COM4511 Speech Technology: Advanced Language Models

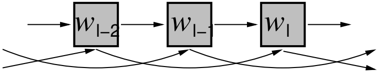
Anton Ragni

March 2, 2020

N-gram Language Models

(a) Unigram DBN

(b) Bigram DBN



(c) Trigram DBN

- N-gram language model for general order n

$$P(\boldsymbol{w}_{1:L}) = P(w_1, \ldots, w_L) = \prod_{l=1}^{L} P(w_l | w_{l-1}, \ldots, w_1) \approx \prod_{l=1}^{L} P(w_l | w_{l-1}, \ldots, w_{l-n+1})$$

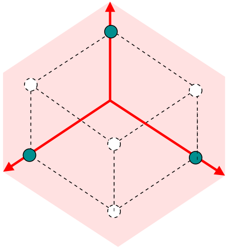
  - words further than $n-1$ positions in the past does not affect probability
- Two primary issues
  - Markov assumption
  - discrete word representation

1-of-K (One-Hot) Encoding

- Any discrete value can be mapped to continuous, vector, representation

$$\boldsymbol{w}_l = \begin{bmatrix} \delta(w_l, w^{(1)}) \\ \delta(w_l, w^{(2)}) \\ \vdots \\ \delta(w_l, w^{(|\mathcal{V}|)}) \end{bmatrix}$$

(a) Vector representation



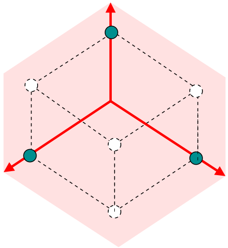
(b) Coordinate space

  - each distinct discrete value is assigned a unique coordinate
  - BUT does not encode any word relationship information
- Example
  - vocabulary of digits $\mathcal{V} = \{\text{zero}, \text{one}, \text{two}, \text{three}, \text{four}, \text{five}, \text{six}, \text{seven}, \text{eight}, \text{nine}\}$
  - write down one-hot encoding matrix $\boldsymbol{V}_{N\times M}$ for 21925, what are N and M?

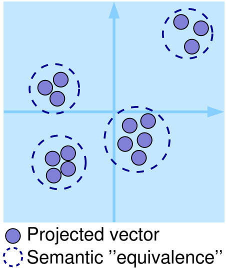
Anton Ragni

Word Embedding

- Project one-hot vectors into a lower-dimensional space



(a) 1-of-K space

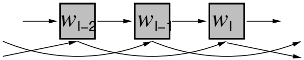


(b) Embedding space

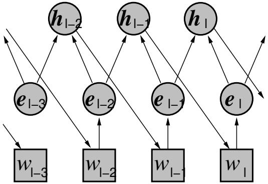
  - optimal embedding is space where location and distances reflect word semantics
- Multiple criteria possible
  - co-occurrence statistics, word2vec, language model

Anton Ragni

Feed-Forward Neural Network



(a) Trigram DBN



(b) FFNN Trigram Pseudo DBN

- Use feed-forward neural network to predict n-gram probabilities
  - project 1-of-K vectors into a common embedding space

$$\boldsymbol{e}_{l-1} = \boldsymbol{E}_{N_1 \times |\mathcal{V}|} \boldsymbol{w}_{l-1_{|\mathcal{V}| \times 1}}, \quad \text{where} \quad N_1 \ll |\mathcal{V}|$$

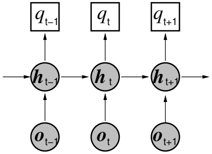
  - add suitable number of feed-forward layers $\boldsymbol{h}^{(l)} = \phi^{(l)}(\boldsymbol{A}^{(l)}\boldsymbol{h}^{(l-1)} + \boldsymbol{b}^{(l)})$
  - softmax non-linearity at the final year yields

$$P(w_l | w_{l-1}, \ldots, w_{l-n+1}) \triangleq h^{(L)}_{w_l}$$

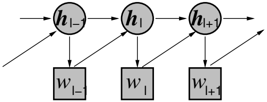
- Advantages compared to maximum likelihood estimates
  - continuous word representation
  - yields probabilities for all, seen and unseen, n-grams

Anton Ragni

Recurrent Neural Networks (RNN)



(a) Acoustic model $P(\boldsymbol{q}_{1:T}|\boldsymbol{O}_{1:T})$



(b) Language model $P(\boldsymbol{w}_{1:L})$

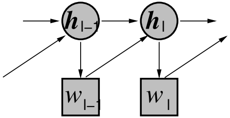
- Relax Markov assumption by modelling complete word history

$$P(\boldsymbol{w}_{1:L}) = \prod_{l=1}^{L} P(w_l|w_{l-1},\ldots,w_1) \approx \prod_{l=1}^{L} P(w_l|w_{l-1},\boldsymbol{h}_{l-1}) \approx \prod_{l=1}^{L} P(w_l|\boldsymbol{h}_l)$$

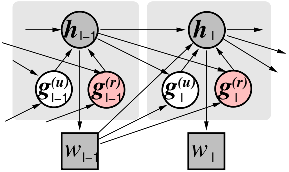
  - a theoretically optimal language model given ... (name conditions)

Anton Ragni

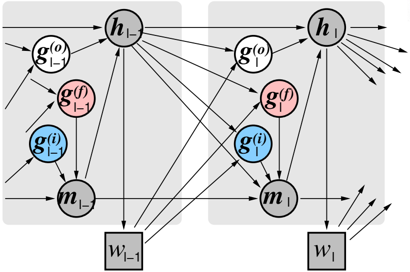
Recurrent Units (Revisited)



(a) Simple recurrent unit



(b) Gated recurrent unit



(b) Long short-term memory

- Simple recurrent unit

$$\boldsymbol{h}_l = \phi(\boldsymbol{A}^{(h)}\boldsymbol{h}_{l-1} + \boldsymbol{C}^{(h)}\boldsymbol{w}_{l-1} + \boldsymbol{b}^{(h)})$$

  - parameters $\boldsymbol{A}^{(h)}_{H\times H}$, $\boldsymbol{C}^{(h)}_{H\times|\mathcal{V}|}$ and $\boldsymbol{b}^{(h)}_{H\times 1}$, one-hot encoding $\boldsymbol{w}_{l-1}$, non-linearity ϕ (sigmoid)
  - all words embedded into continuous space parameterised by $\boldsymbol{C}^{(h)}$

Parameter Estimation

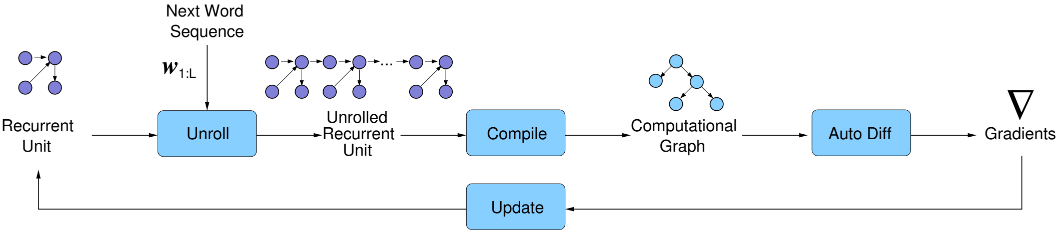
- Standard criterion for recurrent neural networks is cross-entropy*

$$\mathcal{H}(P, Q) = -\sum_{\boldsymbol{w}} P(\boldsymbol{w}) \log(Q(\boldsymbol{w})) \approx -\frac{1}{R}\sum_{r=1}^{R} \log(Q(\boldsymbol{w}_{1:L_r}^{(r)}; \boldsymbol{\theta})) \triangleq \mathcal{L}(\mathcal{D}; \boldsymbol{\theta})$$

  - use empirical estimate since true distribution $P(\boldsymbol{w})$ unknown
- Use stochastic approximation to simplify parameter estimation
  - sample a minibatch of M sequences
  - update parameters
  - repeat till meet termination conditions
- Standard termination conditions
  - fixed number of iterations
  - increase in cross-entropy on held-out validation set

* Also called negative log-likelihood criterion

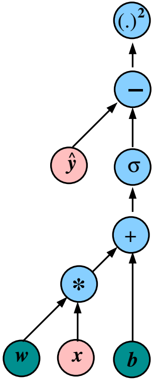
Back-Propagation Through Time (BPTT)



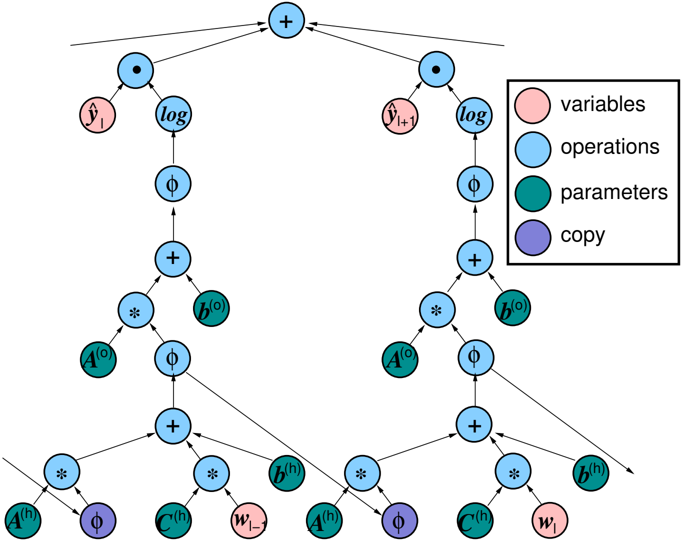
- Unroll recurrent unit to desired length
  - options available: 1-step, N-steps, L-steps where $N < L$
  - parameters are tied across time steps
- Compute gradient with respect to recurrent unit parameters
  - compile unrolled recurrent unit into a computational graph
  - use automatic differentiation to compute gradients
- Update recurrent unit and output distribution parameters
  - and repeat!

Anton Ragni

Automatic Differentiation: Computational Graph



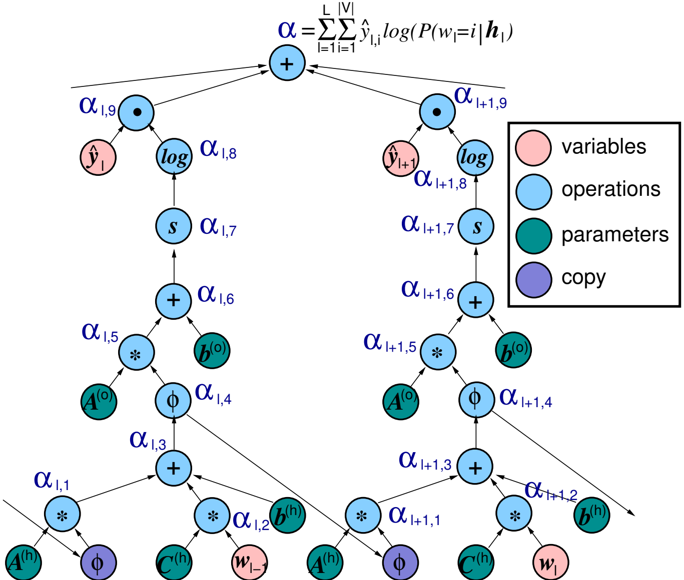
(a) Graph from past lecture



(b) Graph for simple recurrent unit

Anton Ragni

Automatic Differentiation: Forward Pass



- Compute values:

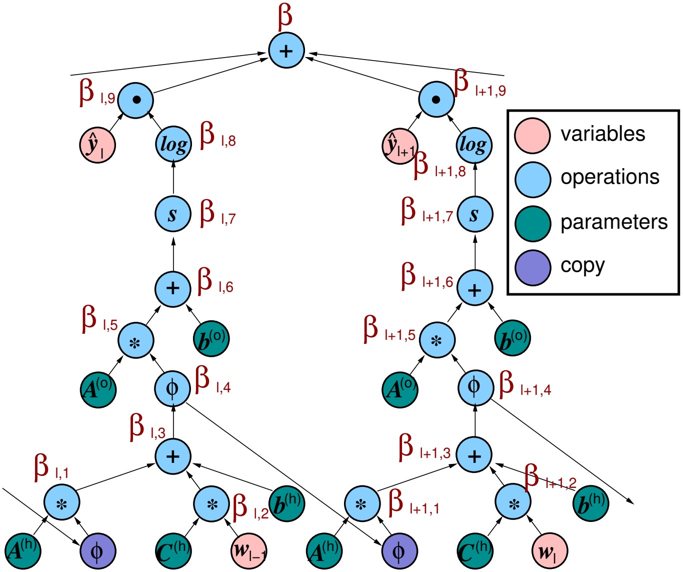
$$
\begin{aligned}
\boldsymbol{\alpha}_{l,1} &= \boldsymbol{A}^{(h)} * \boldsymbol{\alpha}_{l-1,4} \\
\boldsymbol{\alpha}_{l,2} &= \boldsymbol{C}^{(h)} * \boldsymbol{w}_{l-1} \\
\boldsymbol{\alpha}_{l,3} &= \boldsymbol{\alpha}_{l,1} + \boldsymbol{\alpha}_{l,2} + \boldsymbol{b}^{(h)} \\
\boldsymbol{\alpha}_{l,4} &= \phi(\boldsymbol{\alpha}_{l,3}) \\
\boldsymbol{\alpha}_{l,5} &= \boldsymbol{A}^{(o)} * \boldsymbol{\alpha}_{l,4} \\
\boldsymbol{\alpha}_{l,6} &= \boldsymbol{\alpha}_{l,5} + \boldsymbol{b}^{(o)} \\
\boldsymbol{\alpha}_{l,7} &= \boldsymbol{s}(\boldsymbol{\alpha}_{l,6}) \\
\boldsymbol{\alpha}_{l,8} &= \log(\boldsymbol{\alpha}_{l,7}) \\
\alpha_{l,9} &= \hat{\boldsymbol{y}}_l \cdot \boldsymbol{\alpha}_{l,8} \\
\alpha &= \alpha + \alpha_{l,9}
\end{aligned}
$$

where $\boldsymbol{s}$ is softmax, ϕ is sigmoid

$$
\hat{y}_{l,i} = \begin{cases} 1, & \text{if} \quad w^{(i)} = w_l \\ 0, & \text{otherwise} \end{cases}
$$

Anton Ragni

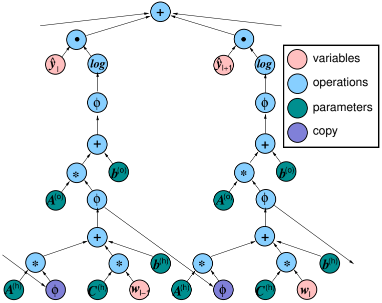
Automatic Differentiation: Backward Pass



- Compute derivatives:

$$
\begin{aligned}
\beta &= \frac{\partial \alpha}{\partial \alpha} \equiv 1 \\
\beta_{l,9} &= \beta \frac{\partial \alpha}{\partial \alpha_{l,9}} \\
\boldsymbol{\beta}_{l,8} &= \beta_{l,9} \frac{\partial \alpha_{l,9}}{\partial \boldsymbol{\alpha}_{l,8}} \\
\boldsymbol{\beta}_{l,7} &= \boldsymbol{\beta}_{l,8} \frac{\partial \boldsymbol{\alpha}_{l,8}}{\partial \boldsymbol{\alpha}_{l,7}} \\
\boldsymbol{\beta}_{l,6} &= \boldsymbol{\beta}_{l,7} \frac{\partial \boldsymbol{\alpha}_{l,7}}{\partial \boldsymbol{\alpha}_{l,6}} \\
\boldsymbol{\beta}_{l,5} &= \boldsymbol{\beta}_{l,6} \frac{\partial \boldsymbol{\alpha}_{l,6}}{\partial \boldsymbol{\alpha}_{l,5}} \\
\boldsymbol{\beta}_{l,4} &= \boldsymbol{\beta}_{l,5} \frac{\partial \boldsymbol{\alpha}_{l,5}}{\partial \boldsymbol{\alpha}_{l,4}} \\
\boldsymbol{\beta}_{l,3} &= \boldsymbol{\beta}_{l,4} \frac{\partial \boldsymbol{\alpha}_{l,4}}{\partial \boldsymbol{\alpha}_{l,3}} \\
\boldsymbol{\beta}_{l,2} &= \boldsymbol{\beta}_{l,3} \frac{\partial \boldsymbol{\alpha}_{l,3}}{\partial \boldsymbol{\alpha}_{l,2}} \\
\boldsymbol{\beta}_{l,1} &= \boldsymbol{\beta}_{l,3} \frac{\partial \boldsymbol{\alpha}_{l,3}}{\partial \boldsymbol{\alpha}_{l,1}}
\end{aligned}
$$

Automatic Differentiation: Parameter Update



- Accumulate derivatives:

$$\frac{\partial \alpha}{\partial \boldsymbol{A}^{(h)}} = \frac{\partial \alpha}{\partial \boldsymbol{A}^{(h)}} + \boldsymbol{\beta}_{l,1} \frac{\partial \boldsymbol{\alpha}_{l,1}}{\partial \boldsymbol{A}^{(h)}}$$

$$\frac{\partial \alpha}{\partial \boldsymbol{C}^{(h)}} = \frac{\partial \alpha}{\partial \boldsymbol{C}^{(h)}} + \boldsymbol{\beta}_{l,2} \frac{\partial \boldsymbol{\alpha}_{l,2}}{\partial \boldsymbol{C}^{(h)}}$$

$$\frac{\partial \alpha}{\partial \boldsymbol{b}^{(h)}} = \frac{\partial \alpha}{\partial \boldsymbol{b}^{(h)}} + \boldsymbol{\beta}_{l,3} \frac{\partial \boldsymbol{\alpha}_{l,3}}{\partial \boldsymbol{b}^{(h)}}$$

$$\frac{\partial \alpha}{\partial \boldsymbol{A}^{(o)}} = \frac{\partial \alpha}{\partial \boldsymbol{A}^{(o)}} + \boldsymbol{\beta}_{l,5} \frac{\partial \boldsymbol{\alpha}_{l,5}}{\partial \boldsymbol{A}^{(o)}}$$

$$\frac{\partial \alpha}{\partial \boldsymbol{b}^{(o)}} = \frac{\partial \alpha}{\partial \boldsymbol{b}^{(o)}} + \boldsymbol{\beta}_{l,6} \frac{\partial \boldsymbol{\alpha}_{l,6}}{\partial \boldsymbol{b}^{(o)}}$$

- Update parameters:

$$\hat{\boldsymbol{A}}^{(h)} = \boldsymbol{A}^{(h)} + \rho \frac{\partial \alpha}{\partial \boldsymbol{A}^{(h)}}$$

  - similar rules for other parameters
  - why derivative is added?

Anton Ragni

Mini-Batches

- Algebraic manipulations with high-dimensional data expensive
  - graphics processing units (GPU) provide support for efficient computation
  - BUT operations needs to be arranged efficiently
- Options available to construct mini-batches

<s>	all	you	get	is	risk	<s>	brokers	...	don't
deny	that	<s>	heating	oil	prices	rose	<s>	...	but
will	it	be	enough	<s>	the	lobby	says	...	<s>

(a) Cropping

<s>	how	interesting	<s>	ϵ	ϵ	ϵ
<s>	all	you	get	is	risk	<s>
<s>	don't	deny	that	<s>	ϵ	ϵ

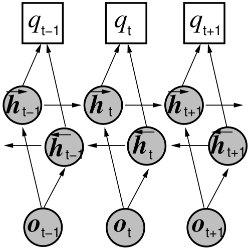
(b) Sentence bunching

<s>	how	interesting	<s>	all	you	...	need	that	<s>	ϵ
<s>	the	lobby	says	no	<s>	...	enough	<s>	ϵ	ϵ

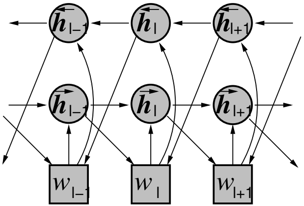
(c) Sentence splicing

  - only last two options yield a valid language model (why?)

Bi-Directional Recurrent Neural Networks (BiRNN)



(a) Acoustic model



(a) Language model

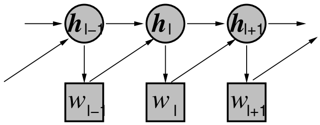
- Use information from future words to help predicting current words

$$P(\boldsymbol{w}_{1:L}) \triangleq \frac{1}{Z}\tilde{P}(\boldsymbol{w}_{1:L}) = \frac{1}{Z}\prod_{l=1}^{L} P(w_l|\boldsymbol{w}_{1:l-1}, \boldsymbol{w}_{l+1:L}) \approx \frac{1}{Z'}\prod_{l=1}^{L} P(w_l|\overrightarrow{\boldsymbol{h}}_{l-1}, \overleftarrow{\boldsymbol{h}}_{l+1})$$

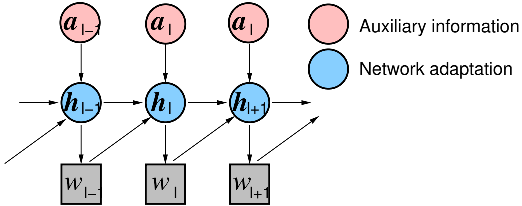
  - compensate for inadequate assumptions/approximations of unidirectional models
  - normalisation term $Z = \sum_{\boldsymbol{w}} \tilde{P}(\boldsymbol{w})$ cannot be efficiently computed (exc trivial cases)
- Performance assessed using pseudo-perplexity based on unnormalised probabilities
  - discuss possible issues
- Exercise: draw pseudo-DBNs for GRU and LSTM

Anton Ragni

Language Model Adaptation



(a) Standard language model



(b) Adaptation options

- Incorporate auxiliary information
  - topic and genre adaptation
  - BUT need a suitable representation
- Adjust parameters and/or activation functions
  - fine-tune model and/or activation parameters on adaptation data
  - need adaptation data
- Other options possible

Summary

- This lecture examined advanced language models
  - continuous word representations
  - full word history modelling
- Focus on recurrent forms of language models
  - recurrent units
  - parameter estimation
  - future words
  - adaptation
- Not examined in this lecture
  - convolutional language models

Anton Ragni