Sequence Models

Machine Learning: Jordan Boyd-Graber
University of Maryland
RNNS

Slides adapted from Richard Socher

Language models

- **Language models** answer the question: How likely is a string of English words good English?
- Autocomplete on phones and websearch
- Creating English-looking documents
- Very common in machine translation systems
  - Help with reordering / style

$$p_{\text{lm}}(\text{the house is small}) > p_{\text{lm}}(\text{small the is house})$$

  - Help with word choice

$$p_{\text{lm}}(\text{I am going home}) > p_{\text{lm}}(\text{I am going house})$$

N-Gram Language Models

- Given: a string of English words $W = w_1, w_2, w_3, ..., w_n$
- Question: what is $p(W)$?
- Sparse data: Many good English sentences will not have been seen before

$\longrightarrow$ Decomposing $p(W)$ using the chain rule:

$$p(w_1, w_2, w_3, ..., w_n) = \\ p(w_1)\, p(w_2|w_1)\, p(w_3|w_1, w_2) \dots p(w_n|w_1, w_2, ... w_{n-1})$$

(not much gained yet, $p(w_n|w_1, w_2, ... w_{n-1})$ is equally sparse)

Markov Chain

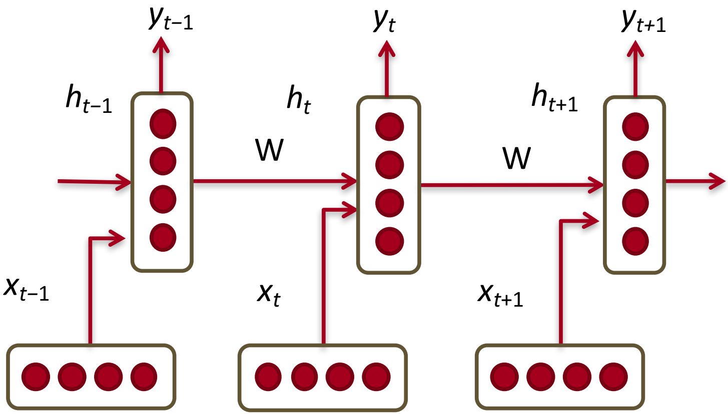
- **Markov independence assumption**:
  - only previous history matters
  - limited memory: only last k words are included in history (older words less relevant)

  $\longrightarrow$ k**th order Markov model**
- For instance 2-gram language model:

$$p(w_1, w_2, w_3, ..., w_n) \simeq p(w_1)\, p(w_2|w_1)\, p(w_3|w_2)...p(w_n|w_{n-1})$$

- What is conditioned on, here w_{i-1} is called the **history**. Estimated from counts.

Recurrent Neural Networks



- Condition on all previous words
- Hidden state at each time step

RNN parameters

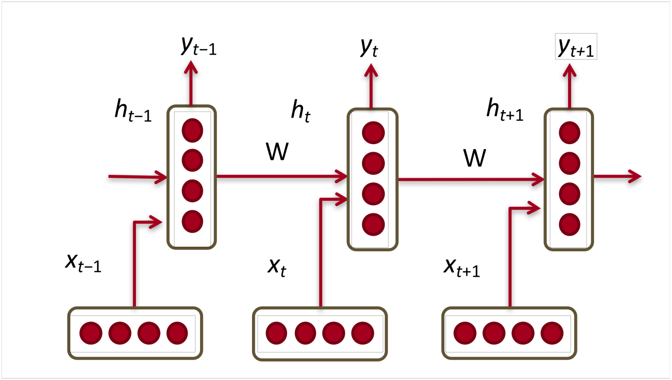
$$h_t = f(W^{(hh)} h_{t-1} + W^{(hx)} x_t) \quad (1)$$

$$\hat{y}_t = \text{softmax}(W^{(S)} h_t) \quad (2)$$

$$P(x_{t+1} = v_j \mid x_t, \ldots x_1) = \hat{y}_{t,j} \quad (3)$$

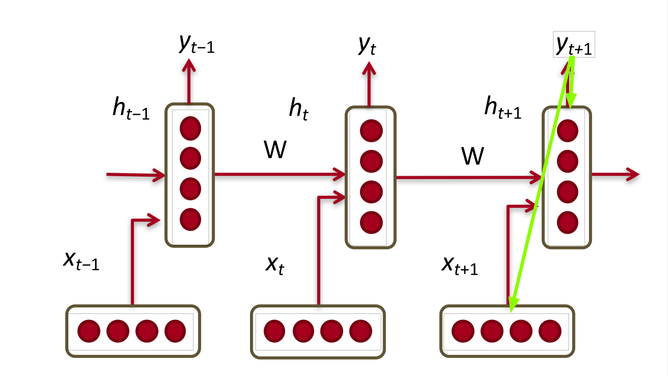
- Learn parameter h_0 to initialize hidden layer
- x_t is representation of input (e.g., word embedding)
- $\hat{y}$ is probability distribution over vocabulary

Training Woes



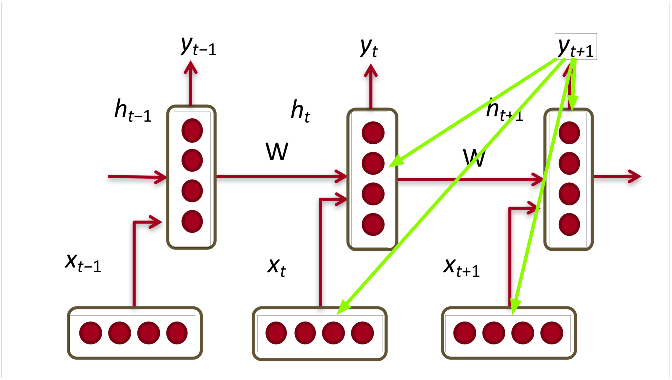
Multiplying same matrix over and over

Training Woes



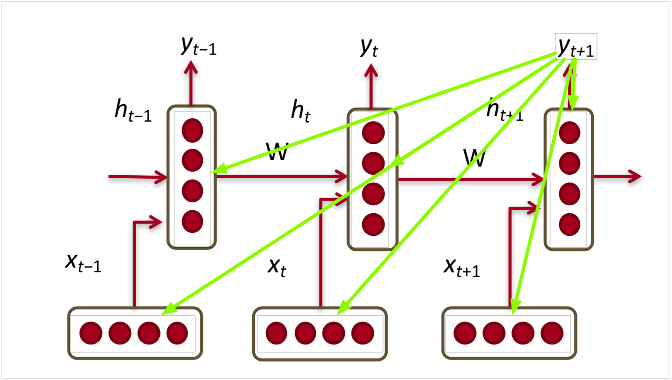
Multiplying same matrix over and over

Training Woes



Multiplying same matrix over and over

Training Woes



Multiplying same matrix over and over

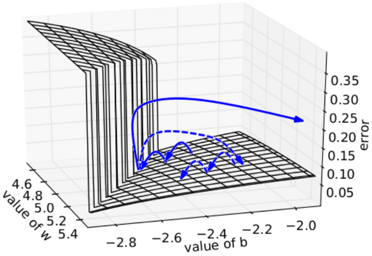
Vanishing / Exploding Gradient

- Work out the math:
  - Define β_W / β_h as upper bound of norms of W, h
  - Bengio et al 1994: Partial derivative is $(\beta_W \beta_h)^{t-k}$
  - This can be very small or very big
- If it's big, SGD jumps too far
- If it's small, we don't learn what we need: "Jane walked into the room. John walked in too. It was late in the day. Jane said hi to ____"

Gradient Clipping

Algorithm 1 Pseudo-code for norm clipping the gradients whenever they explode

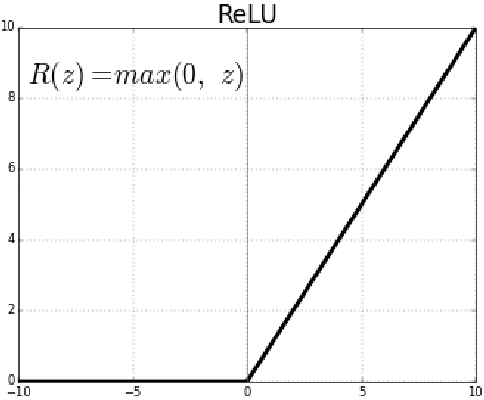
$\hat{\mathbf{g}} \leftarrow \frac{\partial \mathcal{E}}{\partial \theta}$
if $\|\hat{\mathbf{g}}\| \geq threshold$ **then**
  $\hat{\mathbf{g}} \leftarrow \frac{threshold}{\|\hat{\mathbf{g}}\|} \hat{\mathbf{g}}$
end if



From Pascanu et al. 2013

- If they get too big, stop at boundary
- Prevents (dashed) values from jumping around (solid)

Fixing Vanishing Gradients



- ReLU activation
- Initialize W to identity matrix

RNN Recap

- Simple model
- Complicated training (but good toolkits available)
- Do we need to remember everything?