Automatic Speech Recognition (CS753)

Lecture 17: RNN language models + Introduction to Kaldi

Instructor: Preethi Jyothi

Mar 20, 2017

Recap

- Language models using feedforward neural networks with fixed length context

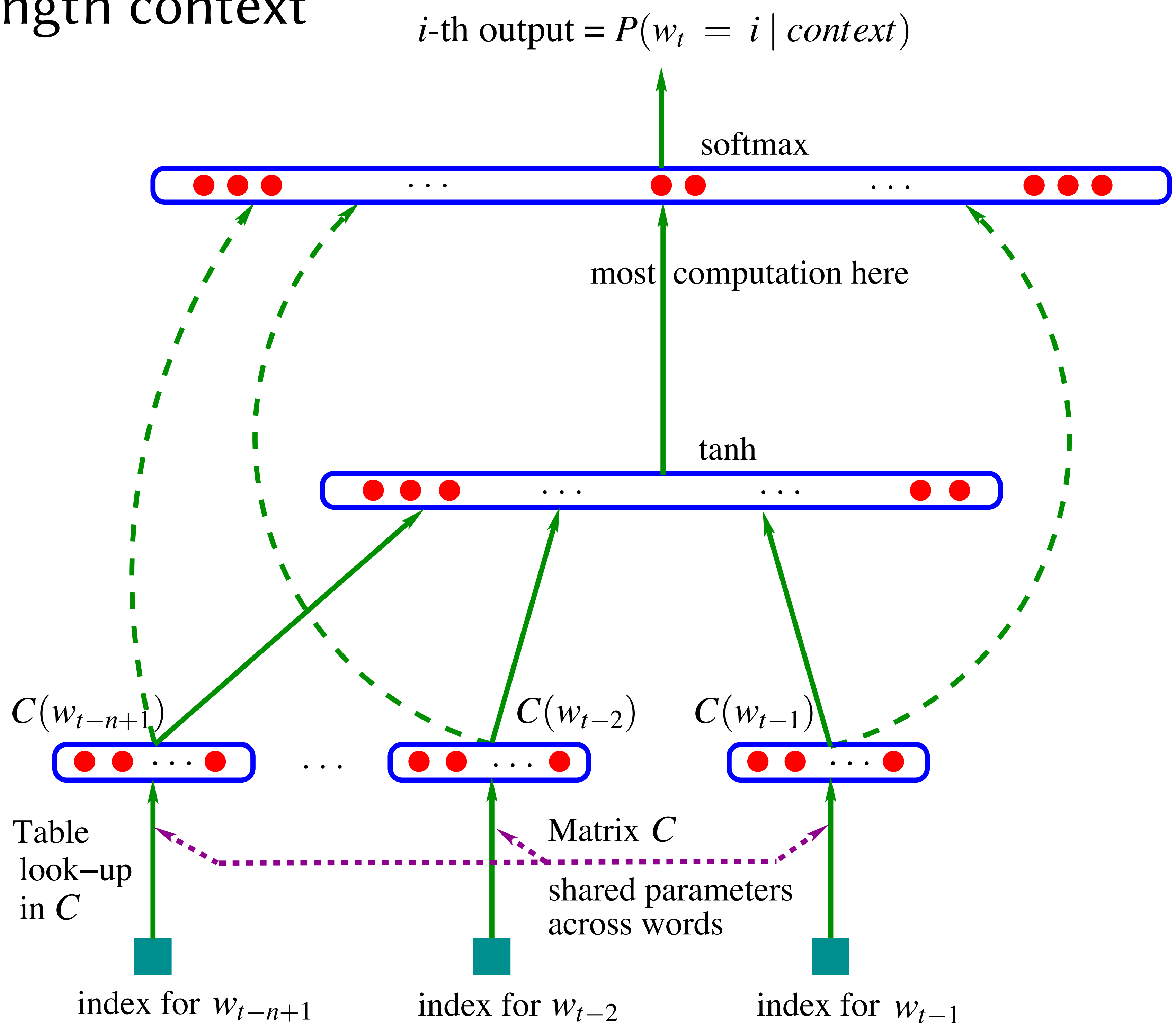


Image from: Bengio et al., "A neural probabilistic language model", JMLR, 03

Recap

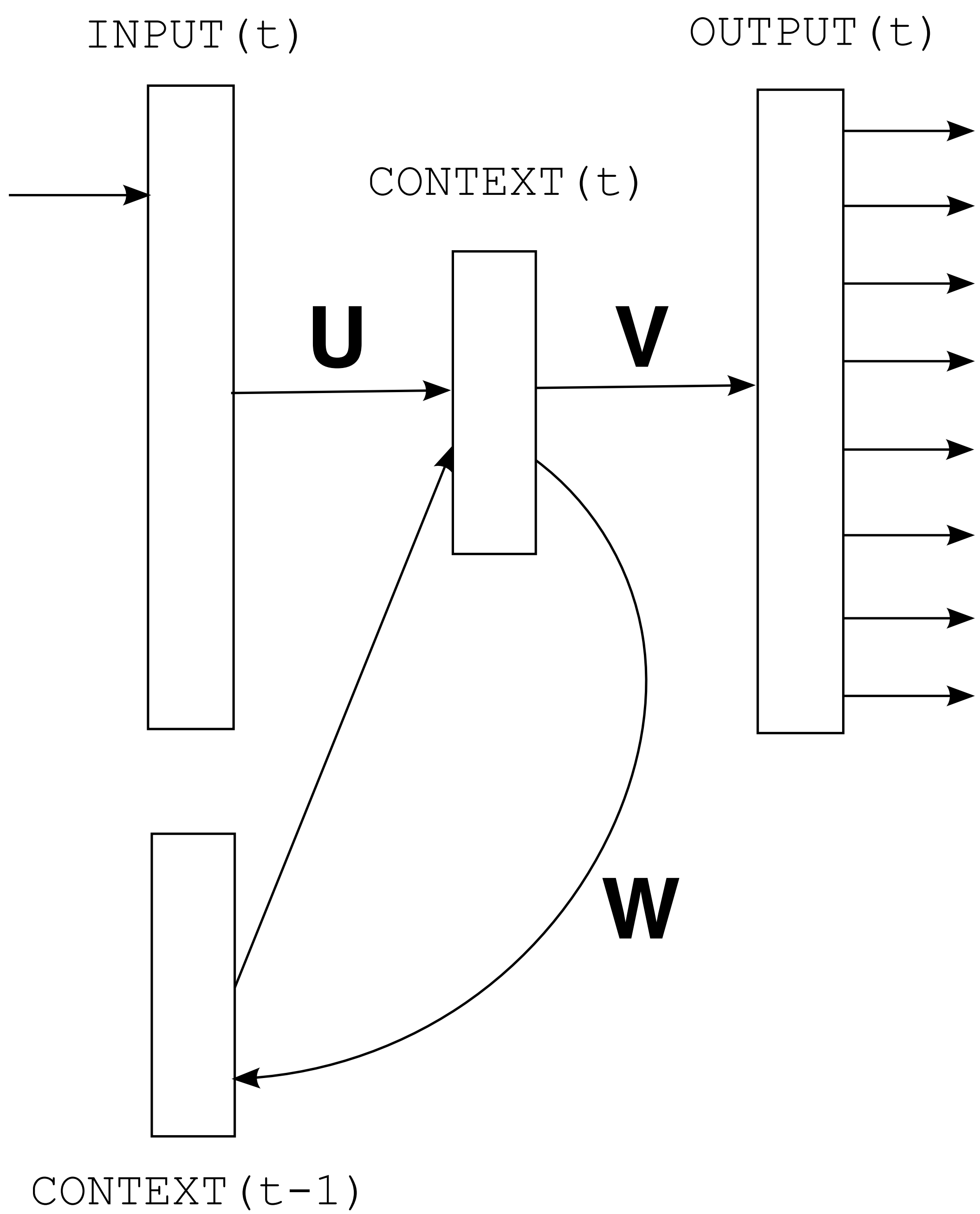
- Language models using feedforward neural networks with fixed length context: Provide significant improvements in ASR performance over Ngram based language models

- But fixed length context needs to be specified before training

- Recurrent neural networks (RNNs) do not use limited size of context

- Build RNN-based language models

Simple RNN language model



- Current word, x_t
  Hidden state, s_t
  Output, y_t

$$s_t = f(Ux_t + Ws_{t-1})$$
$$o_t = \text{softmax}(Vs_t)$$

- RNN is trained using the cross-entropy criterion

Image from: Mikolov et al., "Recurrent neural network based language model", Interspeech 10

RNN-LMs

- Optimizations used for NNLMs are relevant to RNN-LMs as well (rescoring Nbest lists or lattices, using a shortlist, etc.)
- Perplexity reductions over Kneser-Ney models:

Model	# words	PPL	WER
KN5 LM	200K	336	16.4
KN5 LM + RNN 90/2	200K	271	15.4
KN5 LM	1M	287	15.1
KN5 LM + RNN 90/2	1M	225	14.0
KN5 LM	6.4M	221	13.5
KN5 LM + RNN 250/5	6.4M	156	11.7

Image from: Mikolov et al., "Recurrent neural network based language model", Interspeech 10

Training RNN-LMs

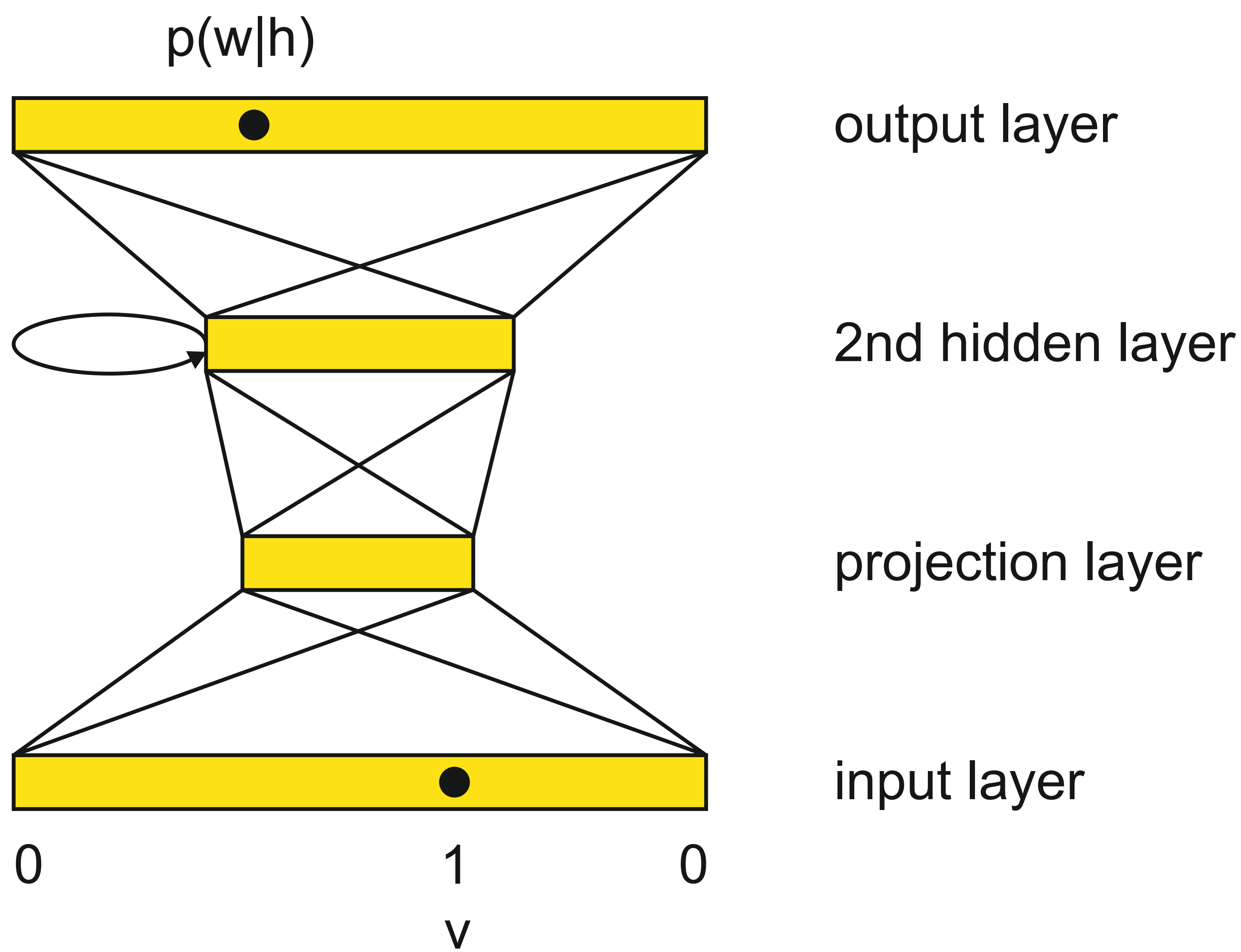
- RNN-LMs are trained using backpropagation through time (BPTT): Unfold the RNN in time + train the unfolded RNN using backpropagation + mini-batch gradient descent

- Main issues with BPTT: Exploding and vanishing gradients

  - Exploding gradients: Gradients can increase exponentially over time during backpropagation. Clip values of gradients to handle this.

  - Vanishing gradients: Magnitude of gradients approach very tiny values as we propagate gradients back in time. Architectures like Long Short Term Memory (LSTMs) networks can handle this.

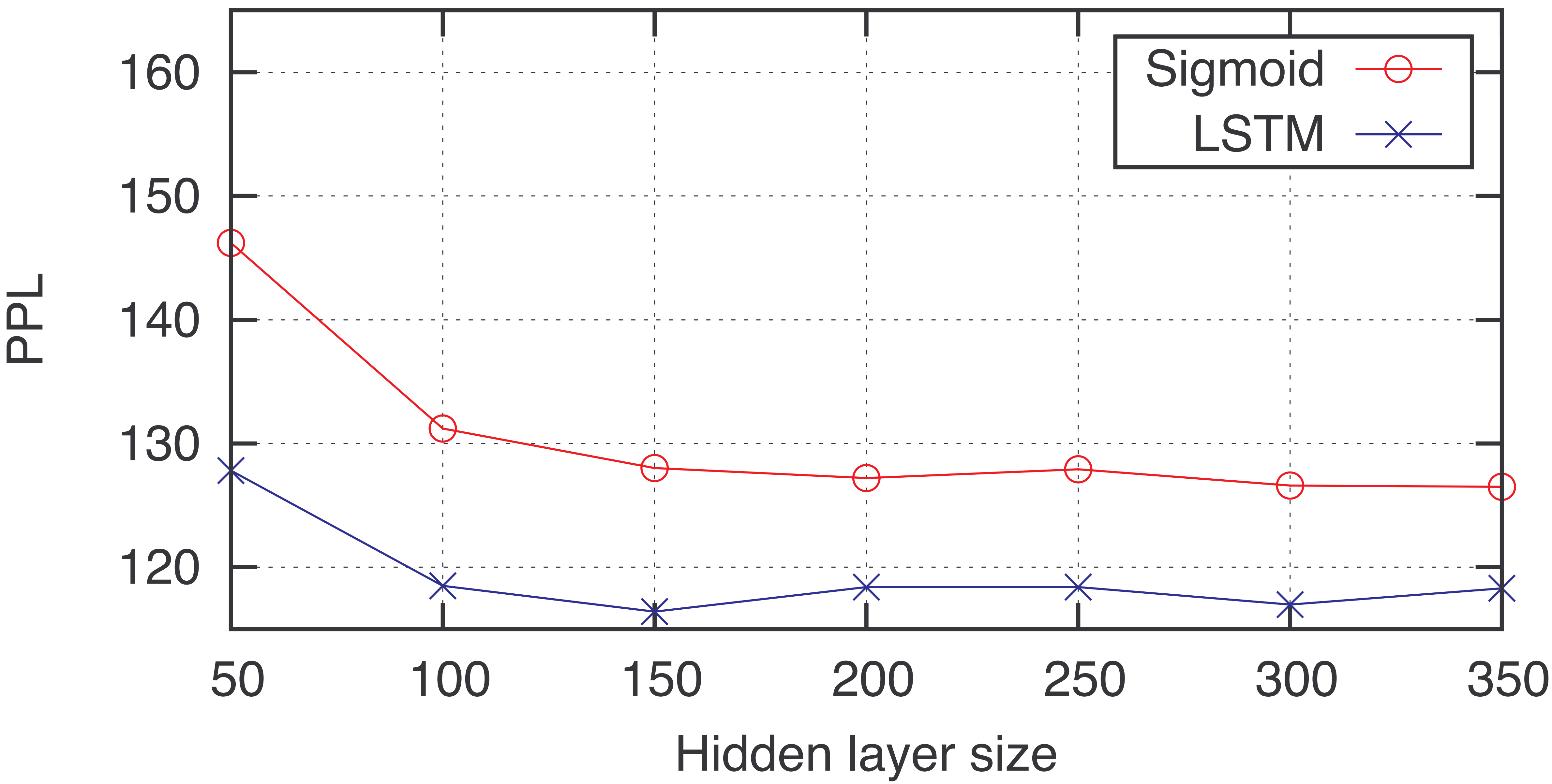
LSTM-LMs



- Vanilla RNN-LMs unlikely to show full potential of recurrent models due to issues like vanishing gradients
- LSTM-LMs: Similar to RNN-LMs except use LSTM units in the 2nd hidden (recurrent) layer

Image from: Sundermeyer et al., "LSTM NNs for Language Modeling",

Comparing RNN-LMs with LSTM-LMs



Character-based RNN-LMs

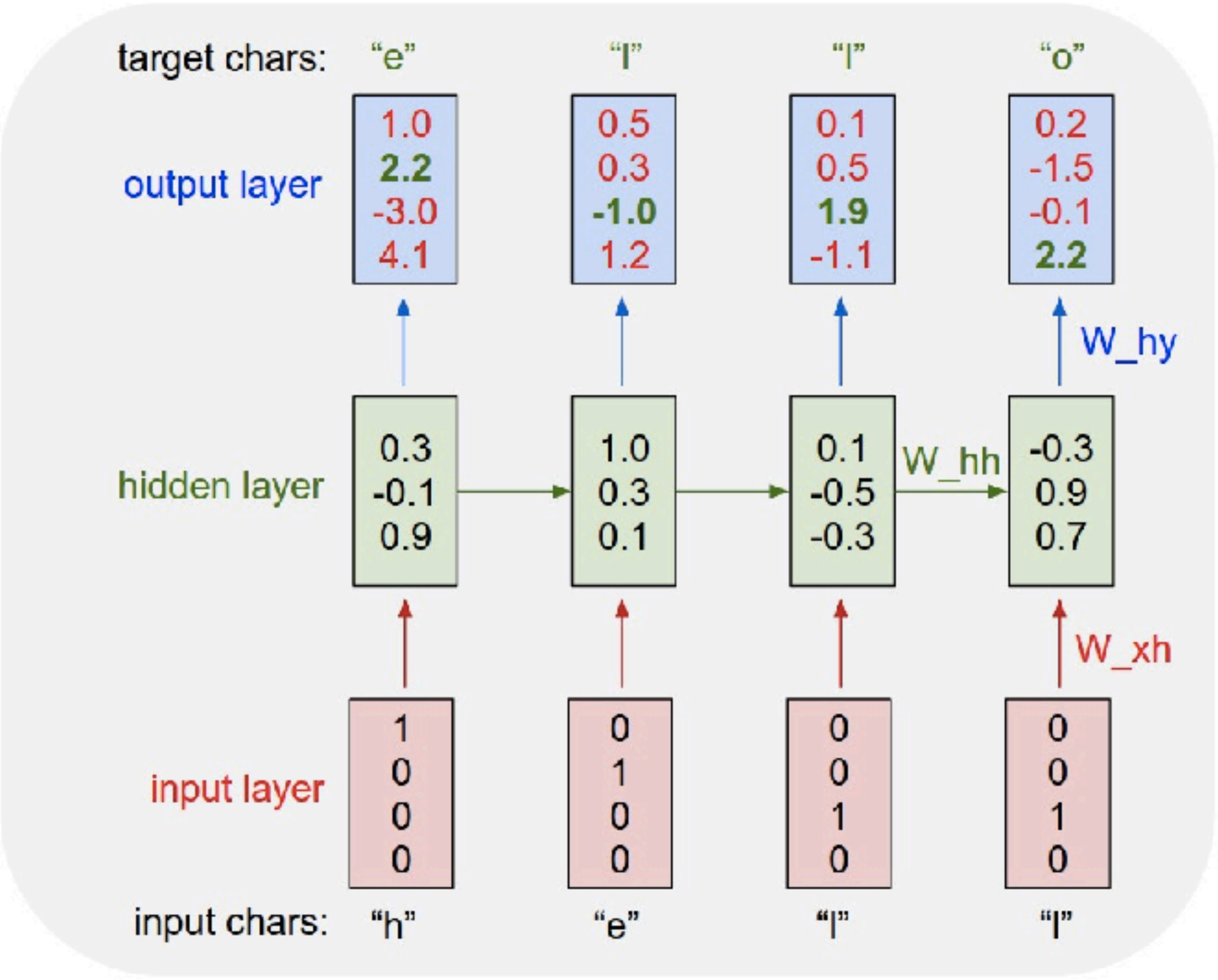


Image from: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Generate text using a trained character-based LSTM-LM

VIOLA:
Why, Salisbury must find his flesh and thought
That which I am not aps, not a man and in fire,
To show the reining of the raven and the wars
To grace my hand reproach within, and not a fair are hand,
That Caesar and my goodly father's world;
When I was heaven of presence and our fleets,
We spare with hours, but cut thy council I am great,
Murdered and by thy master's ready there
My power to give thee but so much as hell:
Some service in the noble bondman here,
Would show him to her wine.

Image from: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

The Big Picture

Putting it all together: How do we recognise an utterance?

- *A:* speech utterance
- O_A: acoustic features corresponding to the utterance A

$$W^* = \arg\max_W \Pr(O_A|W)\Pr(W)$$

- Return the word sequence that jointly assigns the highest probability to O_A
- How do we estimate $\Pr(O_A|W)$ and $\Pr(W)$?
- How do we decode?

Acoustic model

$$W^* = \arg\max_W \boxed{\Pr(O_A|W)} \Pr(W)$$

$$\begin{aligned}
\Pr(O_A|W) &= \sum_Q \Pr(O_A, Q|W) \\
&= \sum_{q_1^T, w_1^N} \prod_{t=1}^{T} \Pr(O_t|O_1^{t-1}, q_1^t, w_1^N) \Pr(q_t|q_1^{t-1}, w_1^N) \\
&\approx \sum_{q_1^T, w_1^N} \prod_{t=1}^{T} \Pr(O_t|q_t, w_1^N) \Pr(q_t|q_{t-1}, w_1^N) \\
&\approx \max_{q_1^T, w_1^N} \prod_{t=1}^{T} \Pr(O_t|q_t, w_1^N) \Pr(q_t|q_{t-1}, w_1^N)
\end{aligned}$$

First-order HMM assumptions

Viterbi approximation

Acoustic Model

Transition probabilities

$$\Pr(O_A|W) = \max_{q_1^T, w_1^N} \prod_{t=1}^{T} \Pr(O_t|q_t, w_1^N) \Pr(q_t|q_{t-1}, w_1^N)$$

Emission probabilities

$$\Pr(O|q; w_1^N) = \sum_{\ell=1}^{L_q} c_{q\ell} \mathcal{N}(O|\mu_{q\ell}, \Sigma_{q\ell}; w_1^N)$$

Modeled using a mixture of Gaussians

- All the free parameters (means, covariances, mixture weights, transition probabilities) are learned using EM (Baum-Welch) algorithm

Language Model

$$W^* = \arg\max_W \Pr(O_A|W) \Pr(W)$$

$$\begin{aligned}\Pr(W) &= \Pr(w_1, w_2, \ldots, w_N)\\ &= \Pr(w_1) \ldots \Pr(w_N|w_{N-m+1}^{N-1})\end{aligned}$$

m-gram language model

- Further optimized using smoothing and interpolation with lower-order Ngram models

Decoding: Search

$$W^* = \boxed{\arg\max_{W} \Pr(O_A|W)\Pr(W)}$$

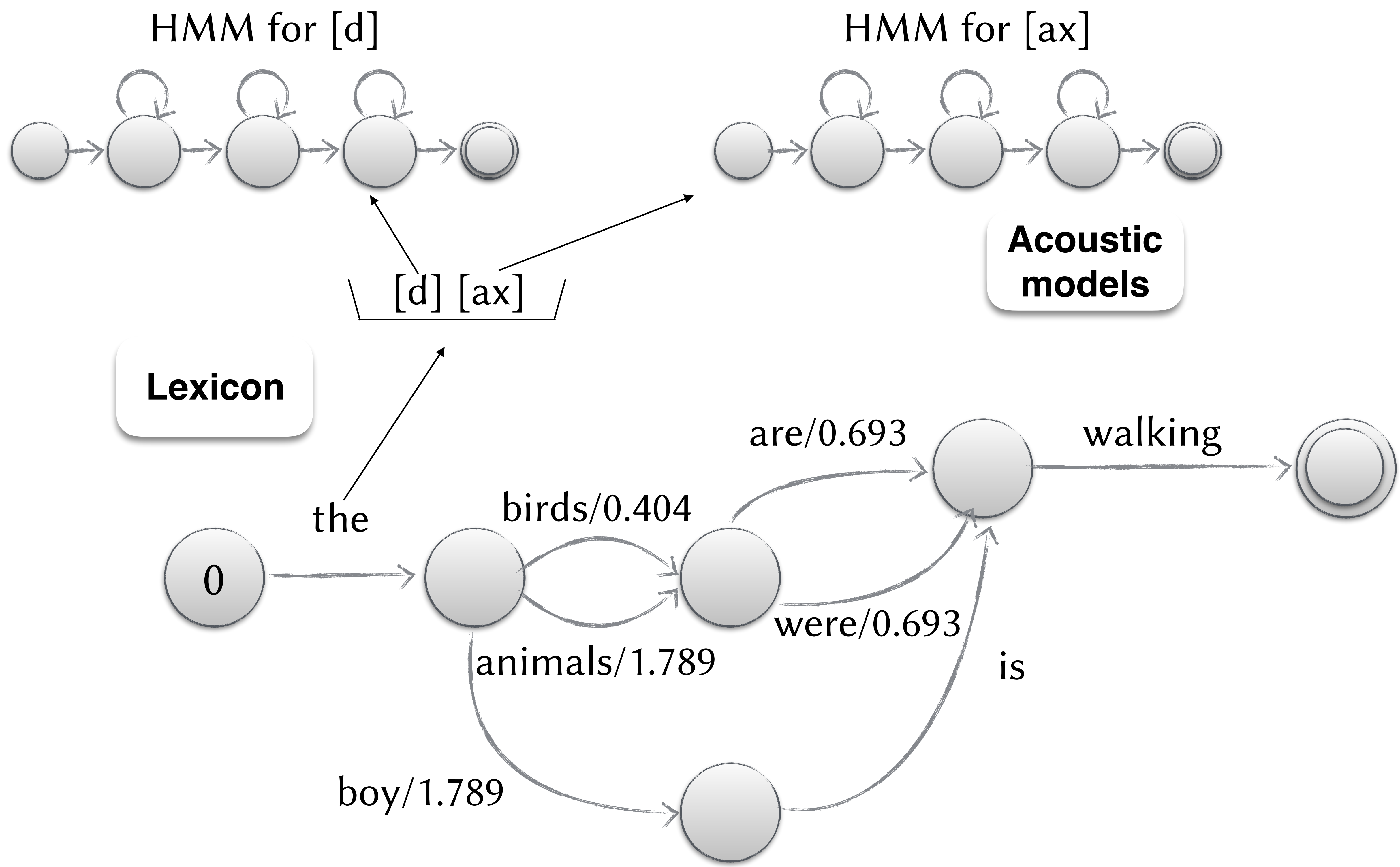
$$W^* = \arg\max_{w_1^N, N} \left\{ \left[\prod_{n=1}^{N} \Pr(w_n|w_{n-m+1}^{n-1}) \right] \left[\sum_{q_1^T, w_1^N} \prod_{t=1}^{T} \Pr(O_t|q_t, w_1^N) \Pr(q_t|q_{t-1}, w_1^N) \right] \right\}$$

Viterbi

$$\approx \arg\max_{w_1^N, N} \left\{ \left[\prod_{n=1}^{N} \Pr(w_n|w_{n-m+1}^{n-1}) \right] \left[\max_{q_1^T, w_1^N} \prod_{t=1}^{T} \Pr(O_t|q_t, w_1^N) \Pr(q_t|q_{t-1}, w_1^N) \right] \right\}$$

- Viterbi approximation divides the above optimisation problem into sub-problems that allows the efficient application of dynamic programming
- Search space still very huge for LVCSR tasks! Use approximate decoding techniques (A* decoding, beam-width decoding, etc.) to visit only promising parts of the search space

ASR Search



HMM for [d]
HMM for [ax]
Acoustic models
[d] [ax]
Lexicon
the
0
birds/0.404
animals/1.789
boy/1.789
are/0.693
were/0.693
is
walking

Kaldi Toolkit & Assignment 2