



## Recurrent Neural Networks

Dr. Grégoire Montavon | Technische Universität Berlin | Machine Learning Group

---



## Outline

### Recurrent Neural Networks

- Overview & Applications

- RNNs from differential equations (Unfolding RNNs)

- Statistics of RNNs

### Exploding and Vanishing Gradient Problems

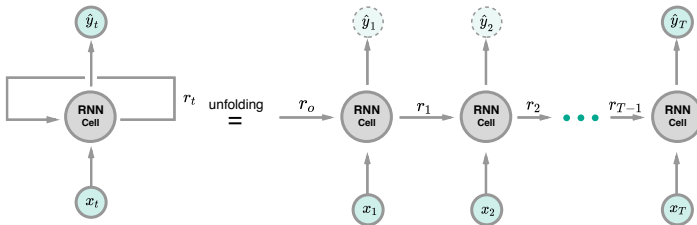
- Echo State Networks (ESNs)

- Long Short-Term Memory & Gated RNNs





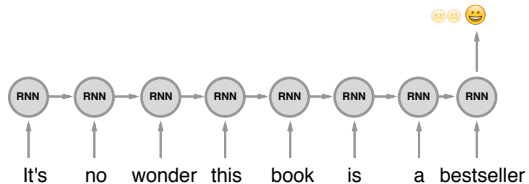
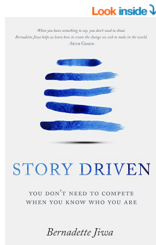
## What are Recurrent Neural Networks (RNNs)?





## What are Recurrent Neural Networks (RNNs)?

### Example Applications



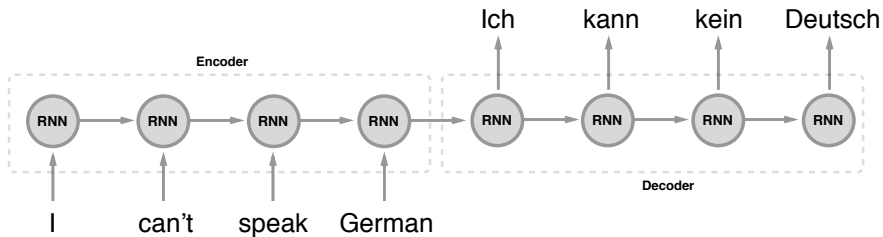
Sentiment Analysis





## What are Recurrent Neural Networks (RNNs)?

### Example Applications



Machine Translation (seq2seq architecture)





## What are Recurrent Neural Networks (RNNs)?

### Example Applications

*Proof. Omitted.* □

**Lemma 0.1.** *Let  $\mathcal{C}$  be a set of the construction.*  
*Let  $\mathcal{C}$  be a gerber covering. Let  $\mathcal{F}$  be a quasi-coherent sheaves of  $\mathcal{O}$ -modules. We have to show that*

$$\mathcal{O}_{\mathcal{O}_X} = \mathcal{O}_X(\mathcal{L})$$

*Proof. This is an algebraic space with the composition of sheaves  $\mathcal{F}$  on  $X_{\text{étale}}$  we have*

$$\mathcal{O}_X(\mathcal{F}) = \{\text{morph}_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

*where  $\mathcal{G}$  defines an isomorphism  $\mathcal{F} \rightarrow \mathcal{F}$  of  $\mathcal{O}$ -modules.* □

**Lemma 0.2.** *This is an integer  $Z$  is injective.*

*Proof. See Spaces, Lemma ??.* □

**Lemma 0.3.** *Let  $S$  be a scheme. Let  $X$  be a scheme and  $X$  is an affine open covering. Let  $\mathcal{U} \subset X$  be a canonical and locally of finite type. Let  $X$  be a scheme. Let  $X$  be a scheme which is equal to the formal complex.*  
*The following to the construction of the lemma follows.*  
*Let  $X$  be a scheme. Let  $X$  be a scheme covering. Let*

$$b : X \rightarrow Y' \rightarrow Y \rightarrow Y' \rightarrow Y' \times_X Y \rightarrow X.$$

*be a morphism of algebraic spaces over  $S$  and  $Y$ .*

*Proof. Let  $X$  be a nonzero scheme of  $X$ . Let  $X$  be an algebraic space. Let  $\mathcal{F}$  be a quasi-coherent sheaf of  $\mathcal{O}_X$ -modules. The following are equivalent*

- (1)  $\mathcal{F}$  is an algebraic space over  $S$ .
- (2) If  $X$  is an affine open covering.

*Consider a common structure on  $X$  and  $X$  the functor  $\mathcal{O}_X(U)$  which is locally of finite type.* □

An RNN learns to create a  $\text{\LaTeX}$  document.

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





## Unfolded computations of an RNN

Assume that  $\theta = \{W^{\text{in}}, W^{\text{rec}}\}$ ,  $r_0 = \mathbf{0}$ , and the **forward pass** is constructed as follows:

$$r_1 = W^{\text{in}} x_1 + W^{\text{rec}} \sigma(r_0)$$

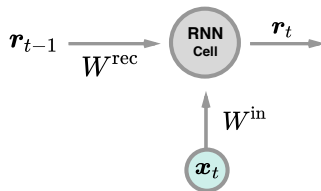
$\vdots$

$$r_t = W^{\text{in}} x_t + W^{\text{rec}} \sigma(r_{t-1})$$

$\vdots$

$$r_T = W^{\text{in}} x_T + W^{\text{rec}} \sigma(r_{T-1})$$

$$\hat{y} = r_T$$

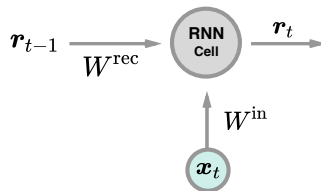




## Unfolded computations of an RNN

**Backpropagation Through Time** can be derived by

$$\begin{aligned}\frac{\partial E}{\partial \theta} &= \sum_{k=1}^T \frac{\partial E}{\partial \mathbf{r}_T} \frac{\partial \mathbf{r}_T}{\partial \mathbf{r}_k} \frac{\partial^+ \mathbf{r}_k}{\partial \theta} \\ \frac{\partial \mathbf{r}_T}{\partial \mathbf{r}_k} &= \prod_{T \geq i > k} \frac{\partial \mathbf{r}_i}{\partial \mathbf{r}_{i-1}} \\ &= \prod_{T \geq i > k} \underbrace{(W^{\text{rec}})^T}_{!} \text{diag}(\sigma'(\mathbf{r}_{i-1}))\end{aligned}$$



$\frac{\partial^+ \mathbf{r}_{t+1}}{\partial \theta}$  is the “immediate” partial derivative. For example, any row in the matrix  $\frac{\partial^+ \mathbf{r}_k}{\partial W^{\text{rec}}}$  is  $\sigma(\mathbf{r}_t)$  [PMB13].







## Exploding and Vanishing Gradient Problems

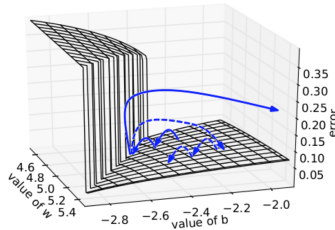
- The exponential term of  $W^{\text{rec}}$  will dominate the calculation of  $\frac{\partial E}{\partial \theta}$ .
- Two problems can happen:
  - Exploding gradient ( $\|\nabla_{\theta} E\| \rightarrow \infty$ )
  - Vanishing gradient ( $\|\nabla_{\theta} E\| \rightarrow 0$ )
- The situation is determined by **spectral radius** of the matrix  $\rho(W^{\text{rec}})$ .
  - *Spectral radius* of a matrix is its largest absolute eigenvalue.





## Exploding Gradient

- happens when  $\rho(W^{\text{rec}}) > 1$  (for linear cases).



Error Function [PMB13].

- causes training *unstable*.





## Exploding Gradient

Exploding gradient can be alleviated by **clipping** the norm of the gradient.

```
 $\epsilon \leftarrow \text{threshold} ;$   
 $\hat{g} \leftarrow \nabla_{\theta} J ;$   
if  $\|\hat{g}\| \geq \text{threshold}$  then  
|  $\hat{g} \leftarrow \frac{\epsilon}{\|\hat{g}\|} \hat{g} ;$   
end
```

**Algorithm 1:** Gradient Clipping [PMB13].





## Varnishing Gradient

- happens when  $\rho(W^{\text{rec}}) < 1$ .
- causes RNNs to learn long-term dependencies with a slow progress.
- **Solutions**
  - Echo State Networks (ESNs)
  - Momentum Technique
  - Long Short-Term Memory Networks





## Echo State Networks [Jae01]

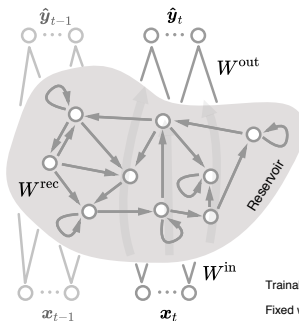
The idea of ESNs is to create a large random reservoir RNN ( $W^{rec}$ ,  $W^{in}$ ). The matrix  $W^{rec}$  is carefully constructed such that reservoir can keep signal **echoed** in the network for long time.

$$\tilde{h}_t = \tanh(W^{in} x_t + W^{rec} h_{t-1}),$$

$$h_t = (1 - \alpha)h_{t-1} + \alpha\tilde{h}_t,$$

$$\hat{y}_t = W^{out} \text{concat}(x_t, h_t),$$

where  $W^{rec} \in \mathbb{R}^{M \times M}$  with  $M$  is the number of reservoir units, and  $\alpha \in (0, 1]$  is the leaking rate.



Trainable weights :  $W^{out}$   
Fixed weights :  $W^{rec}, W^{in}$



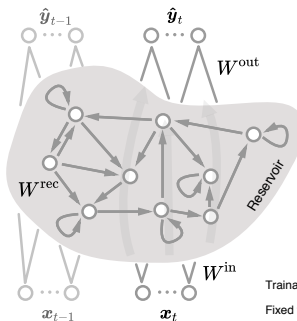


## Echo State Networks [Jae01]

To have the echo state property, the  $W^{\text{rec}}$  is typically initialized as

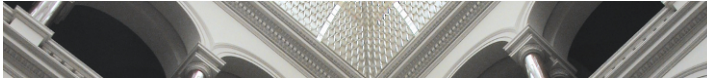
$$W^{\text{rec}} \leftarrow \frac{\xi}{\rho(W^{\text{rec}})} W^{\text{rec}},$$

where  $\xi$  is a hyperparameter.



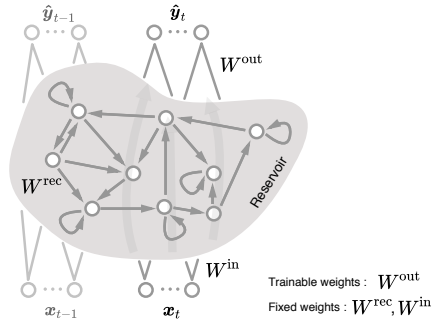
Trainable weights :  $W^{\text{out}}$   
Fixed weights :  $W^{\text{rec}}, W^{\text{in}}$

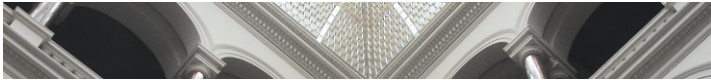




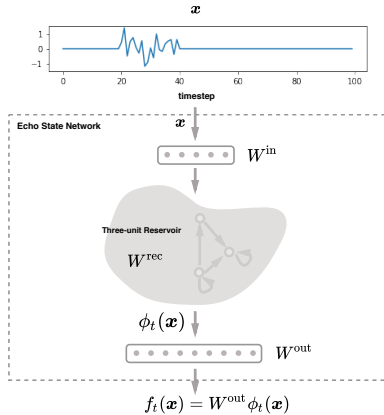
## Echo State Networks [Jae01]

ESNs can be viewed as a kernel trick mapping input to a high-dimensional space and learn a linear classifier ( $W^{\text{out}}$ ) there.





## An example of the echo state property

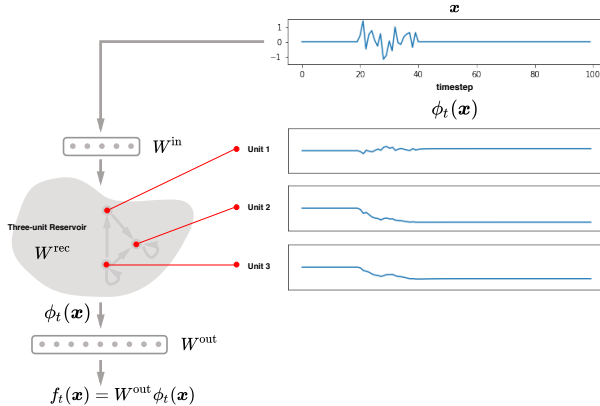






## An example of the echo state property

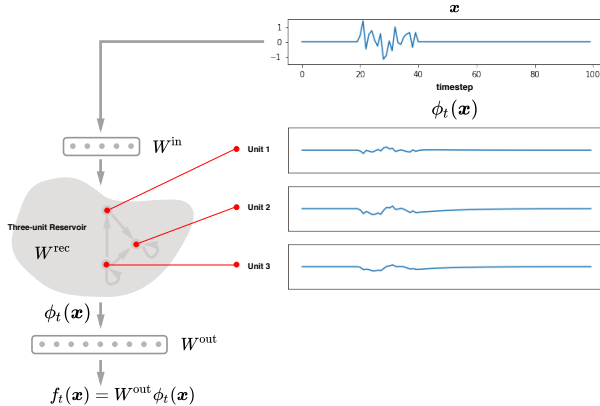
An ESN with  $\alpha = 0.4$  and  $\rho(W^{\text{rec}}) = 1.5$ .





## An example of the echo state property

An ESN with  $\alpha = 0.4$  and  $\rho(W^{\text{rec}}) = 0.8$ .





## Advantages and Disadvantages of ESNs

### Advantages

- Can be trained very fast.
- Work well for low dimensional input.

### Disadvantages

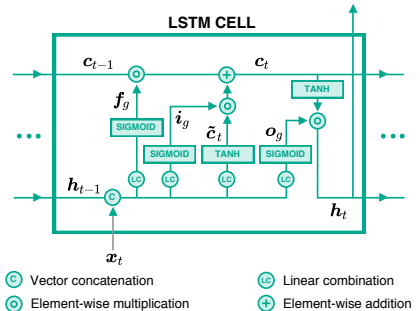
- Require experiences to initialize  $W^{\text{rec}}$  sensibly.
- Require a big reservoir to solve complex problems.





## Long Short-Term Memory (LSTM) [HS97]

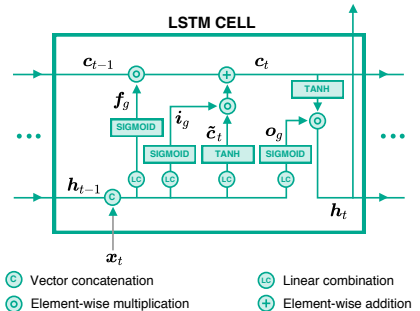
- Use gating mechanisms and additive updates for  $\mathbf{c}_t$ .
  - Eliminate exponential decay factors in the gradient computation





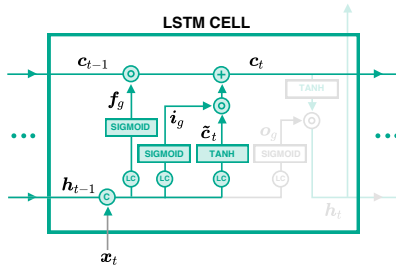
## Long Short-Term Memory (LSTM) [HS97]

- Use gating mechanisms and additive updates for  $c_t$ .
  - Eliminate exponential decay factors in the gradient computation
- Employ 3 gates:
  - Input gate  $i_g$
  - Forget gate  $f_g$
  - Output gate  $o_g$





## Computations in an LSTM cell



$$f_g = \text{sigm}(U^{(f)} \mathbf{x}_t + V^{(f)} \mathbf{h}_{t-1})$$

$$i_g = \text{sigm}(U^{(i)} \mathbf{x}_t + V^{(i)} \mathbf{h}_{t-1})$$

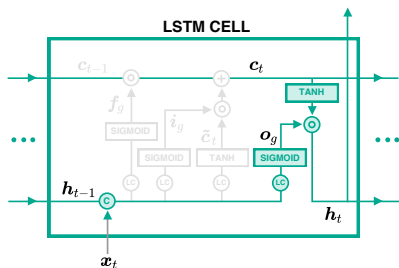
$$\tilde{\mathbf{c}}_t = \tanh(U^{(c)} \mathbf{x}_t + V^{(c)} \mathbf{h}_{t-1})$$

$$\mathbf{c}_t = \mathbf{f}_g \odot \mathbf{c}_{t-1} + \mathbf{i}_g \odot \tilde{\mathbf{c}}_t$$





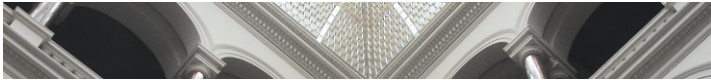
## Computations in an LSTM cell



$$o_g = \text{sigm}(U^{(o)} \mathbf{x}_t + V^{(o)} \mathbf{h}_{t-1})$$

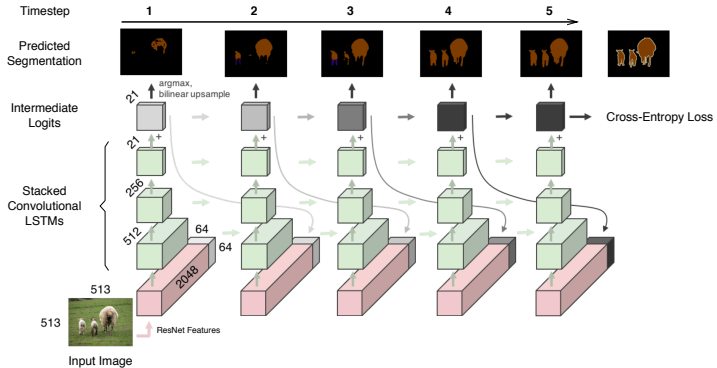
$$\mathbf{h}_t = o_g \circ \tanh(\mathbf{c}_t)$$



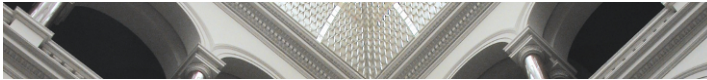


## Deep RNNs

- An RNN for image segmentation [MSMS17].

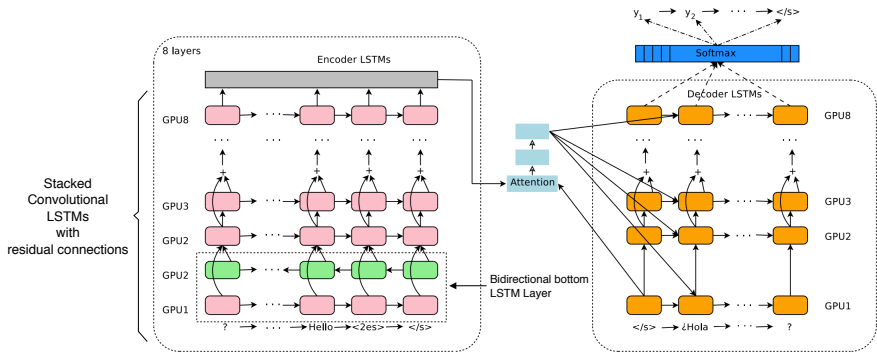


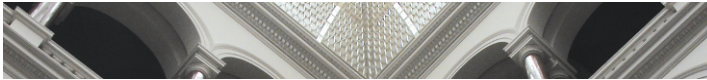




## Deep RNNs

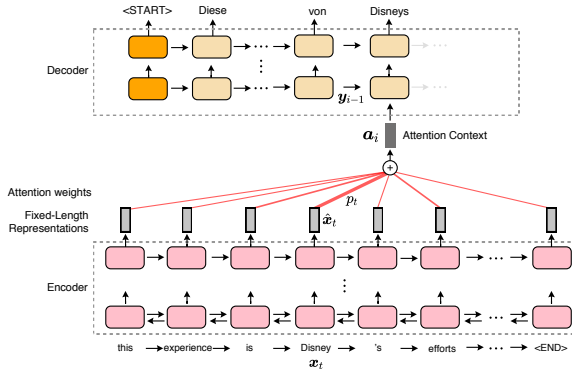
- Google's Neural Machine Translation System [WSC<sup>+</sup>16].





## Deep RNNs

### - Google's Neural Machine Translation System [WSC<sup>+</sup>16].



**Source :** This experience is part of Disney's efforts to enlarge the audience.

**Target :** Diese Erfahrung ist Teil von Disneys Bemühungen, das Publikum zu vergrößern.





## Deep RNNs

- Google's Neural Machine Translation System [WSC<sup>+</sup>16].

$$\forall t, 1 \leq t \leq M$$

$$s_t = \text{AttentionFunction}(\mathbf{y}_{i-1}, \hat{\mathbf{x}}_t)$$

$$p_t = \exp(s_t) / \sum_{t=1}^M \exp(s_t)$$

$$\mathbf{a}_i = \sum_{t=1}^M p_t \hat{\mathbf{x}}_t$$

where

- $M$  is a number of tokens in the sequence.
- *AttentionFunction* can be implemented using a feed forward neural network.





## References I

- [HS97] Sepp Hochreiter and Jürgen Schmidhuber, **Long short-term memory**, Neural Comput. **9** (1997), no. 9, 1735–1780.
- [Jae01] Herbert Jaeger, **The “echo state” approach to analysing and training recurrent neural networks-with an erratum note’**.
- [MSMS17] Lane McIntosh, David Sussillo, Niru Maheswaranathan, and Jonathon Shlens, **Recurrent segmentation for variable computational budgets**, CoRR **abs/1711.10151** (2017).
- [PMB13] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio, **On the difficulty of training recurrent neural networks**, Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013, 2013, pp. 1310–1318.





## References II

[WSC<sup>+</sup>16] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean, **Google's neural machine translation system: Bridging the gap between human and machine translation**, CoRR abs/1609.08144 (2016).

