Rekurrente Neuronale Netze

Gregor Mitscha-Baude

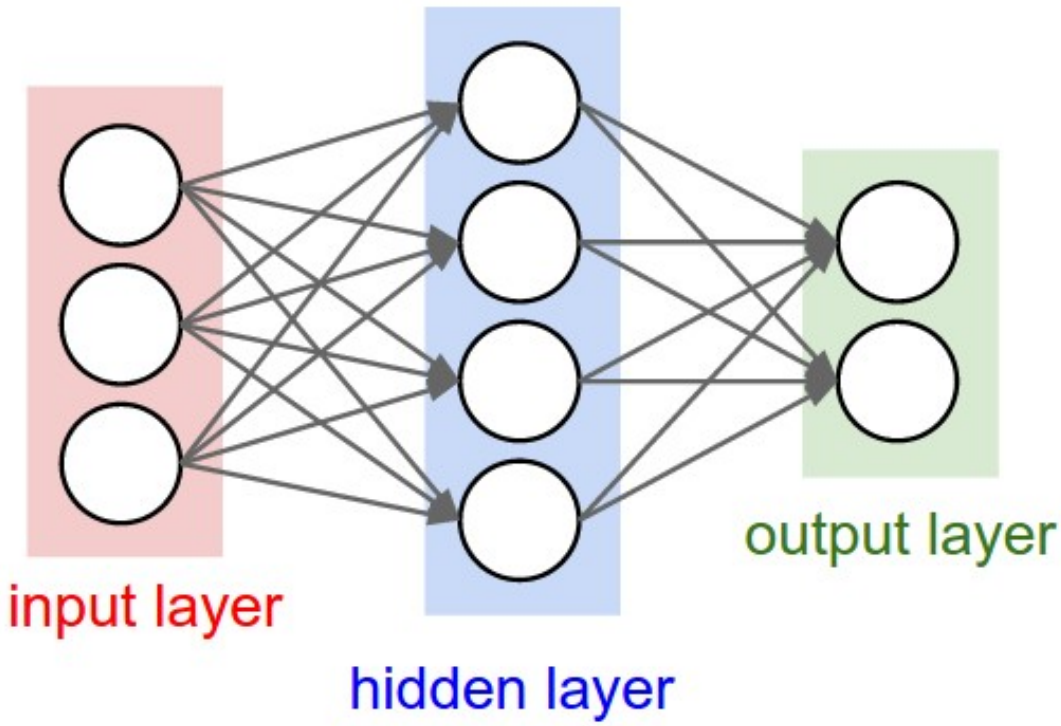
May 9, 2016

Rekurrente Neuronale Netze

Gregor Mitscha-Baude

Motivation

- Standard neuronales Netz:



- Fixe Dimensionen von Input und Output!

Motivation

- In viele Anwendungen variable Input/Output-Länge.
  - Spracherkennung
  - Maschinenübersetzung
  - Image captioning

Motivation

- In viele Anwendungen variable Input/Output-Länge.
  - Spracherkennung
  - Maschinenübersetzung
  - Image captioning
- allgemein: Textgenerierung
- Musik, Videos, Aktienkurse, ...

Motivation

- In viele Anwendungen variable Input/Output-Länge.
  - Spracherkennung
  - Maschinenübersetzung
  - Image captioning
- allgemein: Textgenerierung
- Musik, Videos, Aktienkurse, ...
- menschliches Gehirn

Rekurrente Neurale Netze (RNNs)

- RNNs operieren auf Sequenz von Inputs.
- RNNs modellieren zeitliche Abhängigkeiten.

Übersicht

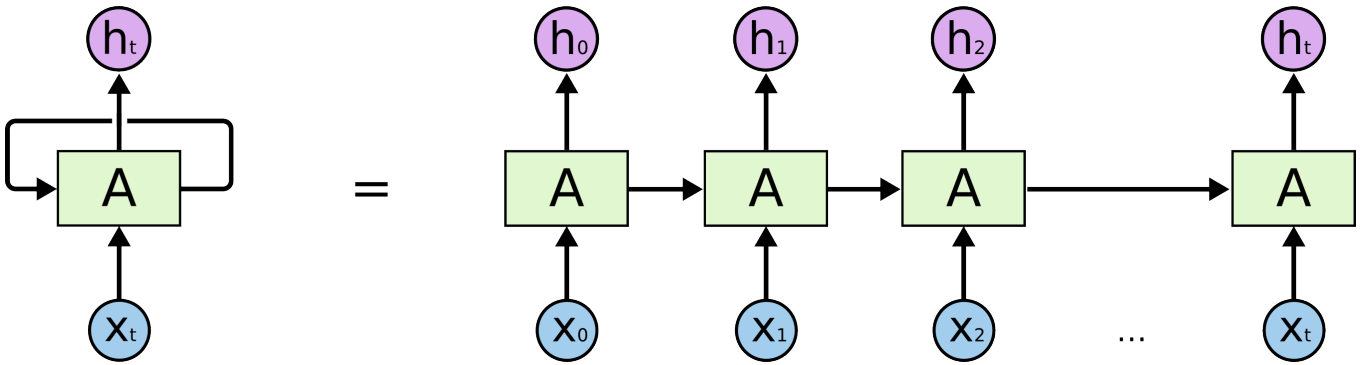
- RNNs Basics
- Moderne RNNs: LSTM
- Aktuelle Forschung zu RNNs: Attention

Section 1

Wie funktionieren RNNs? – Die Basics

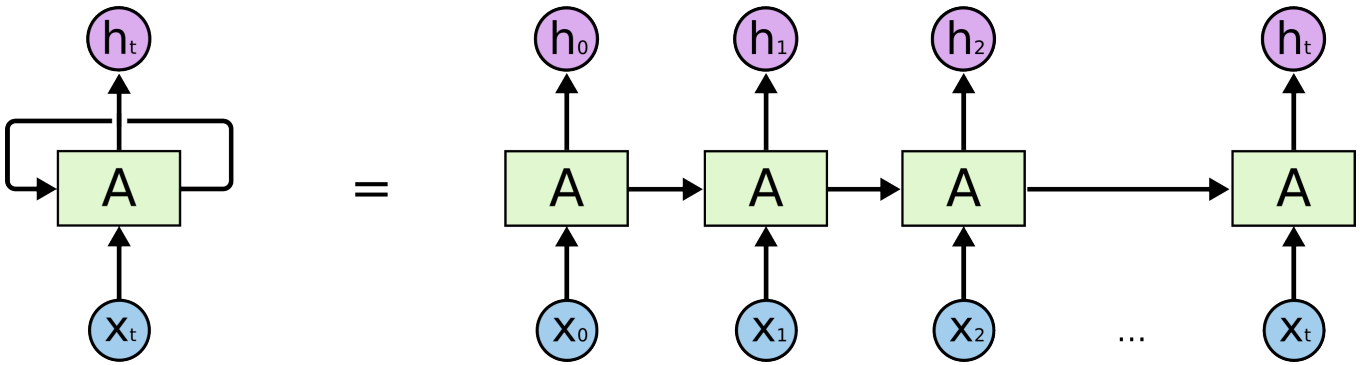
Rekurrente Neurale Netze (RNNs)

- Hidden Layer haben Verbindung zu sich selbst.



Rekurrente Neurale Netze (RNNs)

- Hidden Layer haben Verbindung zu sich selbst.

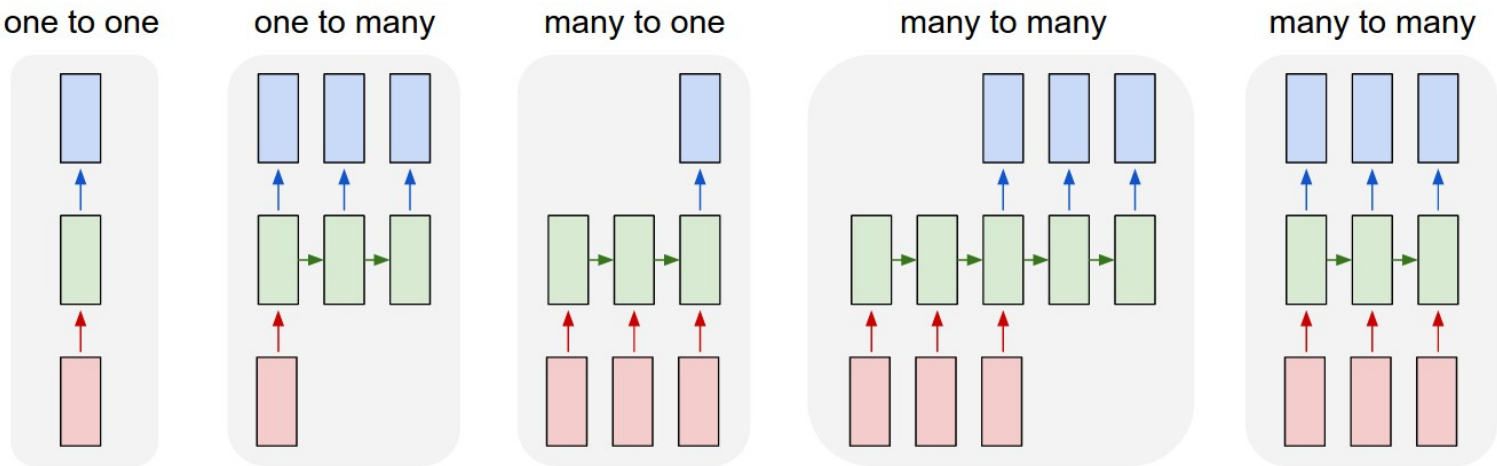


- können eigenen *Zustand (state)* weitergeben
- weight tying entlang der zeitlichen Dimension

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

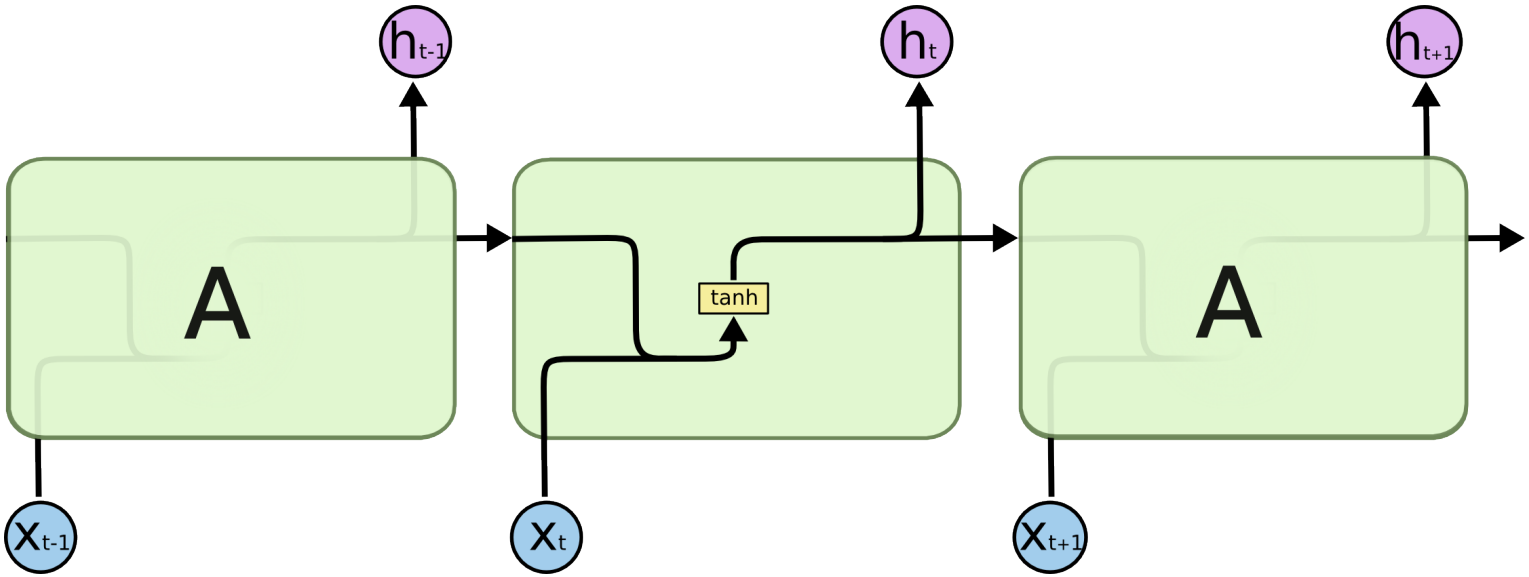
Rekurrente Neurale Netze (RNNs)

- Verschiedene Architekturen sind möglich:



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

Plain Vanilla RNN



- hier nur ein Hidden Layer
- wird in der Praxis nicht verwendet

Mathematische Formulierung

- Plain vanilla RNN

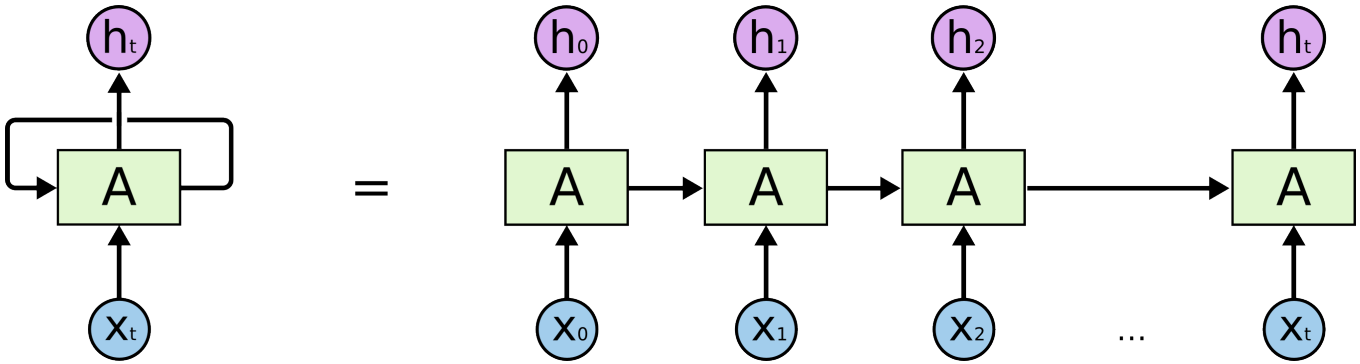
$$h_t = \tanh(W_x x_t + W_h h_{t-1} + b)$$

- Lernen = Optimierung

$$\min_{W,b} \sum_t C(h_t)$$

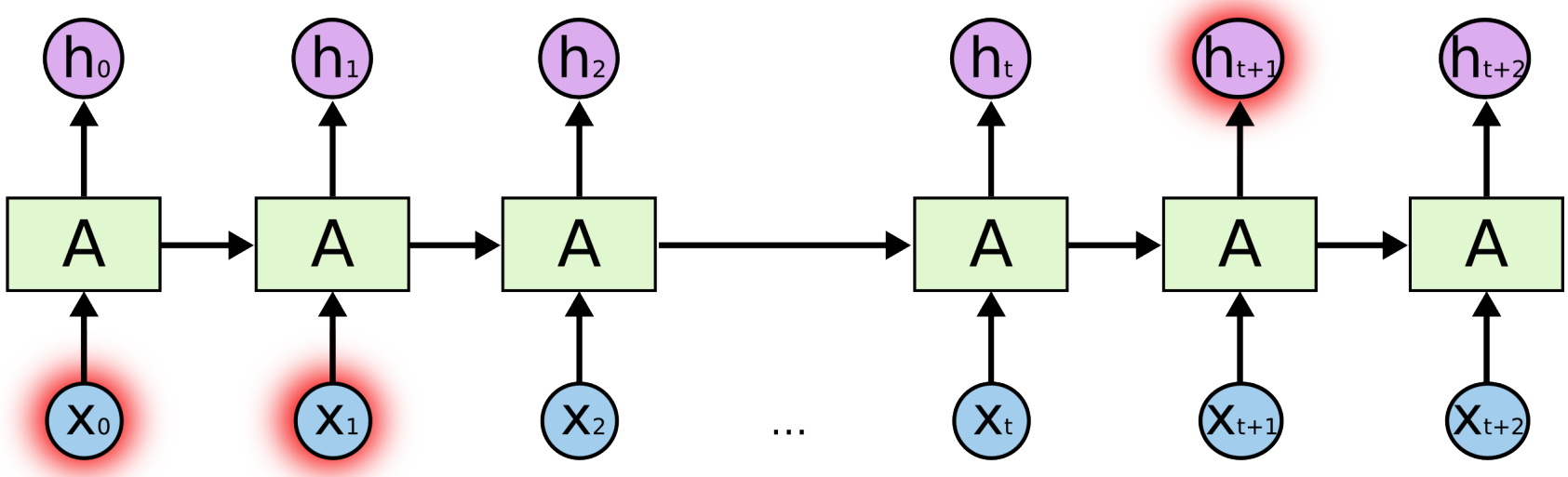
  - C ... Kostenfunktion

Lernen: Backpropagation Through Time



- Output-Kosten generieren Gradienten
- Gradient fließt gegen Pfeilrichtung
- Backpropagation *rückwärts in der Zeit*

RNNs sind immer deep



- Selbes Problem wie bei allen Deep Nets
  - exponentiell verschwindender Gradient
- $\Rightarrow$ *Langzeitabhängigkeiten* sind schwer zu lernen

Section 2

LSTM

LSTM

- h_t in vanilla RNNs ist schlechtes Kurzzeitgedächtnis
  - wird immer mit Gewichten multipliziert
  - durchläuft Nichtlinearität (tanh)
  - "vergisst" dadurch vergangene Schritte schnell

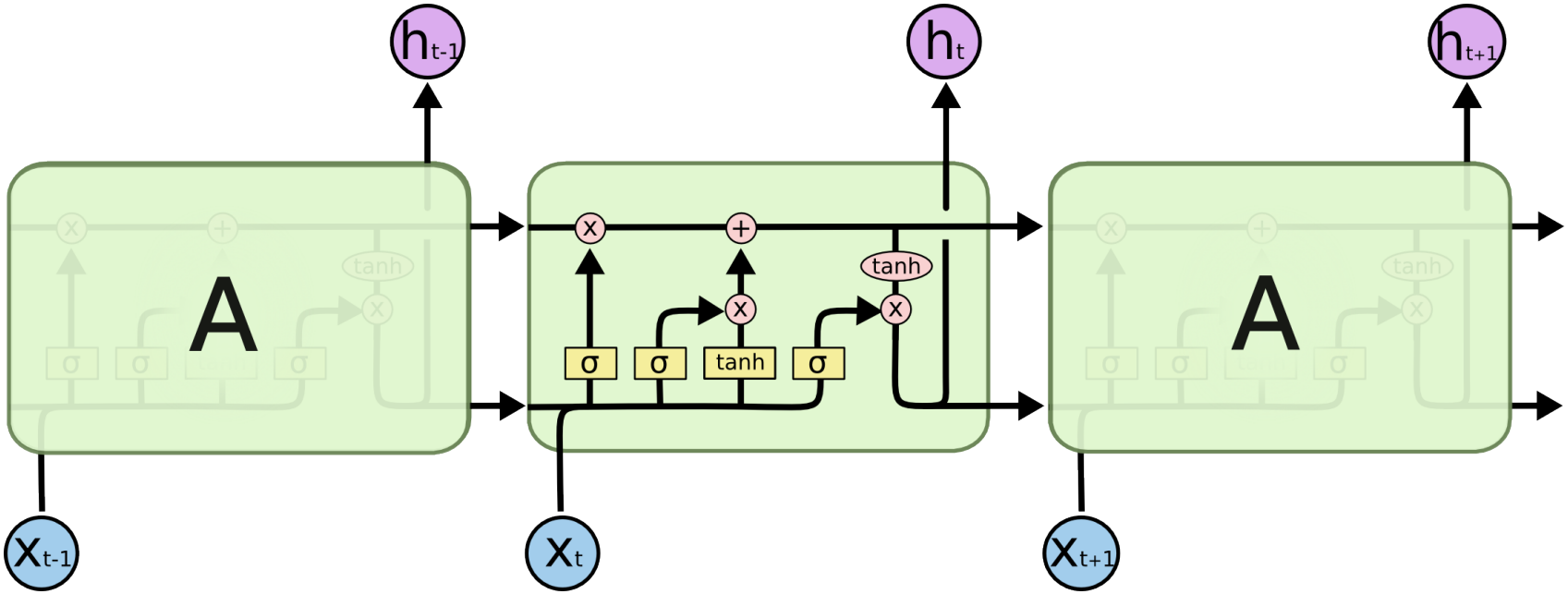
LSTM

- h_t in vanilla RNNs ist schlechtes Kurzzeitgedächtnis
  - wird immer mit Gewichten multipliziert
  - durchläuft Nichtlinearität (tanh)
  - "vergisst" dadurch vergangene Schritte schnell
- Lösung: wir brauchen längeres Kurzzeitgedächtnis.
- LSTM (Long Short-Term Memory):
  Hochreiter, Schmidhuber '97

LSTM

- h_t in vanilla RNNs ist schlechtes Kurzzeitgedächtnis
  - wird immer mit Gewichten multipliziert
  - durchläuft Nichtlinearität (tanh)
  - "vergisst" dadurch vergangene Schritte schnell
- Lösung: wir brauchen längeres Kurzzeitgedächtnis.
- LSTM (Long Short-Term Memory):
  Hochreiter, Schmidhuber '97
  - memory cell C_t
  - durchläuft keine Nichtlinearität

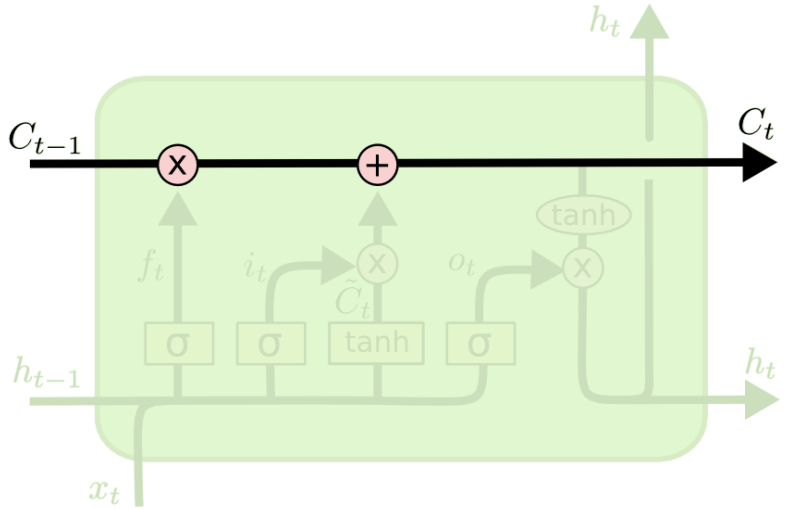
LSTM



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

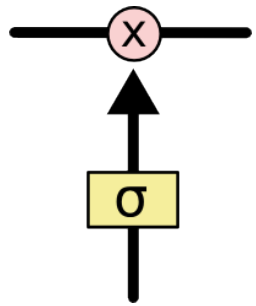
Memory cell

- Zustand, “Gedächtnis”



- durchläuft multiplikatives *Gate* und einfache Addition

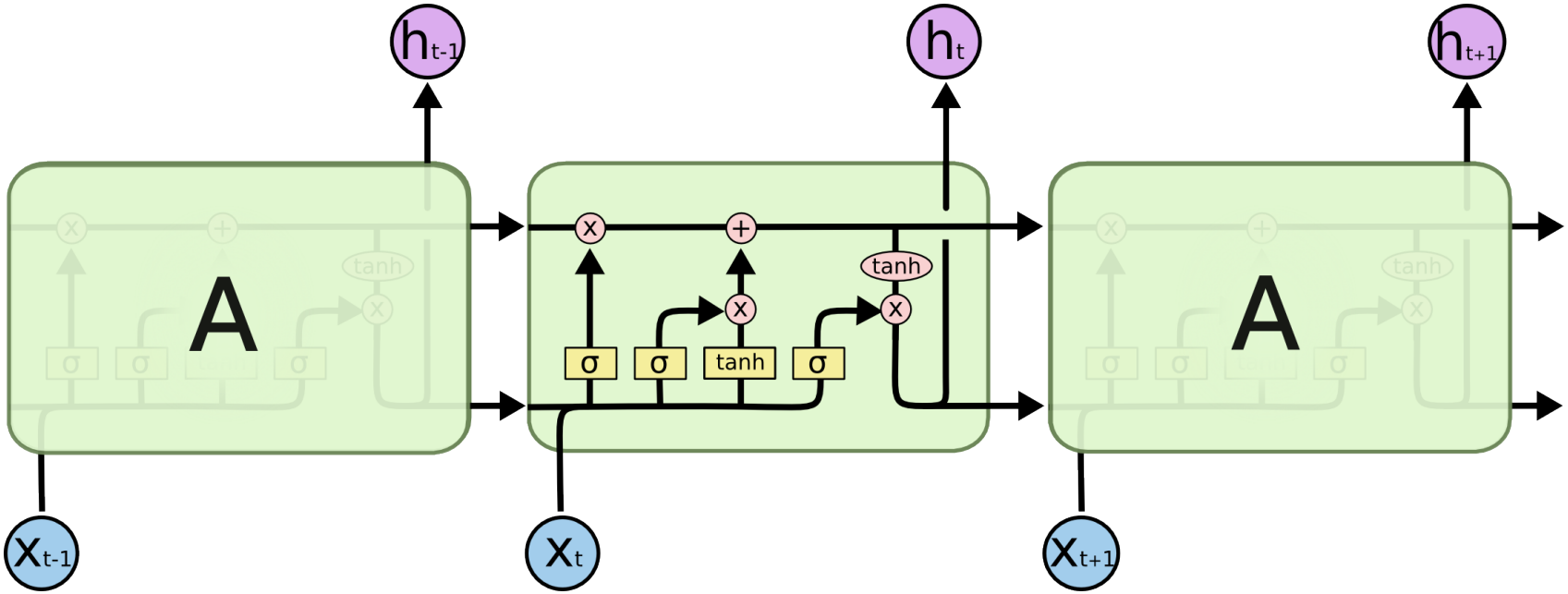
Gate



$$c = \sigma(x) * c$$

- $\sigma(x)$ ist Sigmoid
  - nahe 1 $\Rightarrow$ c fließt durch (offenes Gate)
  - nahe 0 $\Rightarrow$ c wird ausgelöscht (geschlossenes Gate)

LSTM hat drei Gates



- forget gate
- input gate
- output gate

LSTM ist der Standard

- funktioniert besser als vanilla RNN
- in der Praxis mehrere LSTMs übereinander gestapelt
- Neu: Vereinfachungen von LSTM, die ähnlich effektiv sind
  - GRU (Gated Recurrent Unit; Cho et al. '14)
  - DSGU (Deep Simple Gated Unit; Gao, Glowacka '16)

LSTM in Androids Spracherkennung

- 2012: Android setzt ConvNets in Spracherkennung ein
  - erste große industrielle Anwendung von Deep Learning
- 2014: Google Research zeigt Vorteile von LSTM vs. ConvNets (Sak et al. '14)
- September 2015: Android wechselt auf LSTM RNNs

Section 3

Memory/Attention – RNNs in der aktuellen Forschung

RNNs in der aktuellen Forschung

- Memory/Attention
- Recurrent Batch Normalization
- Rekurrente Autoencoder
- Adaptive Computation Time
- ...

Memory/Attention

- Motivation: komplexe Algorithmen brauchen Speicher. Ist in LSTM implizit vorhanden, aber Speichergröße an den Rechenaufwand pro Schritt geknüpft:

$$\#\ \text{Weights} \simeq |C_t|^2$$

$\Rightarrow$ *Ungeeignet für großen Speicher.*
$\Rightarrow$ Brauchen besseres Speichermodell / Adress-Mechanismus.

Memory/Attention

- Motivation: komplexe Algorithmen brauchen Speicher. Ist in LSTM implizit vorhanden, aber Speichergröße an den Rechenaufwand pro Schritt geknüpft:

$$\#\ \text{Weights} \simeq |C_t|^2$$

  $\Rightarrow$ *Ungeeignet für großen Speicher.*
  $\Rightarrow$ Brauchen besseres Speichermodell / Adress-Mechanismus.

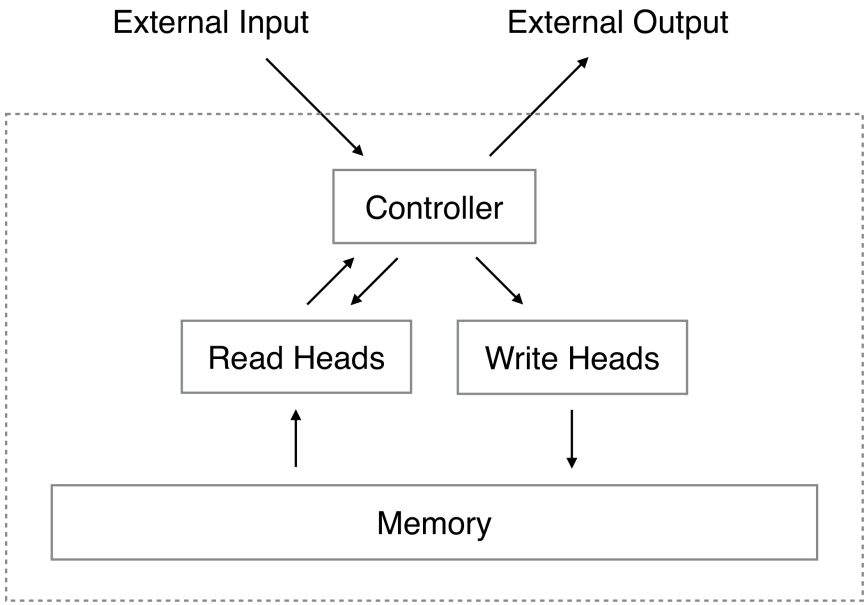
- Verwandte Idee: RNN soll Aufmerksamkeit auf kleinen Teil des Inputs lenken können: zB Detail in Bild, oder bestimmte Fakten in Wikipedia.
  $\Rightarrow$ Attention, benötigt auch Adress-Mechanismus.

Memory/Attention

Drei wichtige Paper fast zeitgleich:

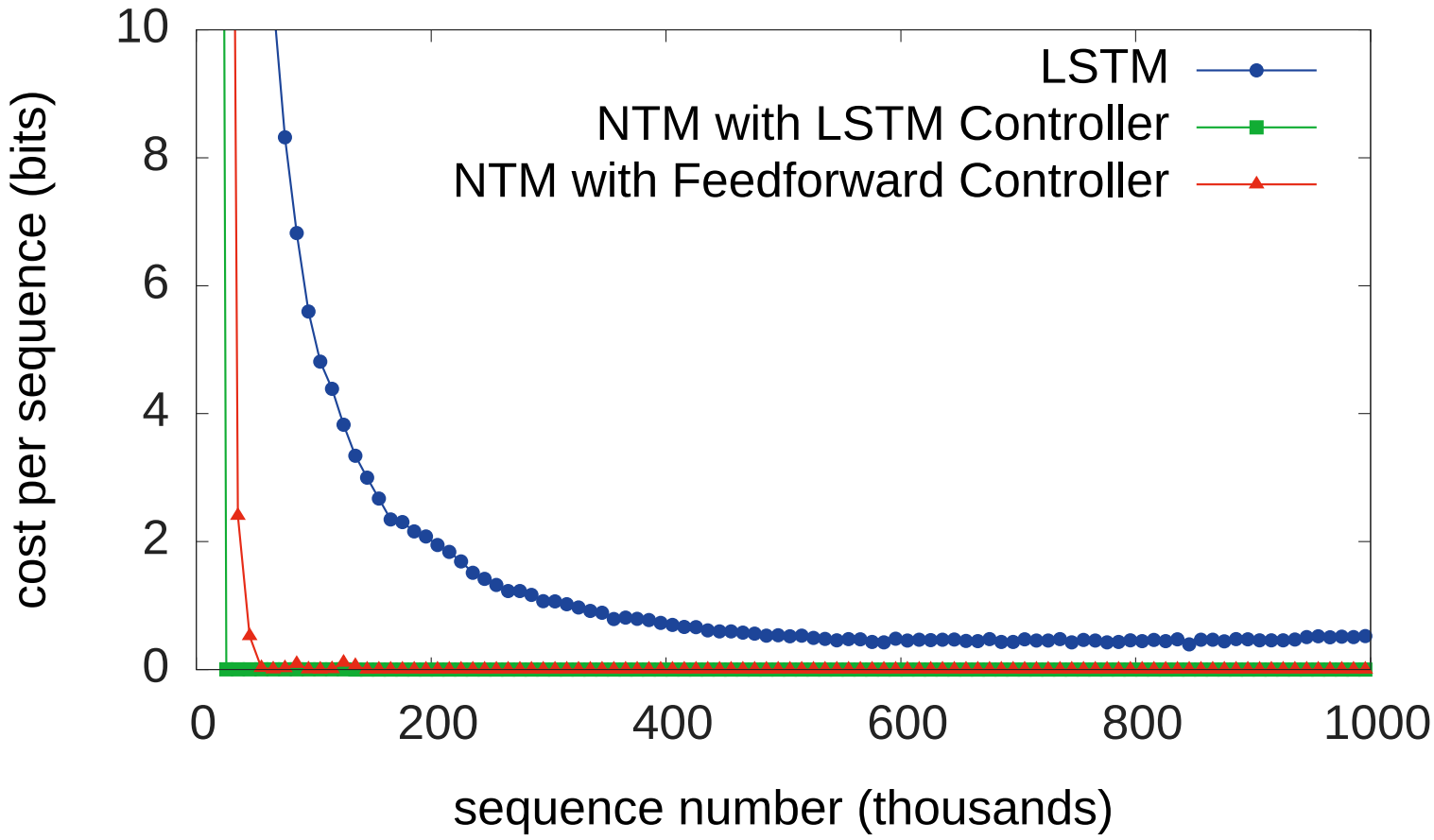
- Bahdanau et al., Sep 2014: NMT mit Attention
- Weston et al., Okt 2014: *Memory Networks*
- Graves et al., Okt 2014: *Neural Turing Machines*

Neural Turing Machines (Graves et. al '14)

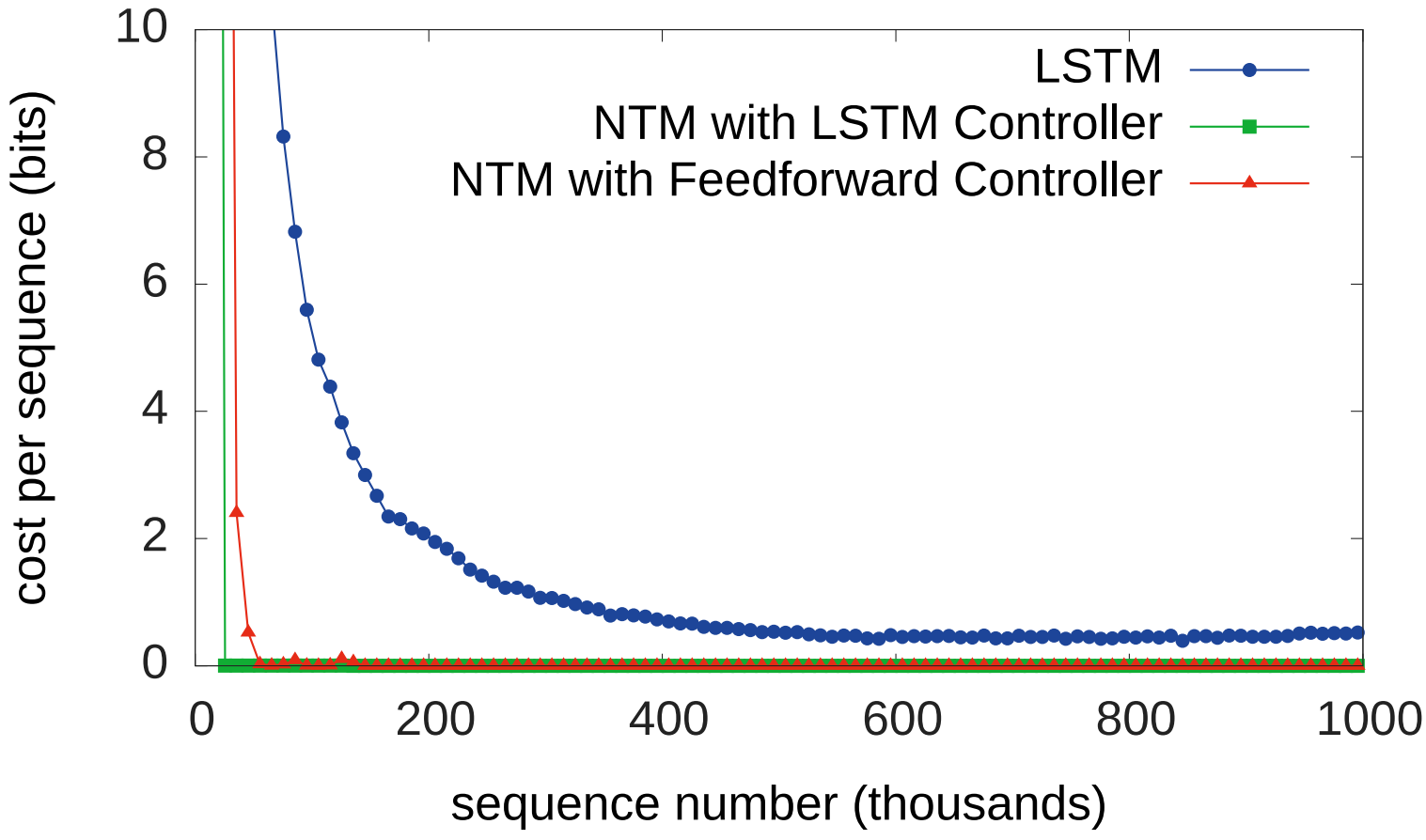


- NN mit Architektur einer Turing-Maschine
  - "differenzierbarer Computer"
- Controller ist LSTM oder Feedforward Netz
- Read/Write Heads produzieren Gewichtungen über N Speicherblocks
  - Parameteranzahl ist *unabhängig von N*

Neural Turing Machines – Copy Task



Neural Turing Machines – Copy Task



- Nicht gezeigt: NTM verallgemeinert auf längere Sequenzen, LSTM nicht

Zusammenfassung (technisch)

- RNNs sind neuronale Netze die auf sich selbst verweisen
  - Training wie bei normalen Netzen

Zusammenfassung (technisch)

- RNNs sind neuronale Netze die auf sich selbst verweisen
  - Training wie bei normalen Netzen
- LSTM ist Standard-Architektur
  - für simple RNNs
  - Building block für kompliziertere

Zusammenfassung (technisch)

- RNNs sind neuronale Netze die auf sich selbst verweisen
  - Training wie bei normalen Netzen
- LSTM ist Standard-Architektur
  - für simple RNNs
  - Building block für kompliziertere
- Attention ist neues Konzept mit klarer Motivation und tollen Ergebnissen

Referenzen

- Andrej Karpathy: The Unreasonable Effectiveness of Recurrent Neural Networks
  http://karpathy.github.io/2015/05/21/rnn-effectiveness/
- Chris Colah: Understanding LSTM Networks
  http://colah.github.io/posts/2015-08-Understanding-LSTMs/