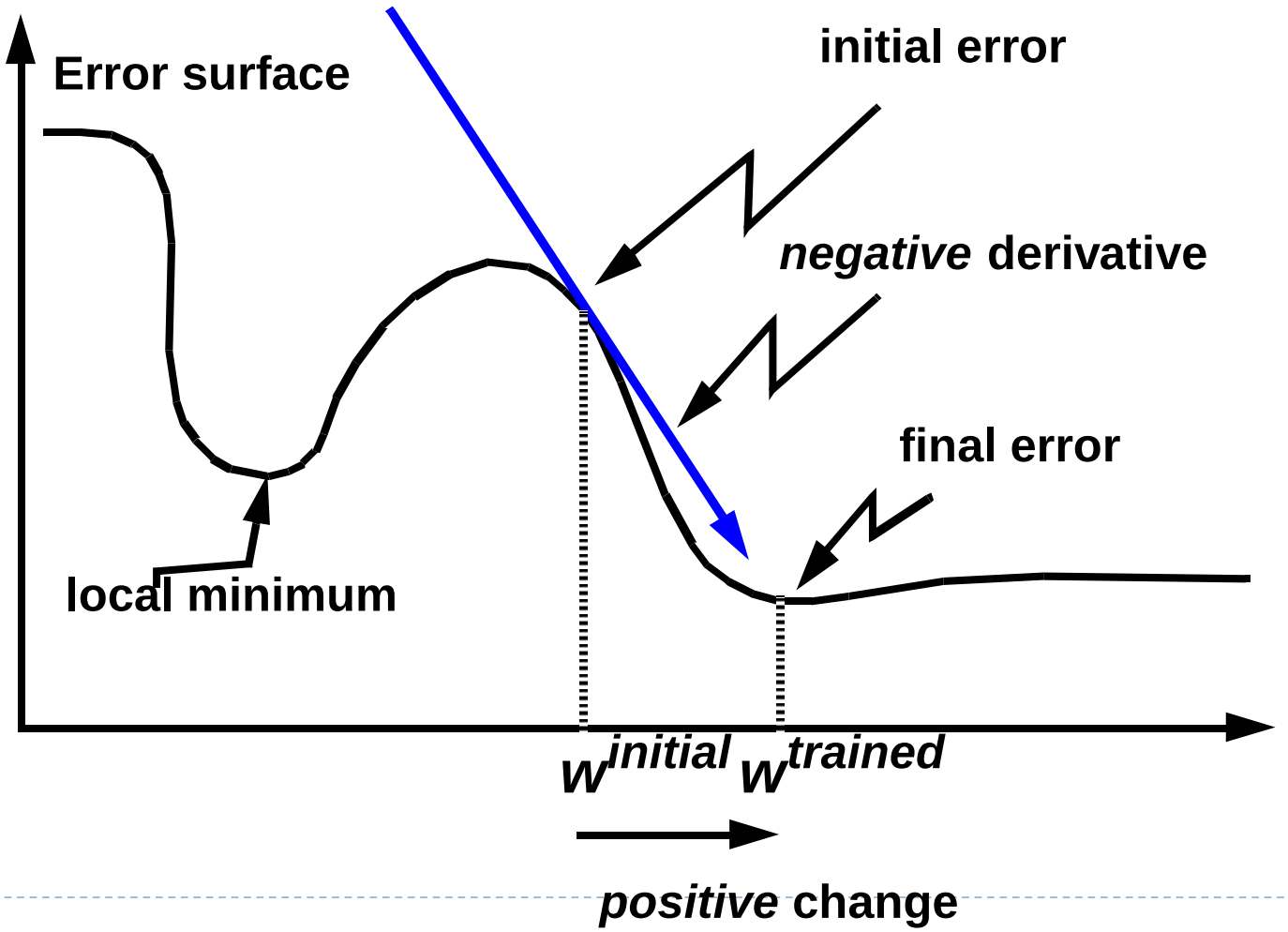
$$\nabla E[\vec{w}] \equiv \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \cdots \frac{\partial E}{\partial w_n}\right]$$

Training rule:

$$\Delta \vec{w} = -\eta \nabla E[\vec{w}]$$

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

Minimizing the Error



Backpropagation

- Sigmoid is easy to differentiate

$$\frac{\partial \sigma(y)}{\partial y} = \sigma(y) \cdot (1 - \sigma(y))$$

- For gradient descent on multiple layers, a little dynamic programming can help:
  - Compute errors at each output node
  - Use these to compute errors at each hidden node
  - Use these to compute weight gradient

The Backpropagation Algorithm

For each input training example, $\langle \vec{x}, \vec{t} \rangle$

1. Input instance $\vec{x}$ to the network and compute the output o_u for every unit u in the network
2. For each output unit k, calculate its error term δ_k

$$\delta_k \leftarrow o_k(1-o_k)(t_k - o_k)$$

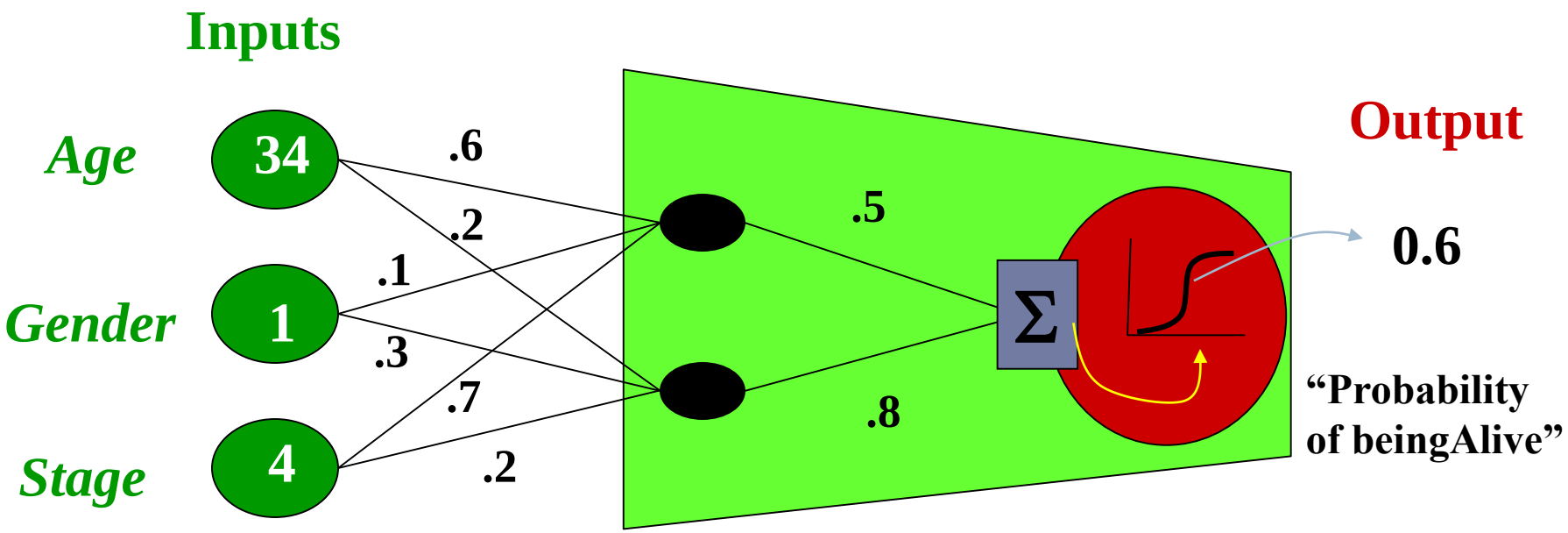
3. For each hidden unit h, calculate its error term δ_h

$$\delta_h \leftarrow o_h(1-o_h) \sum_{k \in outputs} w_{hk}\delta_k$$

4. Update each network weight w_{ji}

$$w_{ji} \leftarrow w_{ji} + \eta \delta_k x_{ji}$$

Learning Weights



Independent variables

Weights

Hidden Layer

Weights

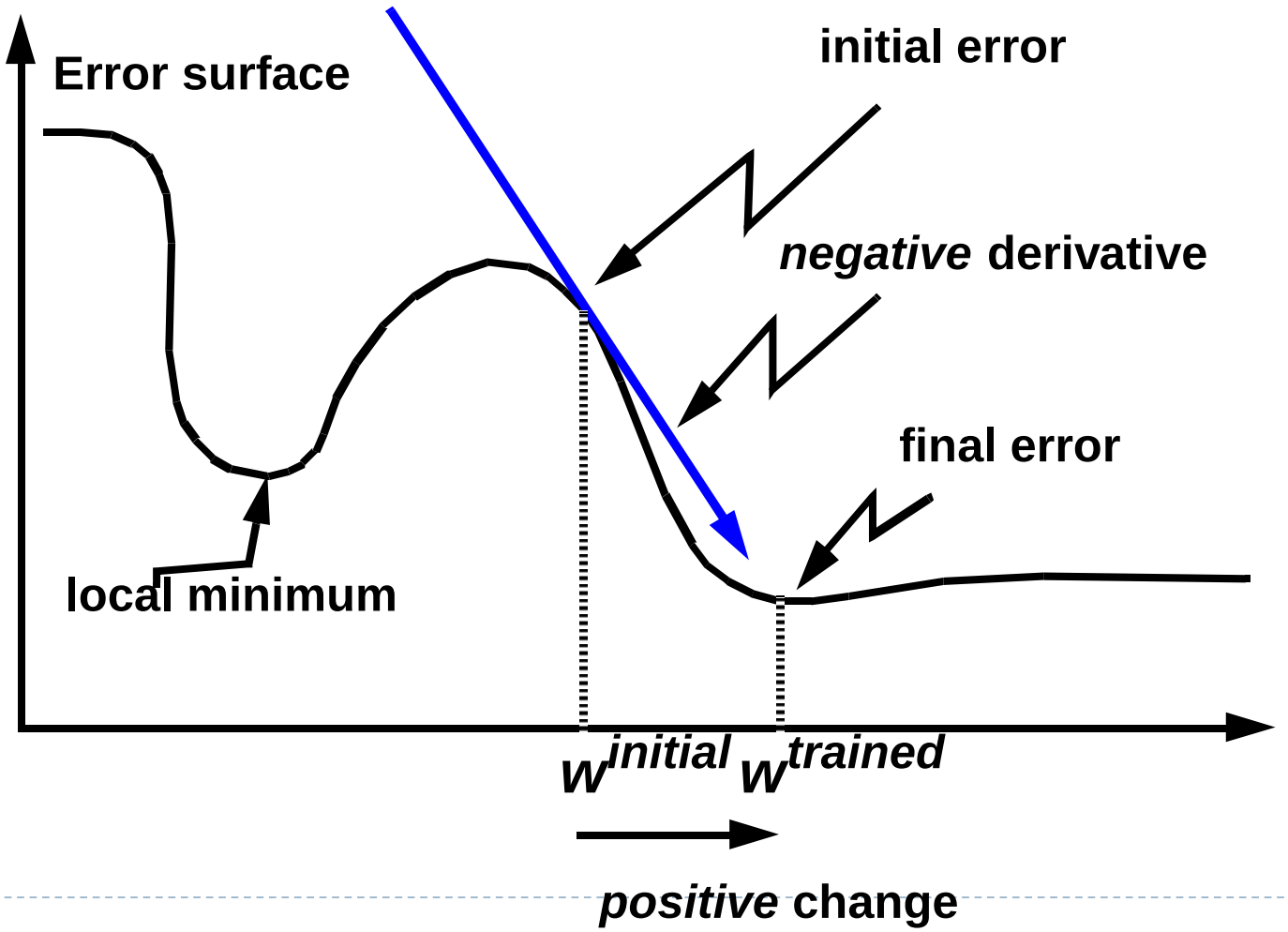
Dependent variable

Prediction

The fine print

- Don't implement back-propagation
  - Use a package
  - Second-order or variable step-size optimization techniques exist
- Feature normalization
  - Typical to normalize inputs to lie in [0,1]
    - (and outputs must be normalized)
- Problems with NN training:
  - Slow training times (though, getting better)
  - Local minima

Minimizing the Error

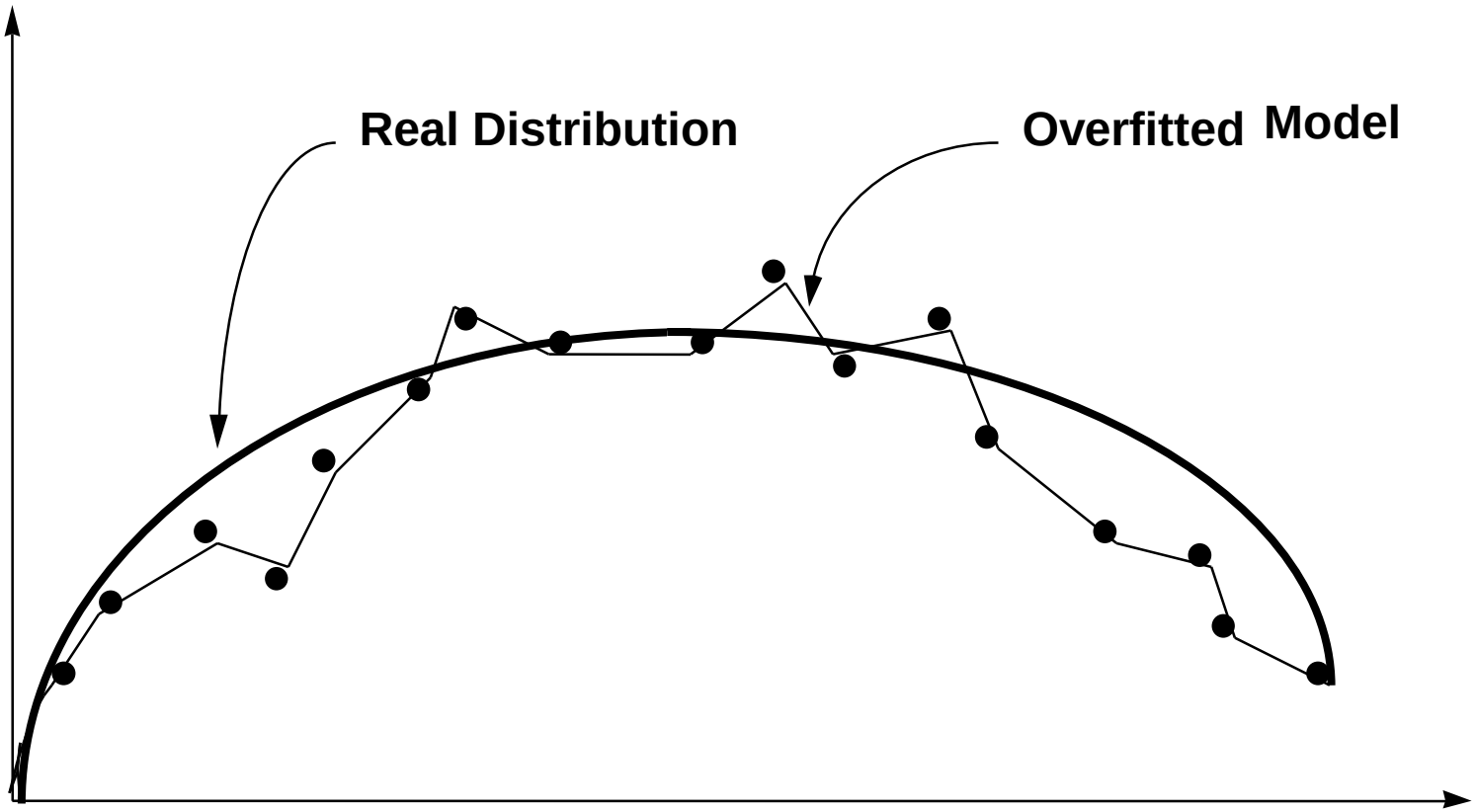


Expressive Power of NNs

- Universal Function Approximator:
  - Given enough hidden units, can approximate *any* continuous function f
- Need 2+ hidden units to learn XOR

- Why not use millions of hidden units?
  - Efficiency (training is slow)
  - Overfitting

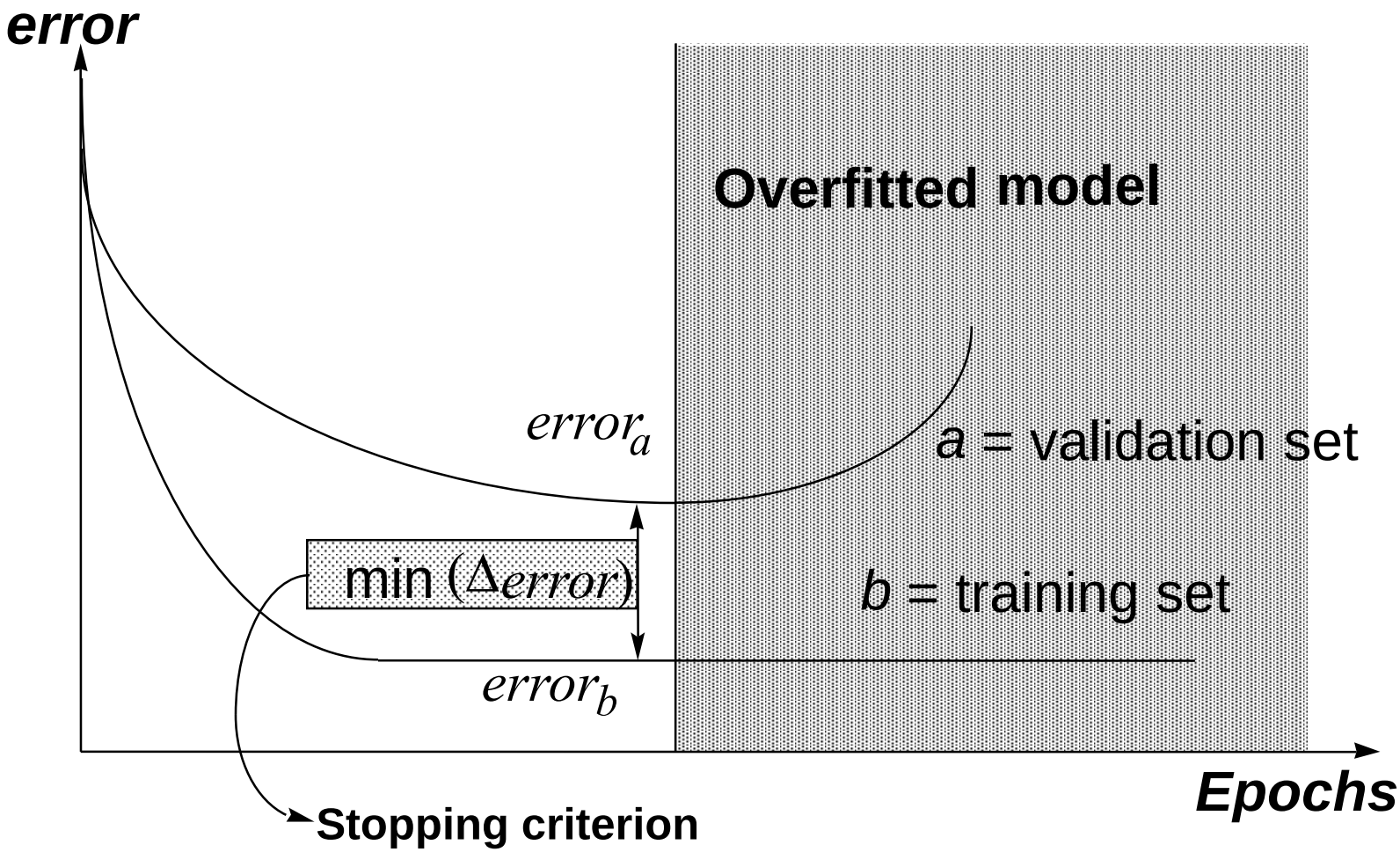
Overfitting



Combating Overfitting in Neural Nets

- Many techniques
- In language modeling, "early stopping" is the go-to technique
  - Use "a lot" of hidden units
  - Just don't over-train

Early Stopping



Learning Rate?

- A "knob" you twist empirically
  - Important
- One popular option: look for validation set acc to decrease/stabilize, then halve learning rate

Neural Networks Today

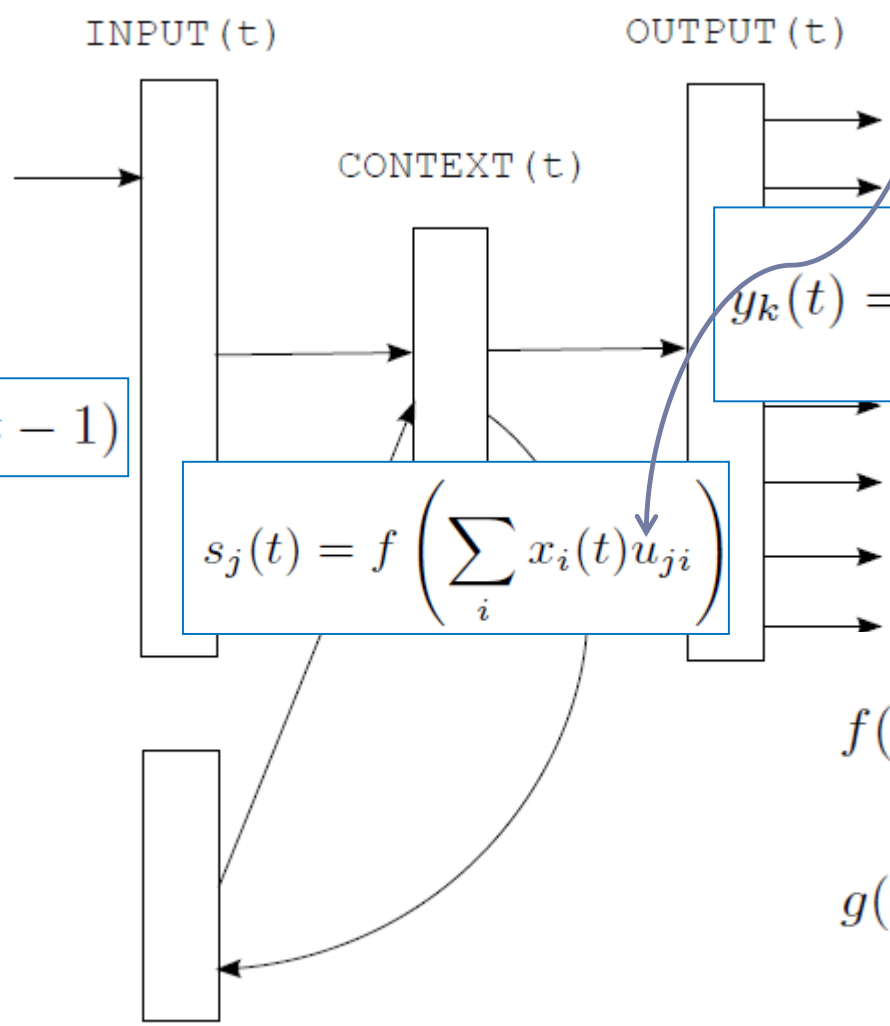
- Democratizing software packages exist
  - Tensorflow
    - Keras
  - Theano, Caffe, Torch…
- Use them!

Outline for Today

- Statistical Language Models (SLMs): What are they?
- Brief Neural Net (NN) primer
- **NN SLMs**
- Class structure
  - Class participation
  - Projects

Weights

Concatenation



$$x(t) = w(t) + s(t-1)$$

$$y_k(t) = g\left(\sum_j s_j(t) v_{kj}\right)$$

$$s_j(t) = f\left(\sum_i x_i(t) u_{ji}\right)$$

1 of |V| encoding

$$f(z) = \frac{1}{1+e^{-z}}$$

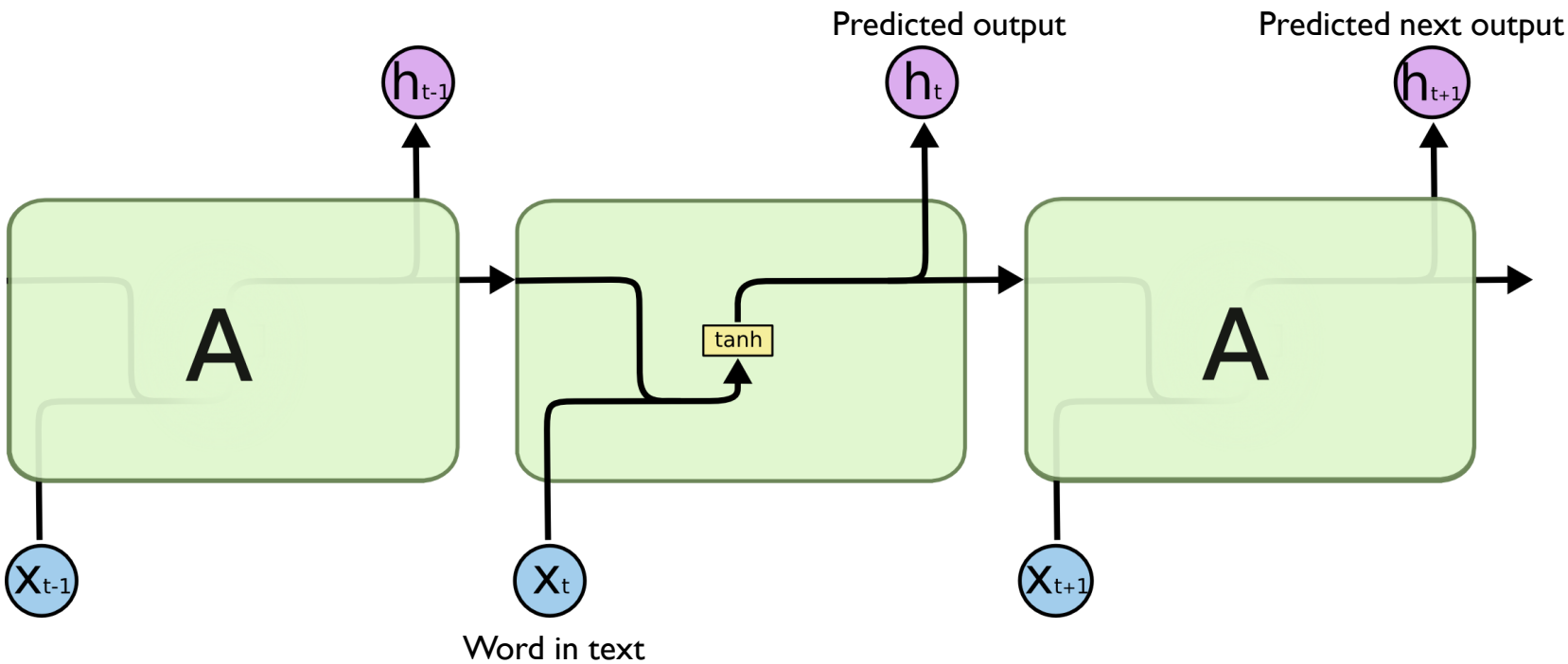
$$g(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}}$$

http://www.fit.vutbr.cz/research/groups/speech/publi/2010/mikolov_interspeech2010_IS100722.pdf

Word Embeddings

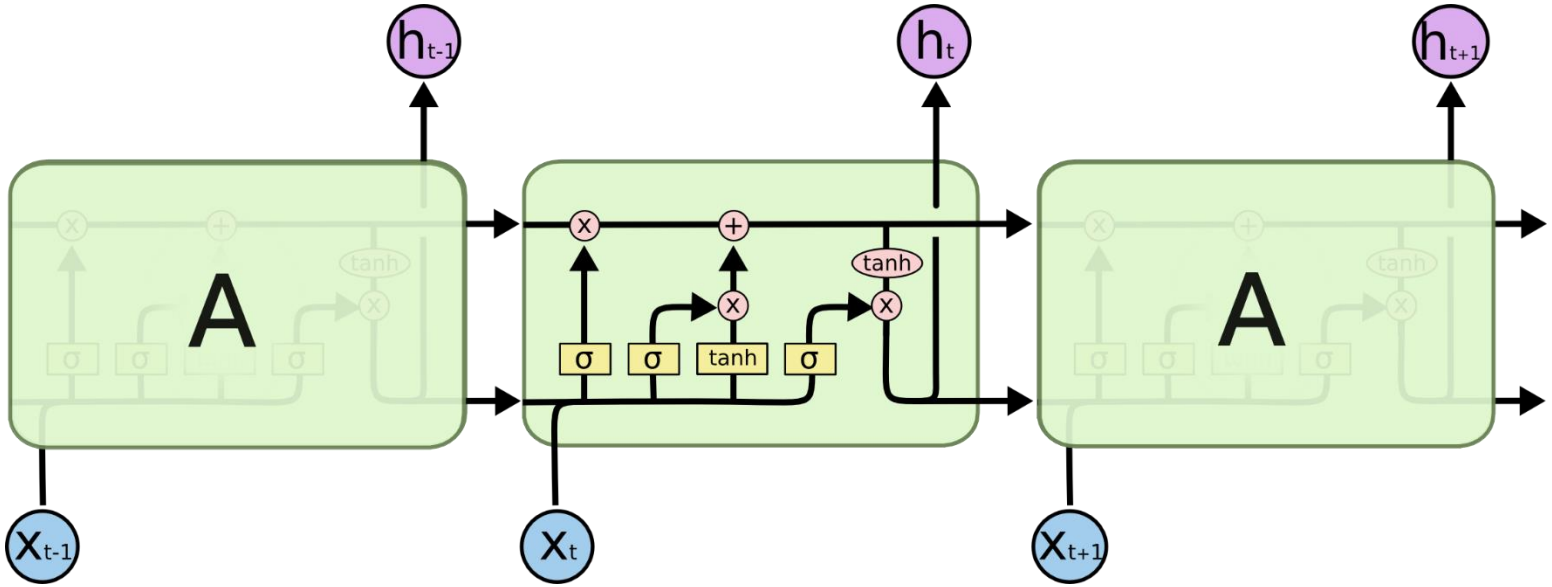
- The neural networks represent each word internally as a vector of weights
  - These have a surprising ability to capture semantics
  - The *distributional hypothesis*:
    - A word is known by the company it keeps
    - Words with similar meanings tend to appear in similar contexts [Harris, 1954]
  - Words with similar vectors tend to have similar meanings

RNNs



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTMs



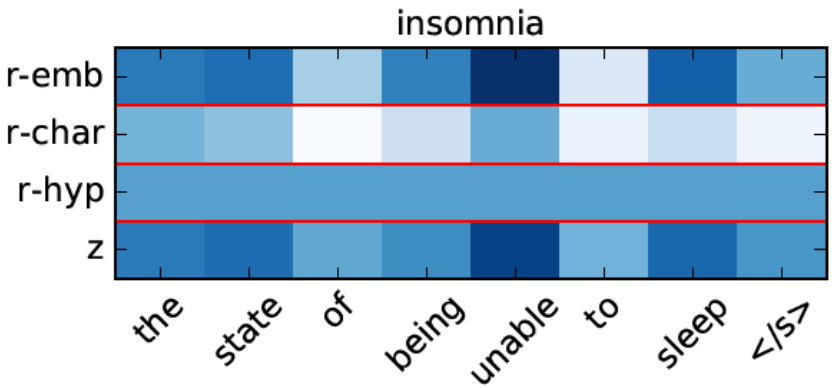
Intuition behind LSTMs

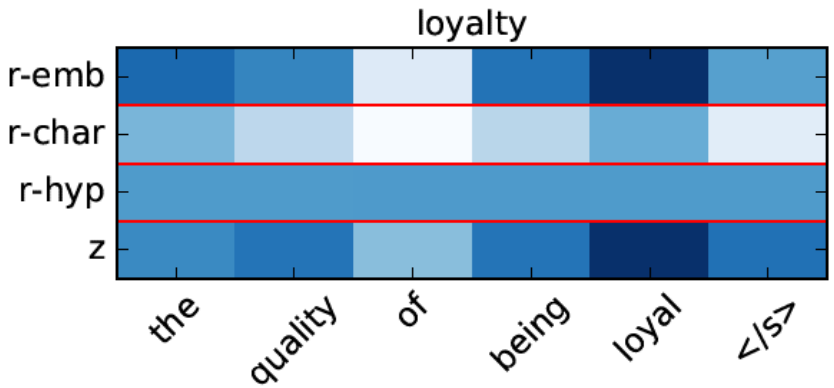
- Harness long-range dependencies
- "She's _____ so I hope that she will _____ ."

  brilliant — join us

  sick — not breathe on me

LSTM example





Noraset et al., AAAI 2017

results

- Incredibly good language modeling results
  - Good Turing Smoothing: ~160 perplexity (lower is better)
  - Kneser-Ney smoothing: ~140 perplexity
  - Today's best LSTMs: ~70 perplexity [Google, ca 2015]

Perplexity numbers on Penn TreeBank data set (approx.)

Outline for Today

- Statistical Language Models (SLMs): What are they?
- Brief Neural Net (NN) primer
- NN SLMs
- **Class structure**
  - **Class participation**
  - **Projects**

Class Participation

- Summarizing papers (10% of grade, in Canvas)
- Class participation (10% of grade, in Canvas)
  - After each class, report what you said. Number of comments, content of each.
- Presenting papers (30% of grade)
  - You will lead discussions of papers in teams of ~2
  - Crucial to solicit participation
    - Think about ways to structure discussion so that people can get involved, even if class is large

Projects

- Generate text, e.g.: http://karpathy.github.io/2015/05/21/rnn-effectiveness/ https://arxiv.org/abs/1612.00394 https://datanice.wordpress.com/2015/10/08/beer-reviews-with-recurrent-neural-networks-rnn/
  - what other tasks could you do this with?
  - Characteristics to look for: utility, tractability, degree to which you can exploit problem structure
- Classify text, e.g. sentiment classification, relation extraction
- Other stuff (you have ideas, I can also give more suggestions)