

RWTH AACHEN UNIVERSITY

Machine Learning – Lecture 17

Recurrent Neural Networks

21.01.2019

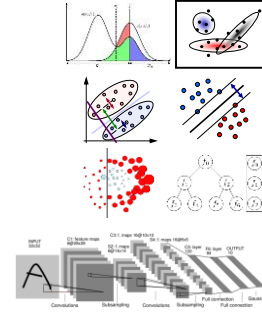
Bastian Leibe
RWTH Aachen
<http://www.vision.rwth-aachen.de>
leibe@vision.rwth-aachen.de

Machine Learning Winter '18

RWTH AACHEN UNIVERSITY

Course Outline

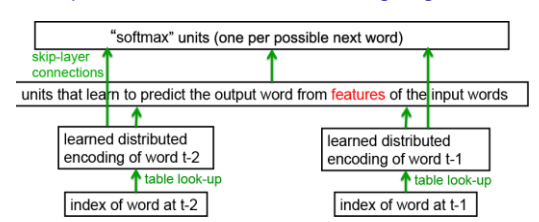
- Fundamentals
 - Bayes Decision Theory
 - Probability Density Estimation
- Classification Approaches
 - Linear Discriminants
 - Support Vector Machines
 - Ensemble Methods & Boosting
 - Random Forests
- Deep Learning
 - Foundations
 - Convolutional Neural Networks
 - Recurrent Neural Networks



B. Leibe

RWTH AACHEN UNIVERSITY

Recap: Neural Probabilistic Language Model



- Core idea
 - Learn a shared distributed encoding (word embedding) for the words in the vocabulary.

Y. Bengio, R. Ducharme, P. Vincent, C. Jauvin, [A Neural Probabilistic Language Model](#), In JMLR, Vol. 3, pp. 1137-1155, 2003.

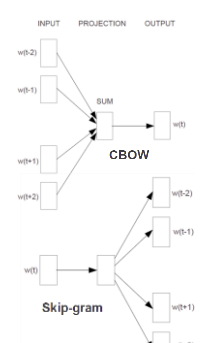
Slide adapted from Geoff Hinton

B. Leibe

RWTH AACHEN UNIVERSITY

Recap: word2vec

- Goal
 - Make it possible to learn high-quality word embeddings from huge data sets (billions of words in training set).
- Approach
 - Define two alternative learning tasks for learning the embedding:
 - “Continuous Bag of Words” (CBOW)
 - “Skip-gram”
 - Designed to require fewer parameters.



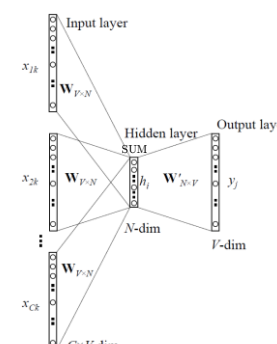
B. Leibe

RWTH AACHEN UNIVERSITY

Recap: word2vec CBOW Model

- Continuous BOW Model
 - Remove the non-linearity from the hidden layer
 - Share the projection layer for all words (their vectors are averaged)

⇒ Bag-of-Words model (order of the words does not matter anymore)

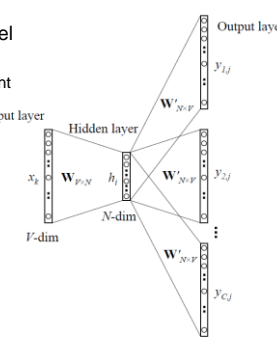


B. Leibe

RWTH AACHEN UNIVERSITY

Recap: word2vec Skip-Gram Model

- Continuous Skip-Gram Model
 - Similar structure to CBOW
 - Instead of predicting the current word, predict words within a certain range of the current word.
 - Give less weight to the more distant words



B. Leibe

Machine Learning Winter '18

Problems with 100k-1M outputs

- Weight matrix gets huge!
 - Example: CBOW model
 - One-hot encoding for inputs
 - Input-hidden connections are just vector lookups.
 - This is not the case for the hidden-output connections!
 - State h is not one-hot, and vocabulary size is 1M.
 - $\Rightarrow \mathbf{W}'_{N \times V}$ has $300 \times 1M$ entries
- Softmax gets expensive!
 - Need to compute normalization over 100k-1M outputs

B. Leibe Image source: Yin Bonn, 2016

Machine Learning Winter '18

Solution: Hierarchical Softmax

- Idea
 - Organize words in binary search tree, words are at leaves
 - Factorize probability of word w_0 as a product of node probabilities along the path.
 - Learn a linear decision function $y = v_{n(w,j)} \cdot h$ at each node to decide whether to proceed with left or right child node.
 - \Rightarrow Decision based on output vector of hidden units directly.

B. Leibe Image source: Yin Bonn, 2016

Machine Learning Winter '18

Topics of This Lecture

- Recurrent Neural Networks (RNNs)
 - Motivation
 - Intuition
- Learning with RNNs
 - Formalization
 - Comparison of Feedforward and Recurrent networks
 - Backpropagation through Time (BPTT)
- Problems with RNN Training
 - Vanishing Gradients
 - Exploding Gradients
 - Gradient Clipping

B. Leibe Image source: Yin Bonn, 2016

Machine Learning Winter '18

Recurrent Neural Networks

- Up to now
 - Simple neural network structure: 1-to-1 mapping of inputs to outputs
- This lecture: Recurrent Neural Networks
 - Generalize this to arbitrary mappings

B. Leibe Image source: Andrei Karpathy

Machine Learning Winter '18

Application: Part-of-Speech Tagging

Legend: Click the legend words to toggle highlighting. [Get help](#) on this page.

Noun Pronoun Verb Adjective Adverb Conjunction Preposition Article Interjection

Andrew and Maria thought their jobs were secure after the rancorous argument with the customer, but alas! Bad news is fast approaching them, especially after they viciously insulted the customer on social media.

B. Leibe Image source: http://wordnik.com

Machine Learning Winter '18

Application: Predicting the Next Word

T. Mikolov, M. Karafiat, L. Burget, J. Cernocky, S. Khudanpur, [Recurrent Neural Network Based Language Model](#), Interspeech 2010.

Slide credit: Andrei Karpathy, Fei-Fei Li Image source: Mikolov et al., 2010

RWTH AACHEN UNIVERSITY

Application: Machine Translation

I. Sutskever, O. Vinyals, Q. Le, [Sequence to Sequence Learning with Neural Networks](#), NIPS 2014.

13

RWTH AACHEN UNIVERSITY

RNNs: Intuition

- Example: Language modeling
 - Suppose we had the training sequence "cat sat on mat"
 - We want to train a language model

$$p(\text{next word} \mid \text{previous words})$$
 - First assume we only have a finite, 1-word history.
 - $p(\text{cat} \mid \langle S \rangle)$
 - $p(\text{sat} \mid \text{cat})$
 - $p(\text{on} \mid \text{sat})$
 - $p(\text{mat} \mid \text{on})$
 - $p(\langle E \rangle \mid \text{mat})$

< S > and < E > are start and end tokens.

14

RWTH AACHEN UNIVERSITY

RNNs: Intuition

- Vanilla 2-layer classification net

Hidden layer (e.g., 500D vectors)
 $h_i = \max\{0, W_{xh}x_i\}$

Word embedding (300D vector for each word)

15

RWTH AACHEN UNIVERSITY

RNNs: Intuition

- Turning this into an RNN (wait for it...)

Hidden layer (e.g., 500D vectors)
 $h_i = \max\{0, W_{xh}x_i\}$

Word embedding (300D vector for each word)

16

RWTH AACHEN UNIVERSITY

RNNs: Intuition

- Turning this into an RNN (done!)

Hidden layer (e.g., 500D vectors)
 $h_i = \max\{0, W_{xh}x_i + W_{hh}h_{i-1}\}$

Word embedding (300D vector for each word)

17

RWTH AACHEN UNIVERSITY

RNNs: Intuition

- Training this on a lot of sentences would give us a language model.
- I.e., a way to predict

$$p(\text{next word} \mid \text{previous words})$$

18

Machine Learning Winter '18

RWTH AACHEN UNIVERSITY

RNNs: Intuition

- Training this on a lot of sentences would give us a language model.
- I.e., a way to predict $p(\text{next word} \mid \text{previous words})$

Slide credit: Andrei Karpathy, Fei-Fei Li

B. Leibe

19

Machine Learning Winter '18

RWTH AACHEN UNIVERSITY

RNNs: Intuition

- Training this on a lot of sentences would give us a language model.
- I.e., a way to predict $p(\text{next word} \mid \text{previous words})$

Slide credit: Andrei Karpathy, Fei-Fei Li

B. Leibe

20

Machine Learning Winter '18

RWTH AACHEN UNIVERSITY

RNNs: Intuition

- Training this on a lot of sentences would give us a language model.
- I.e., a way to predict $p(\text{next word} \mid \text{previous words})$

Slide credit: Andrei Karpathy, Fei-Fei Li

B. Leibe

21

Machine Learning Winter '18

RWTH AACHEN UNIVERSITY

RNNs: Intuition

- Training this on a lot of sentences would give us a language model.
- I.e., a way to predict $p(\text{next word} \mid \text{previous words})$

Slide credit: Andrei Karpathy, Fei-Fei Li

B. Leibe

22

Machine Learning Winter '18

RWTH AACHEN UNIVERSITY

RNNs: Intuition

- Training this on a lot of sentences would give us a language model.
- I.e., a way to predict $p(\text{next word} \mid \text{previous words})$

Slide credit: Andrei Karpathy, Fei-Fei Li

B. Leibe

23

Machine Learning Winter '18

RWTH AACHEN UNIVERSITY

RNNs: Intuition

- Training this on a lot of sentences would give us a language model.
- I.e., a way to predict $p(\text{next word} \mid \text{previous words})$

Slide credit: Andrei Karpathy, Fei-Fei Li

B. Leibe

24

Machine Learning Winter '18

RNNs: Intuition

- Training this on a lot of sentences would give us a language model.
- I.e., a way to predict $p(\text{next word} \mid \text{previous words})$

Slide credit: Andrej Karpathy, Fei-Fei Li, B. Leibe

Machine Learning Winter '18

RNNs: Intuition

- Training this on a lot of sentences would give us a language model.
- I.e., a way to predict $p(\text{next word} \mid \text{previous words})$

Slide credit: Andrej Karpathy, Fei-Fei Li, B. Leibe

Machine Learning Winter '18

Topics of This Lecture

- Recurrent Neural Networks (RNNs)
 - Motivation
 - Intuition
- Learning with RNNs
 - Formalization
 - Comparison of Feedforward and Recurrent networks
 - Backpropagation through Time (BPTT)
- Problems with RNN Training
 - Vanishing Gradients
 - Exploding Gradients
 - Gradient Clipping

B. Leibe 27

Machine Learning Winter '18

RNNs: Introduction

- RNNs are regular NNs whose hidden units have additional forward connections over time.
 - You can unroll them to create a network that extends over time.
 - When you do this, keep in mind that the weights for the hidden units are shared between temporal layers.

B. Leibe 28

Machine Learning Winter '18

RNNs: Introduction

- RNNs are very powerful, because they combine two properties:
 - Distributed hidden state that allows them to store a lot of information about the past efficiently.
 - Non-linear dynamics that allows them to update their hidden state in complicated ways.
- With enough neurons and time, RNNs can compute anything that can be computed by your computer.

Slide credit: Geoff Hinton, B. Leibe, Image source: Andrej Karpathy

Machine Learning Winter '18

Feedforward Nets vs. Recurrent Nets

- Imagine a feedforward network
 - Assume there is a time delay of 1 in using each connection.
 - ⇒ This is very similar to how an RNN works.
 - Only change: the layers share their weights.

⇒ The recurrent net is just a feedforward net that keeps reusing the same weights.

B. Leibe 30

RWTH AACHEN UNIVERSITY

Backpropagation with Weight Constraints

- It is easy to modify the backprop algorithm to incorporate linear weight constraints
 - To constrain $w_1 = w_2$, we start with the same initialization and then make sure that the gradients are the same:

$$\nabla w_1 = \nabla w_2$$
 - We compute the gradients as usual and then use

$$\frac{\partial E}{\partial w_1} + \frac{\partial E}{\partial w_2}$$
 for both w_1 and w_2 .

Machine Learning Winter '18 31

RWTH AACHEN UNIVERSITY

Backpropagation Through Time (BPTT)

- Formalization
 - Inputs x_t
 - Outputs y_t
 - Hidden units h_t
 - Initial state h_0
 - Connection matrices
 - W_{xh}
 - W_{hy}
 - W_{hh}
 - Configuration

$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b)$$

$$\hat{y}_t = \text{softmax}(W_{hy}h_t)$$

Machine Learning Winter '18 32

RWTH AACHEN UNIVERSITY

Recap: Backpropagation Algorithm

$$\frac{\partial E}{\partial z_j^{(k)}} = \frac{\partial y_j^{(k)}}{\partial z_j^{(k)}} \frac{\partial E}{\partial y_j^{(k)}} = \frac{\partial g(z_j^{(k)})}{\partial z_j^{(k)}} \frac{\partial E}{\partial y_j^{(k)}}$$

$$\frac{\partial E}{\partial y_i^{(k-1)}} = \sum_j \frac{\partial z_j^{(k)}}{\partial y_i^{(k-1)}} \frac{\partial E}{\partial z_j^{(k)}} = \sum_j w_{ji}^{(k-1)} \frac{\partial E}{\partial z_j^{(k)}}$$

$$\frac{\partial E}{\partial w_{ji}^{(k-1)}} = \frac{\partial z_j^{(k)}}{\partial w_{ji}^{(k-1)}} \frac{\partial E}{\partial z_j^{(k)}} = y_i^{(k-1)} \frac{\partial E}{\partial z_j^{(k)}}$$

- Efficient propagation scheme
 - $y_i^{(k-1)}$ is already known from forward pass! (Dynamic Programming)
 - ⇒ Propagate back the gradient from layer k and multiply with $y_i^{(k-1)}$.

Machine Learning Winter '18 33

RWTH AACHEN UNIVERSITY

Backpropagation Through Time (BPTT)

- Error function
 - Computed over all time steps: $E = \sum_{1 \leq t \leq T} E_t$

Machine Learning Winter '18 34

RWTH AACHEN UNIVERSITY

Backpropagation Through Time (BPTT)

- Backpropagated gradient
 - For weight w_{ij} : $\frac{\partial E_t}{\partial w_{ij}} = \frac{\partial E_t}{\partial h_t} \frac{\partial h_t}{\partial w_{ij}}$

Machine Learning Winter '18 35

RWTH AACHEN UNIVERSITY

Backpropagation Through Time (BPTT)

- Backpropagated gradient
 - For weight w_{ij} : $\frac{\partial E_t}{\partial w_{ij}} = \frac{\partial E_t}{\partial h_t} \frac{\partial h_t}{\partial w_{ij}} + \frac{\partial E_t}{\partial h_t} \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial w_{ij}}$

Machine Learning Winter '18 36

RWTH AACHEN UNIVERSITY

Backpropagation Through Time (BPTT)

- Backpropagated gradient
 - For weight w_{ij} :
$$\frac{\partial E_t}{\partial w_{ij}} = \frac{\partial E_t}{\partial \mathbf{h}_t} \frac{\partial \mathbf{h}_t}{\partial w_{ij}} + \frac{\partial E_t}{\partial \mathbf{h}_t} \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_{t-1}} \frac{\partial \mathbf{h}_{t-1}}{\partial w_{ij}} + \dots$$
 - In general:
$$\frac{\partial E_t}{\partial w_{ij}} = \sum_{1 \leq k \leq t} \left(\frac{\partial E_t}{\partial \mathbf{h}_t} \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_k} \frac{\partial^+ \mathbf{h}_k}{\partial w_{ij}} \right)$$

37

RWTH AACHEN UNIVERSITY

Backpropagation Through Time (BPTT)

- Analyzing the terms
 - For weight w_{ij} :
$$\frac{\partial E_t}{\partial w_{ij}} = \sum_{1 \leq k \leq t} \left(\frac{\partial E_t}{\partial \mathbf{h}_t} \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_k} \frac{\partial^+ \mathbf{h}_k}{\partial w_{ij}} \right)$$
 - This is the "immediate" partial derivative (with \mathbf{h}_{k-1} as constant)

38

RWTH AACHEN UNIVERSITY

Backpropagation Through Time (BPTT)

- Analyzing the terms
 - For weight w_{ij} :
$$\frac{\partial E_t}{\partial w_{ij}} = \sum_{1 \leq k \leq t} \left(\frac{\partial E_t}{\partial \mathbf{h}_t} \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_k} \frac{\partial^+ \mathbf{h}_k}{\partial w_{ij}} \right)$$
 - Propagation term:
$$\frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_k} = \prod_{t \geq i > k} \frac{\partial \mathbf{h}_i}{\partial \mathbf{h}_{i-1}}$$

39

RWTH AACHEN UNIVERSITY

Backpropagation Through Time (BPTT)

- Summary
 - Backpropagation equations

$$E = \sum_{1 \leq t \leq T} E_t$$

$$\frac{\partial E_t}{\partial w_{ij}} = \sum_{1 \leq k \leq t} \left(\frac{\partial E_t}{\partial \mathbf{h}_t} \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_k} \frac{\partial^+ \mathbf{h}_k}{\partial w_{ij}} \right)$$

$$\frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_k} = \prod_{t \geq i > k} \frac{\partial \mathbf{h}_i}{\partial \mathbf{h}_{i-1}} = \prod_{t \geq i > k} \mathbf{W}_{hh}^T \text{diag}(\sigma'(\mathbf{h}_{i-1}))$$
 - Remaining issue: how to set the initial state \mathbf{h}_0 ?
 - ⇒ Learn this together with all the other parameters.

40

RWTH AACHEN UNIVERSITY

Topics of This Lecture

- Recurrent Neural Networks (RNNs)
 - Motivation
 - Intuition
- Learning with RNNs
 - Formalization
 - Comparison of Feedforward and Recurrent networks
 - Backpropagation through Time (BPTT)
- Problems with RNN Training
 - Vanishing Gradients
 - Exploding Gradients
 - Gradient Clipping

41

RWTH AACHEN UNIVERSITY

Problems with RNN Training

- Training RNNs is very hard
 - As we backpropagate through the layers, the magnitude of the gradient may grow or shrink exponentially
 - ⇒ Exploding or vanishing gradient problem!
 - In an RNN trained on long sequences (e.g., 100 time steps) the gradients can easily explode or vanish.
 - Even with good initial weights, it is very hard to detect that the current target output depends on an input from many time-steps ago.

42

RWTH AACHEN UNIVERSITY

Exploding / Vanishing Gradient Problem

- Consider the propagation equations:

$$\frac{\partial E_t}{\partial w_{ij}} = \sum_{1 \leq k \leq t} \left(\frac{\partial E_t}{\partial h_t} \frac{\partial h_t}{\partial h_k} \frac{\partial^+ h_k}{\partial w_{ij}} \right)$$

$$\frac{\partial h_t}{\partial h_k} = \prod_{t \geq i > k} \frac{\partial h_i}{\partial h_{i-1}} = \prod_{t \geq i > k} \mathbf{W}_{hh}^\top \text{diag}(\sigma'(\mathbf{h}_{i-1}))$$

$$= (\mathbf{W}_{hh}^\top)^l$$
 - if t goes to infinity and $l = t - k$.
- ⇒ We are effectively taking the weight matrix to a high power.
- The result will depend on the eigenvalues of \mathbf{W}_{hh} .
 - Largest eigenvalue > 1 ⇒ Gradients *may* explode.
 - Largest eigenvalue < 1 ⇒ Gradients *will* vanish.
 - This is very bad...

43

RWTH AACHEN UNIVERSITY

Why Is This Bad?

- Vanishing gradients in language modeling
 - Words from time steps far away are not taken into consideration when training to predict the next word.
- Example:
 - „Jane walked into the room. John walked in too. It was late in the day. Jane said hi to ____“
 - ⇒ The RNN will have a hard time learning such long-range dependencies.

44

RWTH AACHEN UNIVERSITY

Gradient Clipping

- Trick to handle exploding gradients
 - If the gradient is larger than a threshold, clip it to that threshold.

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

```

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

```

- This makes a big difference in RNNs

45

RWTH AACHEN UNIVERSITY

Gradient Clipping Intuition

- Example
 - Error surface of a single RNN neuron
 - High curvature walls
 - Solid lines: standard gradient descent trajectories
 - Dashed lines: gradients rescaled to fixed size

46

RWTH AACHEN UNIVERSITY

Handling Vanishing Gradients

- Vanishing Gradients are a harder problem
 - They severely restrict the dependencies the RNN can learn.
 - The problem gets more severe the deeper the network is.
 - It can be very hard to diagnose that Vanishing Gradients occur (you just see that learning gets stuck).
- Ways around the problem
 - Glorot/He initialization (more on that in Lecture 21)
 - ReLU
 - More complex hidden units (LSTM, GRU)

47

RWTH AACHEN UNIVERSITY

ReLU to the Rescue

- Idea
 - Initialize \mathbf{W}_{hh} to identity matrix
 - Use Rectified Linear Units (ReLU)

$$g(a) = \max\{0, a\}$$
- Effect
 - The gradient is propagated with a constant factor

$$\frac{\partial g(a)}{\partial a} = \begin{cases} 1, & a > 0 \\ 0, & \text{else} \end{cases}$$
 - ⇒ Huge difference in practice!

48

References and Further Reading

- RNNs
 - R. Pascanu, T. Mikolov, Y. Bengio, [On the difficulty of training recurrent neural networks](#), JMLR, Vol. 28, 2013.
 - A. Karpathy, [The Unreasonable Effectiveness of Recurrent Neural Networks](#), blog post, May 2015.