

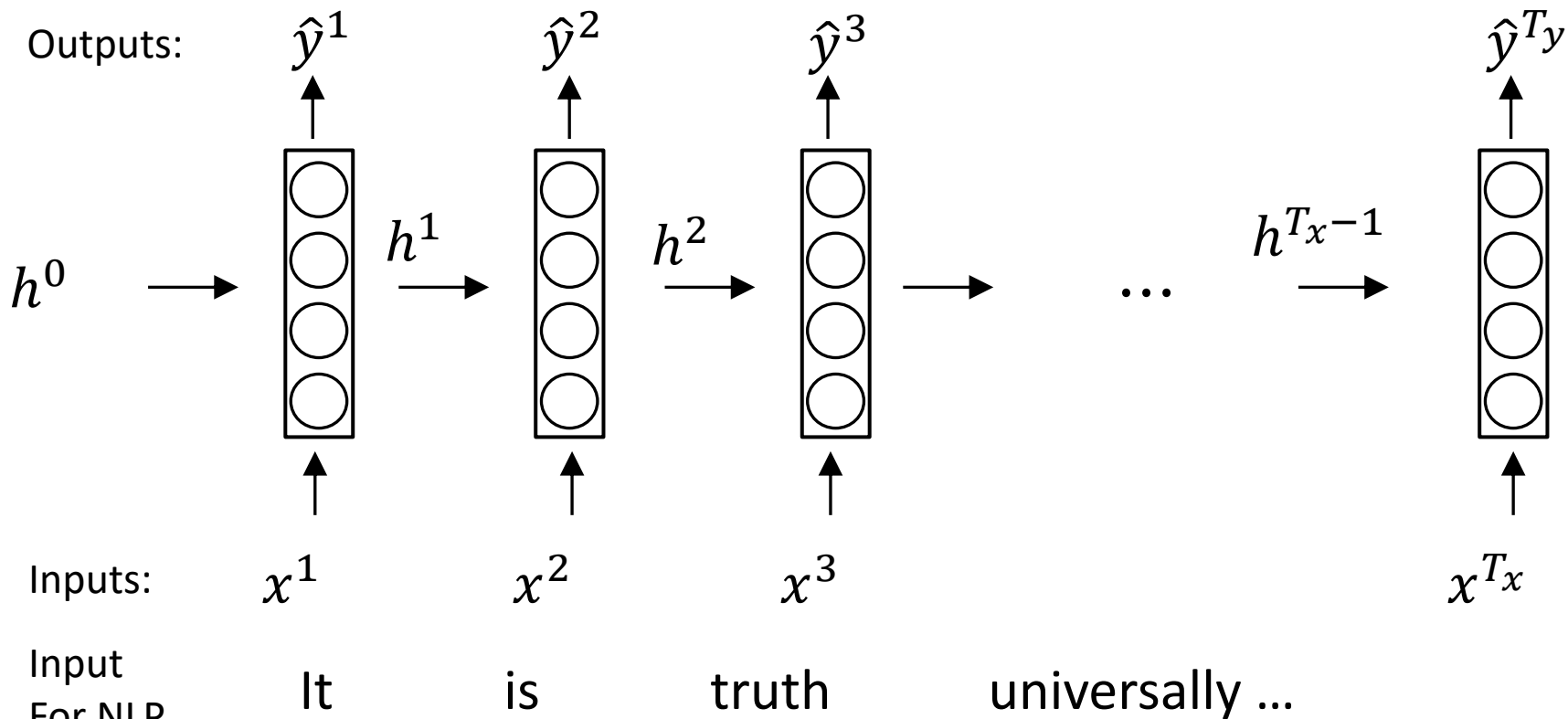
# Deconstructing Recurrent Neural Networks

30 April, 2020

Alexey Zaytsev, head of Laboratory LARSS, PhD

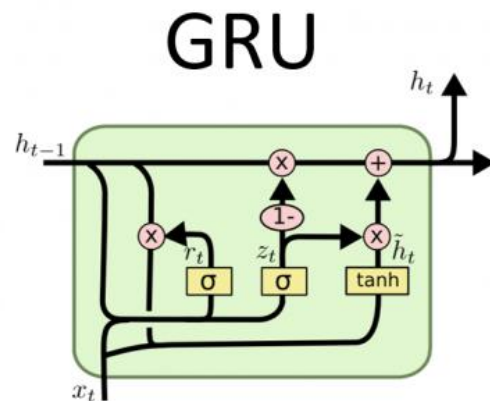
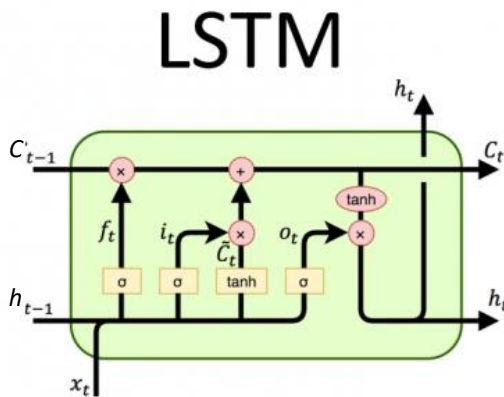
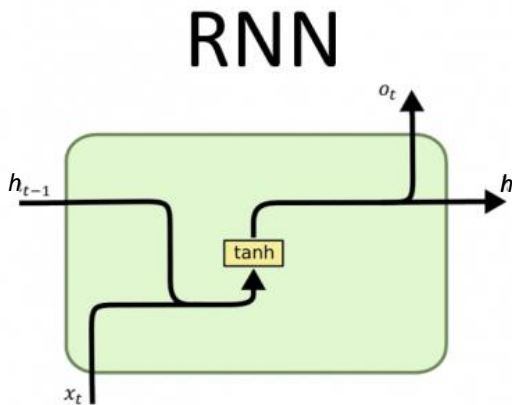
Foundations of Data Science

# Forward propagation through Recurrent Neural Network



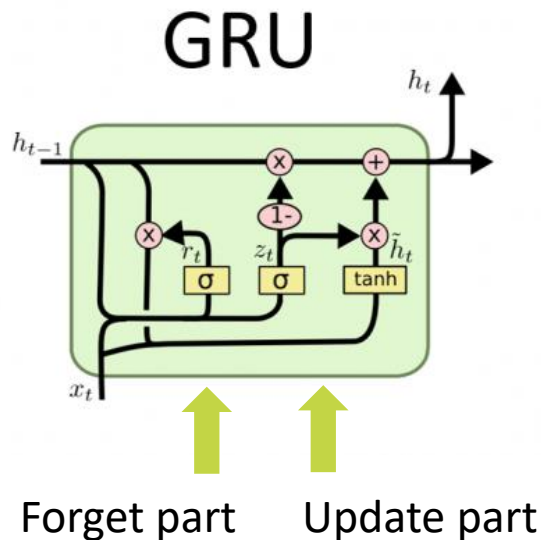
# Good RNN units are LSTM and GRU. Vanilla RNN is not good.

- LSTM: Long Short Term Memory
- GRU: Gated Recurrent Unit



# GRU – Gated Recurrent Unit

- Update gate – what to pay attention to
- Reset gate – what to forget



$$\mathbf{r}^t = \sigma(W_{xr}\mathbf{x}^t + W_{hr}\mathbf{h}^{t-1} + b_r)$$

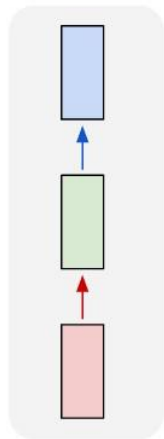
$$\mathbf{z}^t = \sigma(W_{xz}\mathbf{x}^t + W_{hz}\mathbf{h}^{t-1} + b_z)$$

$$\tilde{\mathbf{h}}^t = \tanh(W_{xh}\mathbf{x}^t + W_{hr}(\mathbf{r}^t \odot \mathbf{h}^{t-1}) + b_h)$$

$$\mathbf{h}^t = \mathbf{z}^t \odot \mathbf{h}^{t-1} + (1 - \mathbf{z}^t) \odot \tilde{\mathbf{h}}^t$$

# The Unreasonable Effectiveness of Recurrent Neural Networks

one to one



Example:

one to many

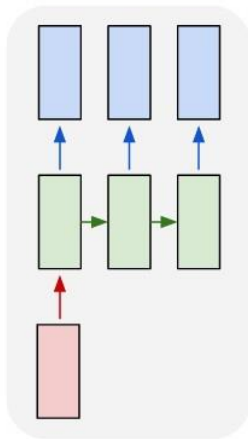
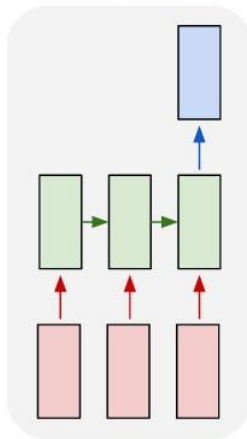


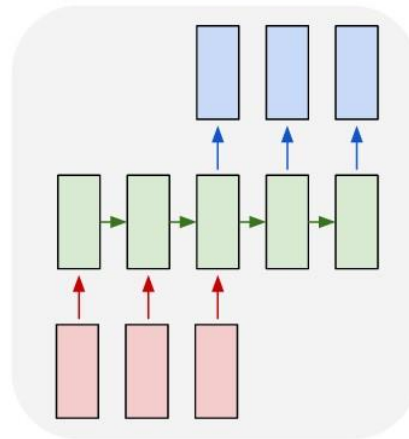
Image  
captioning

many to one



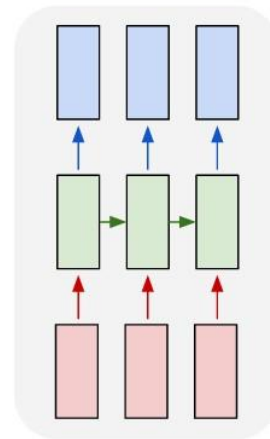
Sentiment  
analysis

many to many



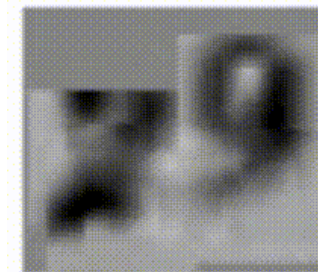
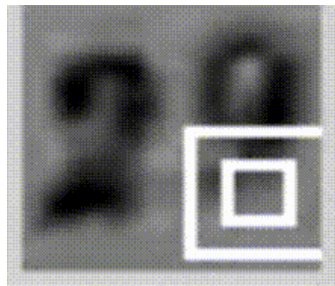
Machine  
Translation

many to many

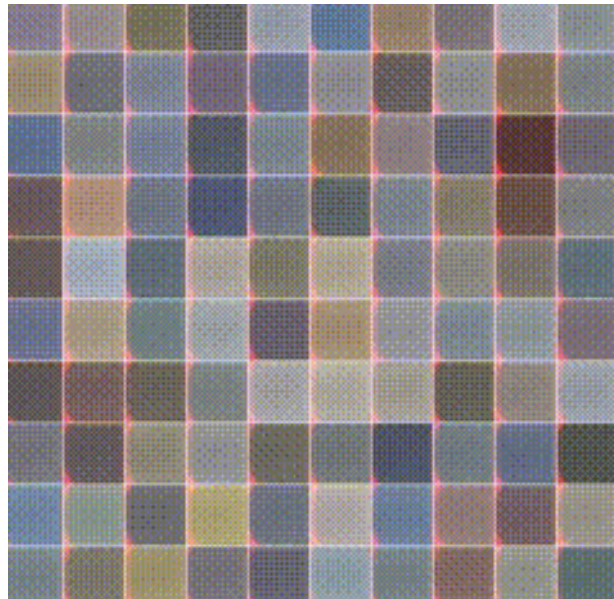


Video  
classification

# Sequential processing in absence of sequences



RNN learns to  
read house numbers



RNN learns to  
paint house numbers

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

Ba, J., Mnih, V., Kavukcuoglu, K. Multiple object recognition with visual attention. arXiv preprint arXiv:1412.7755.

Gregor, K. et al. 2015. Draw: A recurrent neural network for image generation. arXiv preprint arXiv:1502.04623.

# Text generation (now there are better alternatives)

- Paul Graham generator
- Shakespeare
- Wikipedia
- Algebraic Geometry (Latex)
- Linux Source Code
- Generating Baby Names

For  $\bigoplus_{i=1}^n \mathcal{L}_{\alpha_i} = 0$ , hence we can find a closed subset  $H$  in  $H$  and any sets  $F$  on  $X$ ,  $U$  is a closed immersion of  $S$ , then  $U \rightarrow T$  is a separated algebraic space.

*Proof.* Proof of (1). It also start we get

$S = \text{Spec}(R) = U \times_X U \times_X U$   
and the comparability in the fibre product covering we have to prove the lemma generated by  $\coprod U \times_U U \rightarrow V$ . Consider the maps  $M$  along the set of points  $\text{Sch}_{\text{aff}}$  and  $U \rightarrow U$  is the fibre category of  $S$  in  $U$  in Section 77 and the fact that any  $U$  affine, see Morphisms, Lemma 77. Hence we obtain a scheme  $S$  and any open subset  $W \subset U$  in  $\text{Sch}(G)$  such that  $\text{Spec}(R) \rightarrow S$  is smooth or an

$$U = \bigcup U_i \times_S U_i$$

which has a nonzero morphism we may assume that  $f_i$  is of finite presentation over  $S$ . We claim that  $\mathcal{O}_{X,S}$  is a scheme where  $x, x', x'' \in S'$  such that  $\mathcal{O}_{X,S'} \rightarrow \mathcal{O}_{X,S''}$  is separated. By Algebra, Lemma 77 we can define a map of complexes  $\text{GL}_2(x'/S'')$  and we win.

To prove study we see that  $\mathcal{F}_i^{\vee}$  is a covering of  $X'$ , and  $\mathcal{T}_i$  is an object of  $\mathcal{F}_{X/S}$  for  $i > 0$  and  $\mathcal{F}_p$  exists and let  $\mathcal{F}_i$  be a presheaf of  $\mathcal{O}_{X,S}$ -modules on  $\mathcal{C}$  as a  $\mathcal{F}$ -module. In particular  $\mathcal{F} = U/F$  we have to show that

$$\tilde{\Delta}^* = \mathcal{F}^* \otimes_{\text{Spec}(k)} \mathcal{O}_{S,k} - \mathcal{I}_k^* \mathcal{F}$$

is a unique morphism of algebraic stacks. Note that  
 $\text{Arrows} = (\text{Sch}/S)_{\text{fppf}} / (\text{Sch}/S)_{\text{fppf}}$

and

$$V = \Gamma(S, \mathcal{O}) \rightarrow \Gamma(U, \text{Spec}(A))$$

is an open subset of  $X$ . Thus  $U$  is affine. This is a continuous map of  $X$  is the inverse, the groupoid scheme  $S$ .

*Proof.* See discussion of sheaves of sets.  $\square$

The result for prove any open covering follows from the loss of Example 77. It may replace  $S$  by  $X_{\text{fppf}, \text{fppf}}$  which gives an open subspace of  $X$  and  $T$  equal to  $S_{2, \text{fppf}}$  see Descent, Lemma 77. Namely, by Lemma 77 we see that  $R$  is geometrically regular over  $S$ .

**Lemma 0.1.** Assume (1) and (3) by the construction in the description.

Suppose  $X = \lim X_i$  (by the formal open covering  $X$  and a single map  $\text{Proj}_X(A) = \text{Spec}(B)$  over  $U$  compatible with the complex

$$\text{Set}(A) = \Gamma(X, \mathcal{O}_X \otimes_{X_i} \mathcal{O}_{X_i})$$

When in this case of to show that  $Q \rightarrow \mathcal{C}_{2, X}$  is stable under the following result in the several conditions of (1), and (3). This finishes the proof. By Definition 77 (without element is when the closed subschemes are category. If  $T$  is surjective we may assume that  $T$  is connected with residue fields of  $S$ . Moreover there exists a closed subspace  $Z \subset X$  of  $X$  where  $U$  in  $X'$  is proper (some defining as a closed subset of the uniqueness it suffices to check the fact that the following theorem

(1)  $f$  is locally of finite type. Since  $S = \text{Spec}(R)$  and  $Y = \text{Spec}(R)$ .

*Proof.* This is form all sheaves of sheaves on  $X$ . But given a scheme  $U$  and a surjective étale morphism  $U \rightarrow X$ . Let  $U \cap U_i = \bigcup_{j=1, \dots, n} U_j$  be the scheme  $X$  over  $S$  at the schemes  $X_i \rightarrow X$  and  $U = \lim X_i$ .  $\square$

The following lemma surjective restroscomposes of this implies that  $\mathcal{F}_n = \mathcal{F}_n = \mathcal{F}_{n-1} = \mathcal{F}_{n-2}$ .

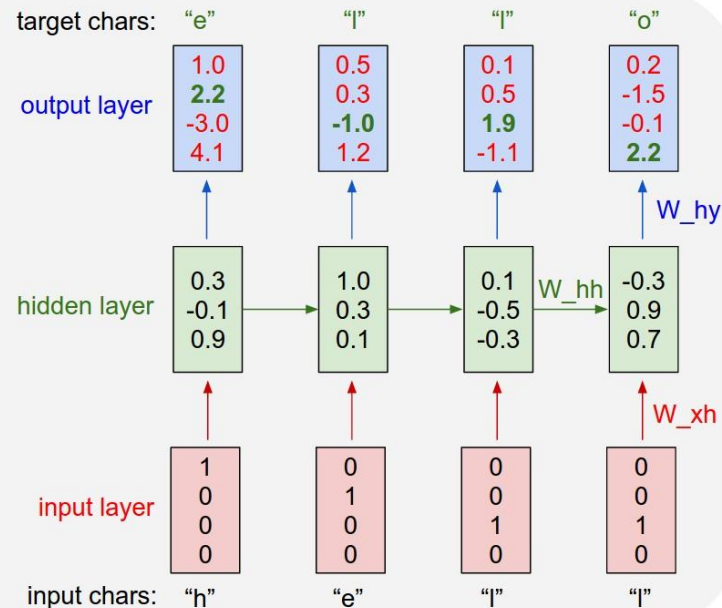
**Lemma 0.2.** Let  $X$  be a locally Noetherian scheme over  $S$ ,  $E = \mathcal{F}_{X/S}$ . Set  $\mathcal{I} = \mathcal{I}_1 \subset \mathcal{I}_2 \subset \dots$ . Since  $\mathcal{T}^n \subset \mathcal{T}^n$  are nonzero over  $u_0 \leq p$  is a subset of  $\mathcal{F}_{n,0} \circ \mathcal{A}_2$  works.

**Lemma 0.3.** In Situation 77. Hence we may assume  $q^* = 0$ .

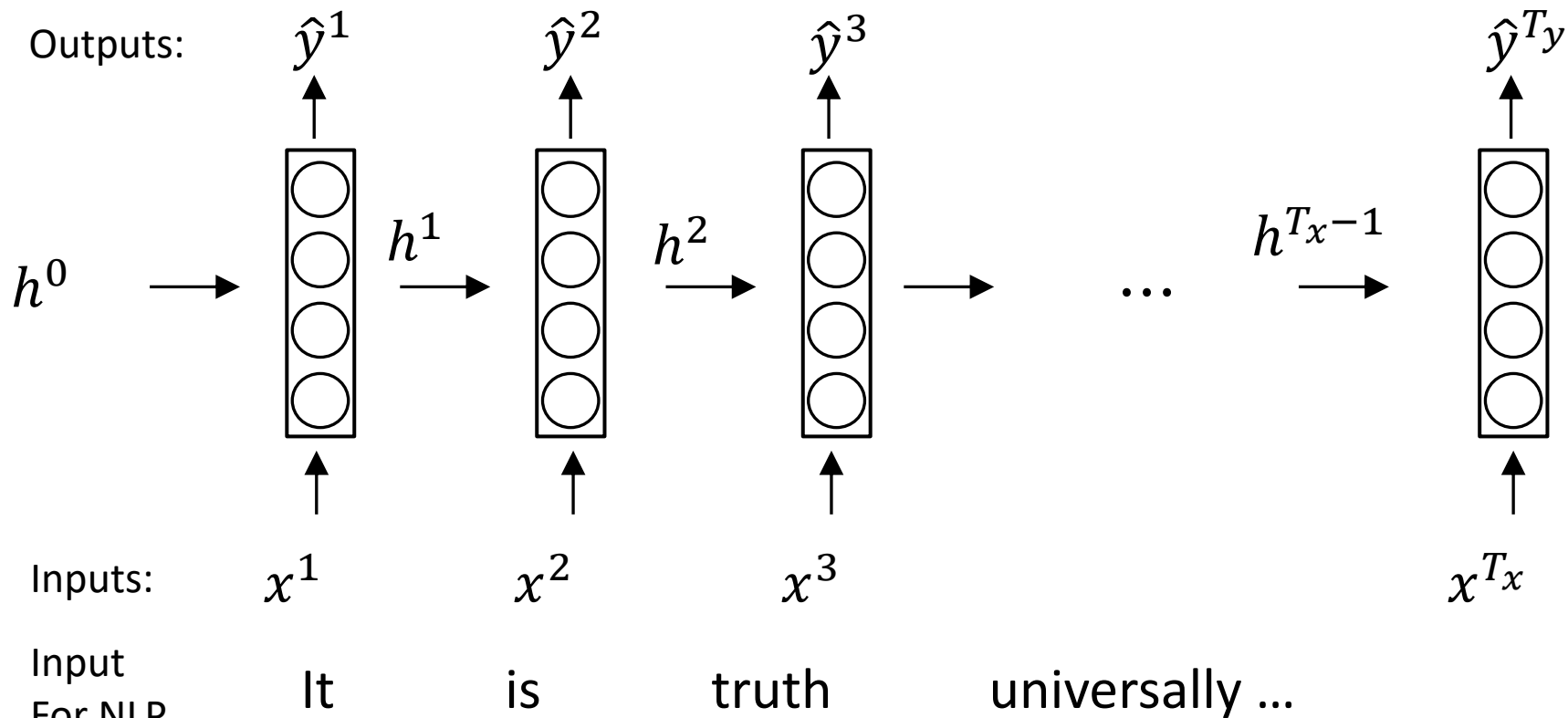
*Proof.* We will use the property we see that  $p$  is the next functor (77). On the other hand, by Lemma 77 we see that

$$D(\mathcal{O}_{X'}) = \mathcal{O}_{X'}(D)$$

where  $K$  is an  $F$ -algebra where  $\delta_{n+1}$  is a scheme over  $S$ .  $\square$

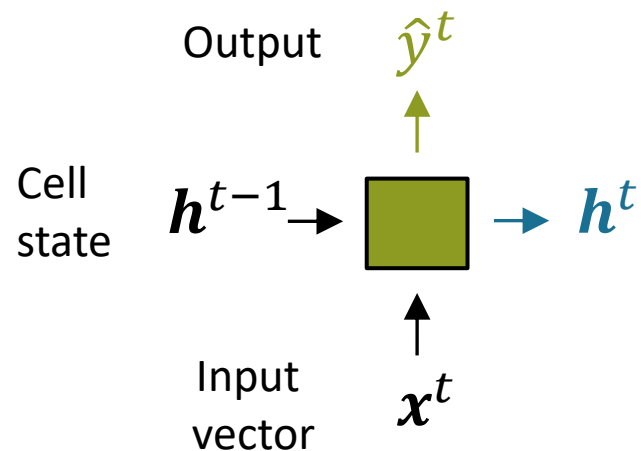


## Forward propagation through RNN





# Simple RNN model



$$\mathbf{h}^t = f_h(\mathbf{x}^t, \mathbf{h}^{t-1})$$

$$\mathbf{h}^t = \tanh(V\mathbf{x}^t + W\mathbf{h}^{t-1} + b_h)$$

$$\hat{\mathbf{y}}^t = f_y(\mathbf{h}^t)$$

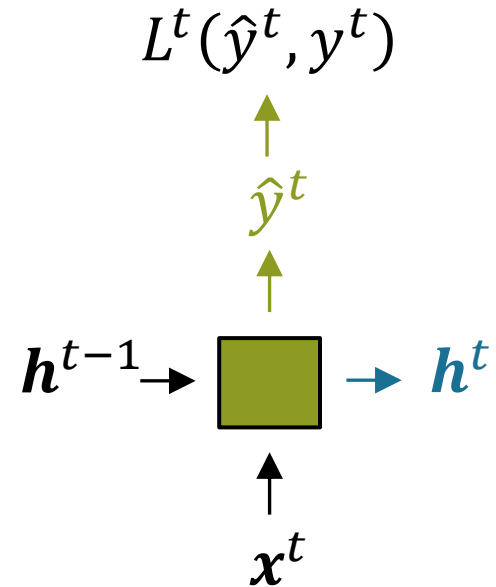
$$\hat{\mathbf{y}}^t = \text{softmax}(U\mathbf{h}^t + b_y)$$

## Backpropagation w.r.t. $U$

$$L = \sum_{i=1}^{T_y} L^i(\hat{y}^i, y^i)$$

$$\frac{\partial L}{\partial U} = \sum_{i=1}^{T_y} \frac{\partial L_i}{\partial U} = \sum_{i=1}^{T_y} \frac{\partial L_i}{\partial \hat{y}^i} \frac{\partial \hat{y}^i}{\partial U}$$

$$\hat{y}^t = \text{softmax}(U\mathbf{h}^t + b_y)$$



## Backpropagation w.r.t. $W$

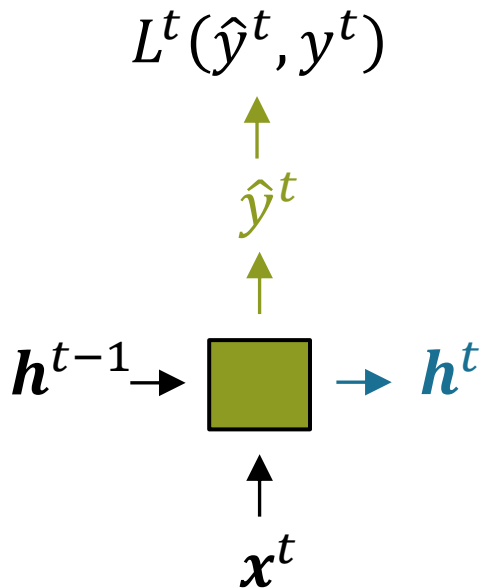
$$L = \sum_{i=1}^{T_y} L^i(\hat{y}^i, y^i)$$

$$\mathbf{h}^t = \tanh(V\mathbf{x}^t + W\mathbf{h}^{t-1} + b_h)$$

$$\frac{\partial L_i}{\partial \hat{y}^i} \frac{\partial \hat{y}^i}{\partial W} = \frac{\partial L_i}{\partial \hat{y}^i} \frac{\partial \hat{y}^i}{\partial h_t} \left( \frac{\partial h_t}{\partial W} + \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial W} + \dots \right)$$

$$\frac{\partial L_i}{\partial \hat{y}^i} \frac{\partial \hat{y}^i}{\partial W} = \frac{\partial L_i}{\partial \hat{y}^i} \frac{\partial \hat{y}^i}{\partial h_t} \sum_{i=0}^{T_y} \left( \prod_{j=i+1}^{T_y} \frac{\partial h_j}{\partial h_{j-1}} \right) \frac{\partial h_i}{\partial W}$$

$$\hat{y}^t = \text{softmax}(U\mathbf{h}^t + b_y)$$



## Jacobian matrix is the cause of all problems

$$\mathbf{h}^t = \tanh(V\mathbf{x}^t + W\mathbf{h}^{t-1} + b_h)$$

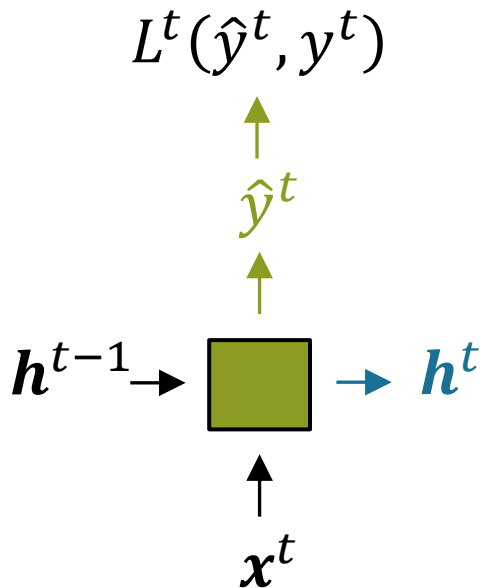
$$\frac{\partial L_i}{\partial \hat{y}^i} \frac{\partial \hat{y}^i}{\partial W} = \frac{\partial L_i}{\partial \hat{y}^i} \frac{\partial \hat{y}^i}{\partial h_t} \left( \frac{\partial h_t}{\partial W} + \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial W} + \dots \right)$$

$$\frac{\partial L_i}{\partial \hat{y}^i} \frac{\partial \hat{y}^i}{\partial W} = \frac{\partial L_i}{\partial \hat{y}^i} \frac{\partial \hat{y}^i}{\partial h_t} \sum_{i=0}^{T_y} \left( \prod_{j=i+1}^{T_y} \frac{\partial h_j}{\partial h_{j-1}} \right) \frac{\partial h_i}{\partial W}$$

$$\left\| \frac{\partial h_j}{\partial h_{j-1}} \right\|_2 > 1 \quad \text{gradient exploration}$$

$$\left\| \frac{\partial h_j}{\partial h_{j-1}} \right\|_2 < 1 \quad \text{gradient vanishing}$$

$$\hat{\mathbf{y}}^t = \text{softmax}(U\mathbf{h}^t + b_y)$$



## Gradient exploding and vanishing

$$\mathbf{h}^j = f(V\mathbf{x}^j + W\mathbf{h}^{j-1} + b_h) = f(\mathbf{z}_t)$$

$$\frac{\partial \mathbf{h}_j}{\partial \mathbf{h}_{j-1}} = \frac{\partial \mathbf{h}_j}{\partial \mathbf{z}_j} \frac{\partial \mathbf{z}_j}{\partial \mathbf{h}_{j-1}} = \text{diag}(f'(\mathbf{z}_t))W$$

$$\left\| \frac{\partial \mathbf{h}_j}{\partial \mathbf{h}_{j-1}} \right\| \leq \|\text{diag}(f'(\mathbf{z}_t))\| \cdot \|W\|$$

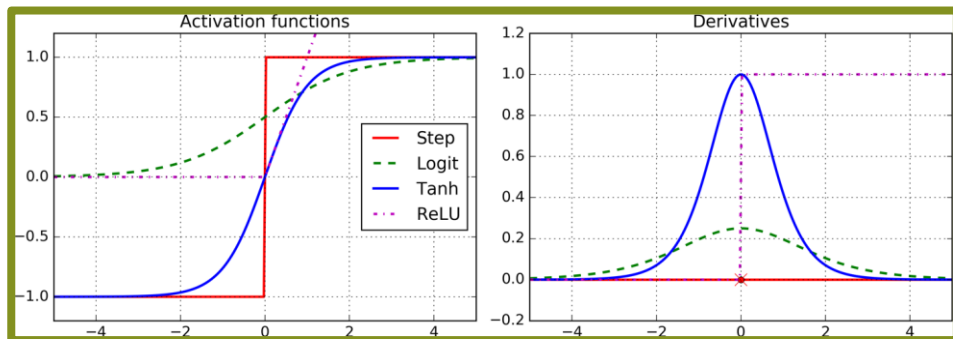
# Gradient exploding and vanishing

$$\mathbf{h}^j = f(V\mathbf{x}^j + W\mathbf{h}^{j-1} + b_h) = f(\mathbf{z}_t)$$

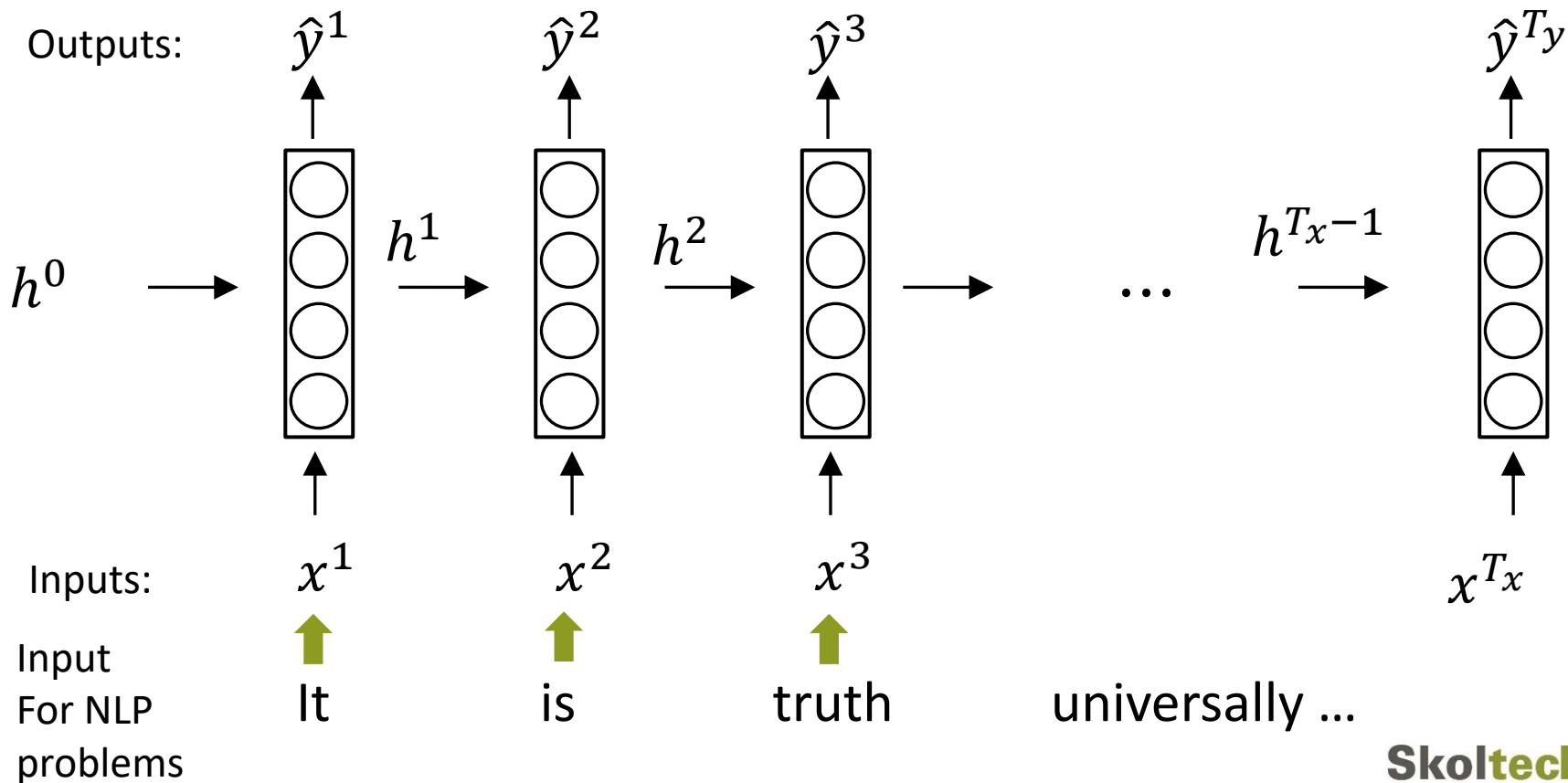
$$\frac{\partial \mathbf{h}_j}{\partial \mathbf{h}_{j-1}} = \frac{\partial \mathbf{h}_j}{\partial \mathbf{z}_j} \frac{\partial \mathbf{z}_j}{\partial \mathbf{h}_{j-1}} = \text{diag}(f'(\mathbf{z}_t))W$$

activation function  
– ReLU is the best option

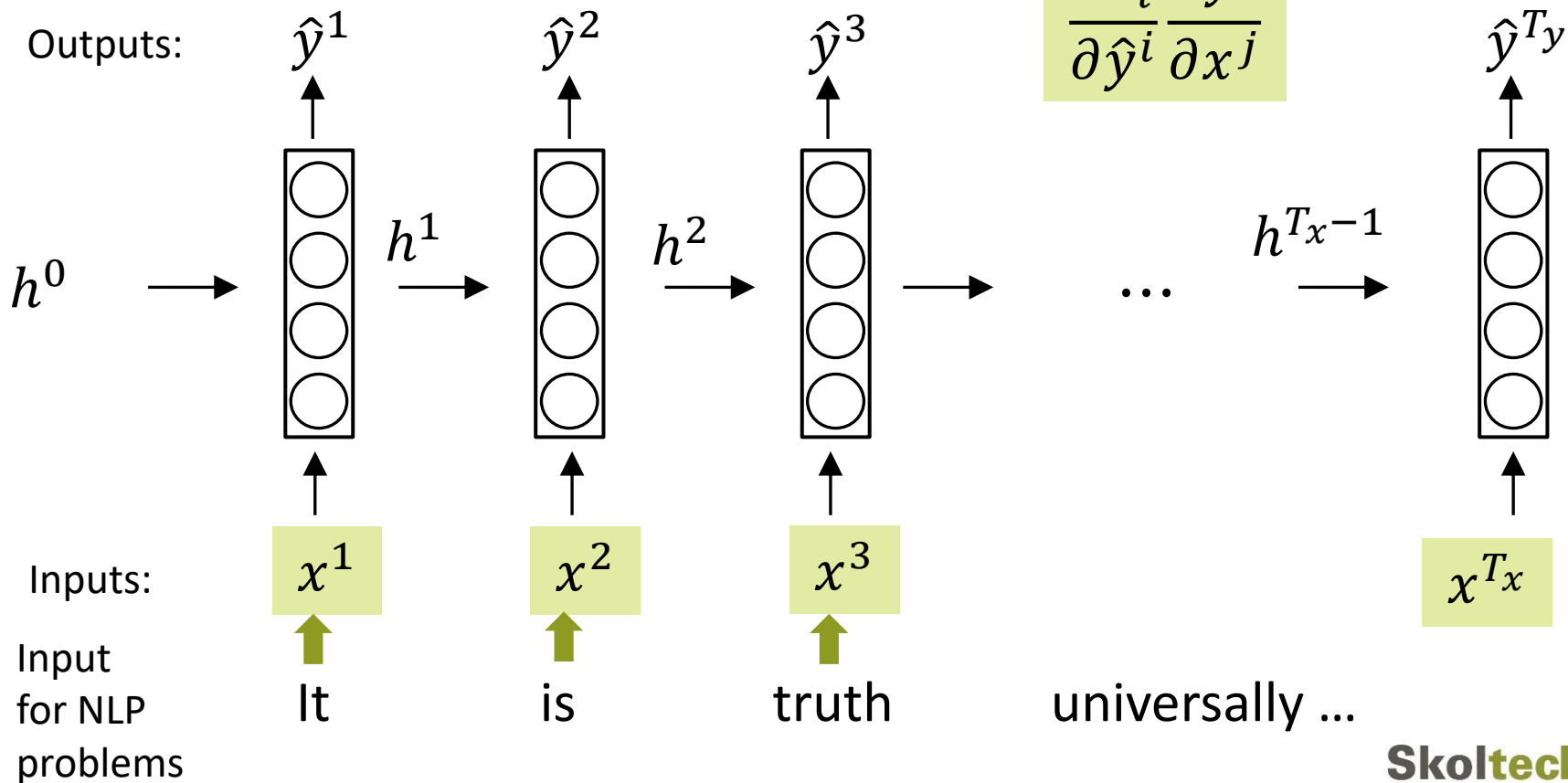
- The initialization matters  
For orthogonal matrix  $Q^T = Q^{-1}$   
all eigenvalues equals one
- Regularization



## Embedding layer for RNNs



## Backpropagation for embedding layer for RNNs

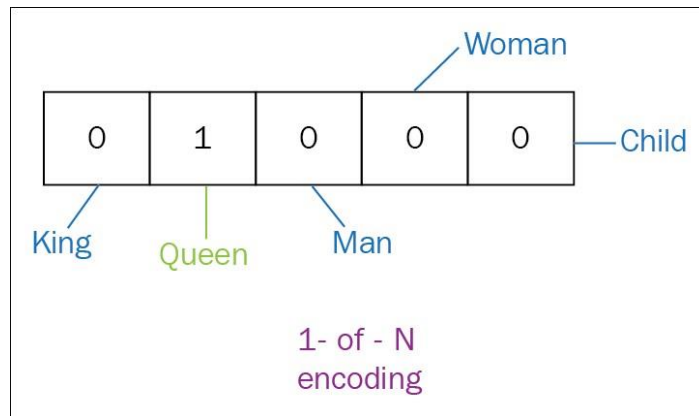




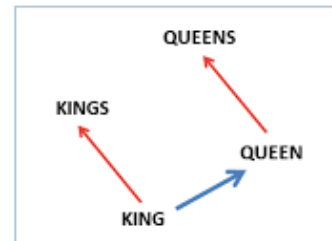
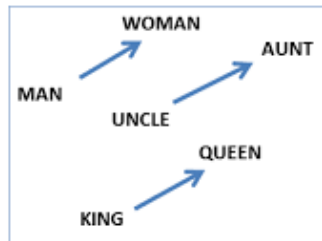
# Learning of embeddings

End2end learning of embeddings for:

- Words
- Codes of transactions (gas station)
- Prescription drugs
- Symbols or groups of symbols



One-hot encoding is also an embeddings

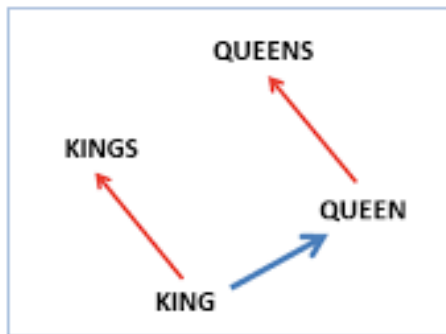
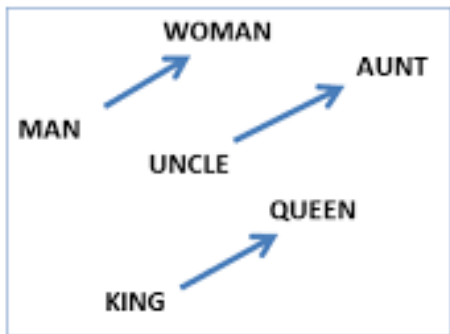


But we want something smarter

# Why we need embeddings?

In NLP it is hard to measure quality of a solution, for example in Machine translation

- Check that we learn something adequate by comparing feature vectors for objects with similar relationships

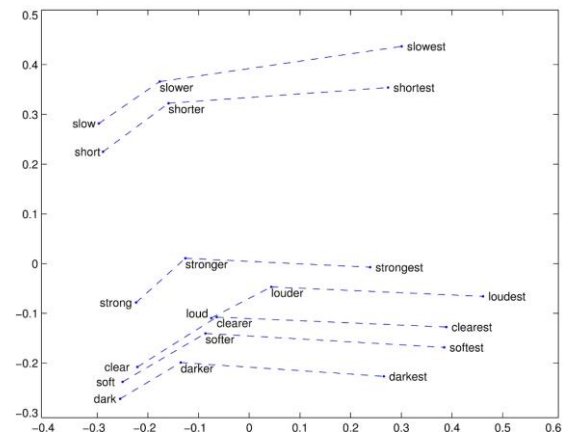


Paris – France

Moscow – Russia

Vienna – Austria

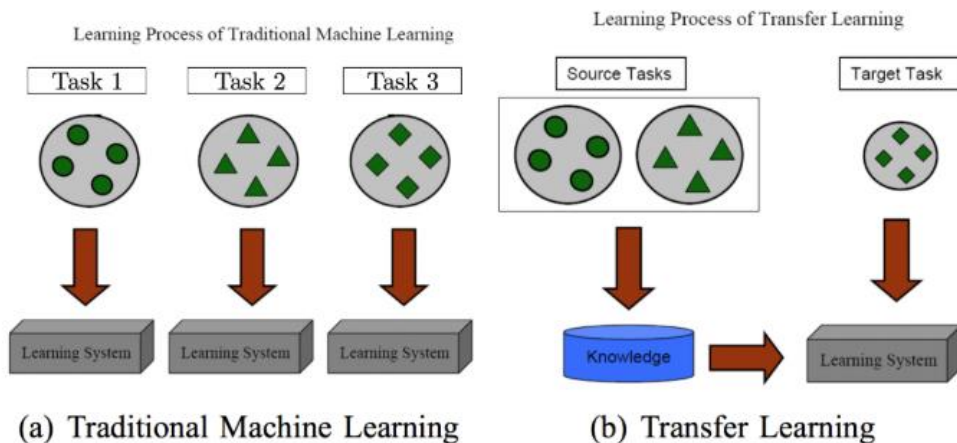
Antananarivo – Madagascar



# Why we need embeddings?

We finally have representation learning –  
but now for sequences

- Use embeddings for transfer learning



# Why we need embeddings?

Natural way to look at tokens

- Use embeddings to find their similarity

	animal	fluffiness	dangerous	spooky
aardvark	0.97	0.03	0.15	0.04
black	0.07	0.01	0.20	0.95
cat	0.98	0.98	0.45	0.35
duvet	0.01	0.84	0.12	0.02
zombie	0.74	0.05	0.98	0.93

# Visualizing embeddings

## Character-level LSTM, look at some dimensions

t t p : / / w w w . y n e t n e w s . c o m / ] E n g l i s h - l a n g u a g e w e b s i t e o f I s r a e l ' s l a r  
t p : / / w w w . b a c a h e t s . c o m / - x g l i s h l i n g u a g e s a i r s i t e o f t s l a e l i s s i n g  
d : x n e . w a e a . . a w a t o a . s & n t i a c a - s a r d e e l h o a n t b i s a n f a n r e i f ' a a t d  
m w - 2 p i i i s o e s s i s . / e r n . c ] ( d c e e n e p e s a a i k i i e e l e d h , i r t h r a o n s e , c o s e  
d r . < : a h b - n p t w t . x i g h / m a ) T v d r y z i c o u e d l s u : t h a - o o t u , s t u i f l v e p e r y  
s t p , t c o a 2 d r u l w o c l e n s r ] p . l i l v a o d , , e y t c - n d m - o i b u v s ] b b i m s u l t a t l y b n

g e s t n e w s p a p e r ' ' [ [ Y e d i o t h A h r e n o t h ] ] ' ' ' ' H e b r e w - l a n g u a g e p e r i o d  
e t a a w s p a p e r s o [ [ T e l t t i ( f e a n e m t i ) ' ' ' ' [ e r r e w s l e n g u a g e : a r o s o d i  
i r s c o e e n a i T T h A o a i n n h S r m u w ] e y s [ ' i n e i a ' s i w d d e ' h s o l r i f r :  
u s . . s e t l g o r s . a s a t C a r e e g ' a C l r i s z ] i e ' : : , # : T A a a a a t B a s e e i l o ' i a n f v l  
- t u a e v r t i d , t B A m S u s y u t ] ] A s a o i g s ] , . : s M B o l o u s : T o u a - n : d w o a p n u  
a , d , i i u i t i c p . ] ( l S v H v t u s u i e D n o e g a n o . , ] : { C C u i b o h e C y b k s l s : r - e p c n t s

i c a l s : ' ' ' ' \* ' ' [ [ G l o b e s ] ] ' ' [ h t t p : / / w w w . g l o b e s . c o . i l / ] b u s i n e s s d a  
c a l : ' ' ' ' \* ' ' [ T a a b a ] ' ' ( [ t t p : / / w w w . b u o b a l . c o m u n / s A - y t i n e s s a e t  
s t l ' ' ' ' [ h A e o v e l t s a h a d : x g e . w a o i r . r t o a . e l . i T & a i e g e o o y  
t t ' ' ' ' ' & [ & m C o e r o n e ' : : , i ' o d w . , n i i i s a a u e . e n i / o m l c C . ( e f t g i r i i u  
a ' n : , C : & : # \* : a f D r u s u ] l , . o m e l p < , d h a : d e u o o t / i h n c s i f S , u r h o s t , t u n  
n k i < ] : & 1 1 s T G u i t r s i , : b a c m r - x t p o b - g r e s i s l e r l n a f a D ] l o s p t a d , i f r m

i l y \* ' ' [ [ H a a r e t z ] H a ' A r e t z ] ' ' [ h t t p : / / w w w . h a a r e t z . c o . i l / ] R e l a t i v  
l y \* ' ' [ [ T e r r d n F e r a n t a h ] ] ' ' ( [ t t p : / / w w w . b o n m d s t . c o m u n / s - e s a t e o i  
r e ' ' h A i l n n t t e H a l s r c n o l ' s a h a d : x n e . w a a m r t d h e o h . o l . c & o p i n i v e  
k i . : \* s C O S a n l t h i T i m ' l i ] e : , i m c d w - 2 p h i i s e r d i t . i n a / c m f i . ( a f l c a n a  
d s - ! [ t B T C o m m g d ] ] W o n a a e , : . b a e r r . < t a i b - d u l c n n c / a r n e s i ] l i c e y s t o  
n d s # & : G l D u v c c s a o S u c l t e l ] z ] , : o ' o m t ] , : e o a 2 n i v f s r o e i u n a l a ) u v v r o

The neuron highlighted in this image seems to get very excited about URLs and turns off outside of the URLs. The LSTM is likely using this neuron to remember if it is inside a URL or not.

# Visualizing embeddings

## Character-level LSTM, look at some dimensions

Cell sensitive to position in line:

```
The sole importance of the crossing of the Berezina lies in the fact
that it plainly and indubitably proved the fallacy of all the plans for
cutting off the enemy's retreat and the soundness of the only possible
line of action--the one Kutuzov and the general mass of the army
demanded--namely, simply to follow the enemy up. The French crowd fled
at a continually increasing speed and all its energy was directed to
reaching its goal. It fled like a wounded animal and it was impossible
to block its path. This was shown not so much by the arrangements it
made for crossing as by what took place at the bridges. When the bridges
broke down, unarmed soldiers, people from Moscow and women with children
who were with the French transport, all--carried on by vis inertiae--
pressed forward into boats and into the ice-covered water and did not,
surrender.
```

Cell that turns on inside quotes:

```
"You mean to imply that I have nothing to eat out of... On the
contrary, I can supply you with everything even if you want to give
dinner parties," warmly replied Chichagov, who tried by every word he
spoke to prove his own rectitude and therefore imagined Kutuzov to be
animated by the same desire.
```

```
Kutuzov, shrugging his shoulders, replied with his subtle penetrating
smile: "I meant merely to say what I said."
```

Cell that robustly activates inside if statements:

```
static int __dequeue_signal(struct sigpending *pending, sigset_t *mask,
    siginfo_t *info)
{
    int sig = next_signal(pending, mask);
    if (sig) {
        if (current->notifier) {
            if (sigismember(current->notifier_mask, sig)) {
                if (!current->notifier(current->notifier_data)) {
                    clear_thread_flag(TIF_SIGPENDING);
                    return 0;
                }
            }
        }
        collect_signal(sig, pending, info);
    }
    return sig;
}
```

A large portion of cells are not easily interpretable. Here is a typical example:

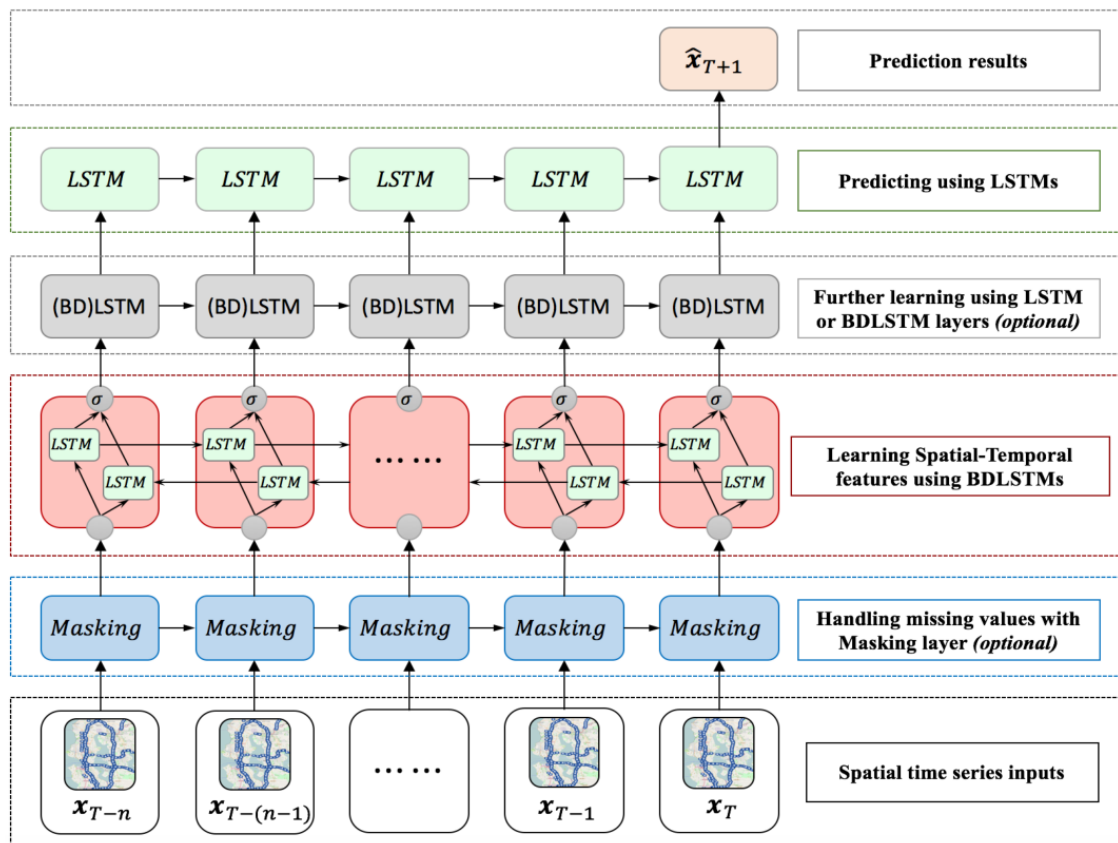
```
/* Unpack a filter field's string representation from user-space
 * buffer. */
char *audit_unpack_string(void **bufp, size_t *remain, size_t len)
{
    char *str;
    if (!*bufp || (len == 0) || (len > *remain))
        return ERR_PTR(-EINVAL);
    /* of the currently implemented string fields, PATH_MAX
     * defines the longest valid length.
     */
}
```

Look at this video  
for more examples  
for LSTMViz



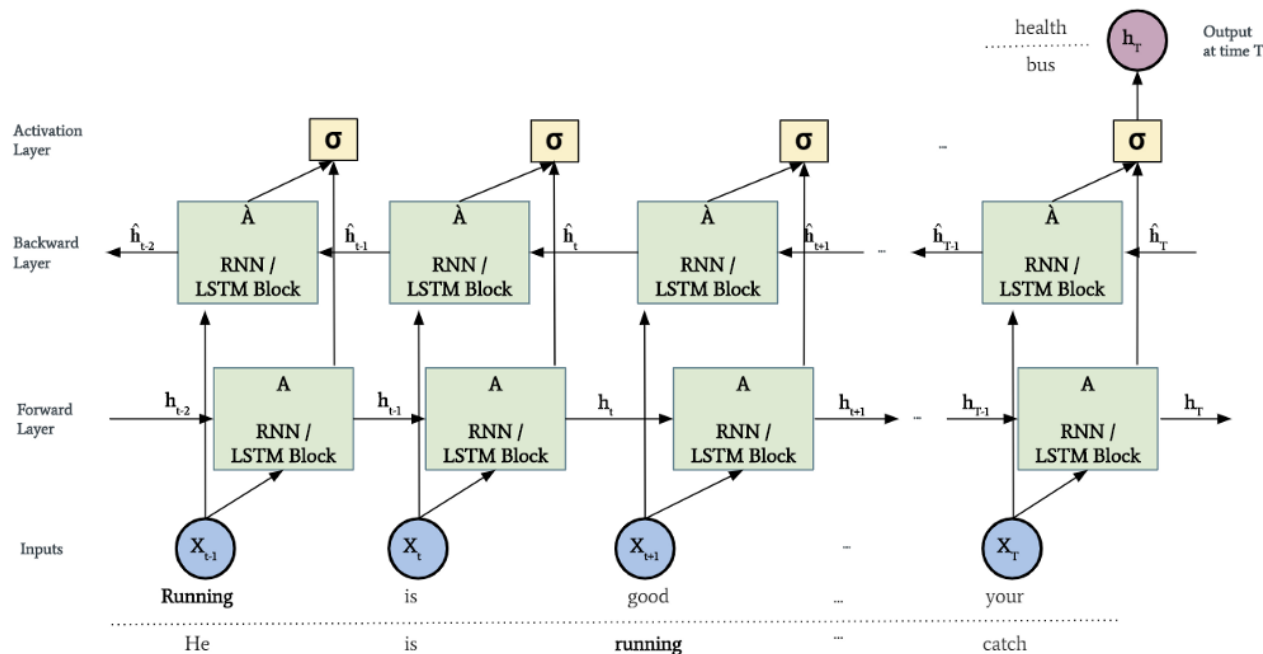
<http://lstm.seas.harvard.edu/>

# Various architectures: Going deeper





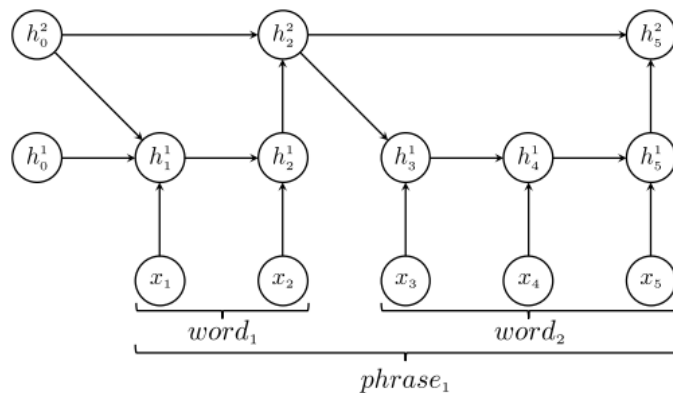
# Various architectures: Bidirectional LSTM





# Various architectures: Hierarchical RNN

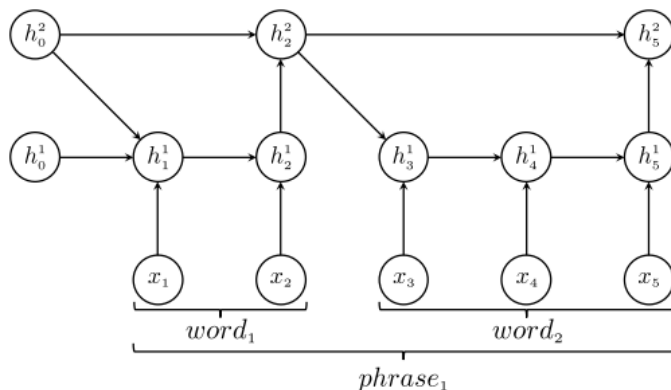
Our goal is to use hierarchical structure in data



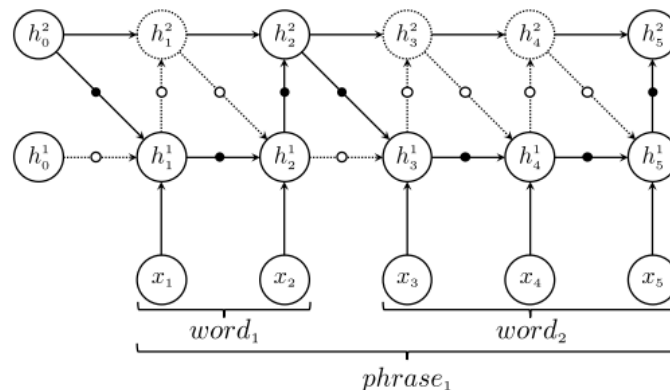
Hierarchical structure can be provided  
and used with the following architecture

# Various architectures: Hierarchical RNN

Our goal is to discover hierarchical structure in data



Hierarchical structure can be provided and used



Hierarchical structure can be learned  
The boundary detector is the key

# Various architectures: Hierarchical RNN

Finally some formulas!

$$\mathbf{h}_t^\ell, \mathbf{c}_t^\ell, z_t^\ell = f_{\text{HM-LSTM}}^\ell(\mathbf{c}_{t-1}^\ell, \mathbf{h}_{t-1}^\ell, \mathbf{h}_t^{\ell-1}, \mathbf{h}_t^{\ell+1}, z_{t-1}^\ell, z_t^{\ell-1}). \quad (1)$$

Here,  $\mathbf{h}$  and  $\mathbf{c}$  denote the hidden and cell states, respectively. The function  $f_{\text{HM-LSTM}}^\ell$  is implemented as follows. First, using the two boundary states  $z_{t-1}^\ell$  and  $z_t^{\ell-1}$ , the cell state is updated by:

$$\mathbf{c}_t^\ell = \begin{cases} \mathbf{f}_t^\ell \odot \mathbf{c}_{t-1}^\ell + \mathbf{i}_t^\ell \odot \mathbf{g}_t^\ell & \text{if } z_{t-1}^\ell = 0 \text{ and } z_t^{\ell-1} = 1 \text{ (UPDATE)} \\ \mathbf{c}_{t-1}^\ell & \text{if } z_{t-1}^\ell = 0 \text{ and } z_t^{\ell-1} = 0 \text{ (COPY)} \\ \mathbf{i}_t^\ell \odot \mathbf{g}_t^\ell & \text{if } z_{t-1}^\ell = 1 \text{ (FLUSH),} \end{cases} \quad (2)$$

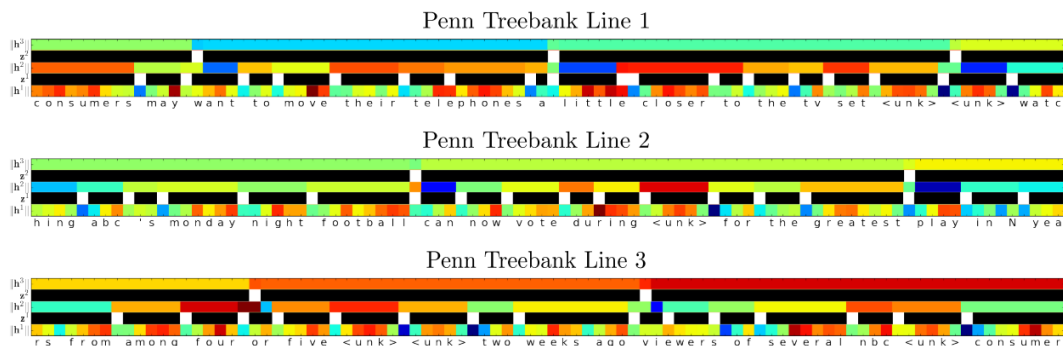
and then the hidden state is obtained by:

$$\mathbf{h}_t^\ell = \begin{cases} \mathbf{h}_{t-1}^\ell & \text{if COPY,} \\ \mathbf{o}_t^\ell \odot \tanh(\mathbf{c}_t^\ell) & \text{otherwise.} \end{cases} \quad (3)$$

But boundary states are discrete. How can we optimize them?

# Various architectures: Hierarchical RNN

- Straight-through estimator
  - REINFORCE
- ← Simpler but biased approach used by the authors



Learned hierarchical structure

Also: the slope annealing trick

*Increase slope of a sigmoid during training  
to turn it into a step function*

# Other issues with RNNs

- Technical details e.g. Batch normalization or Dropout
- Memory mechanism
- Many applied problems solved with RNNs
- Theoretical view on RNNs
  - They are Turing complete

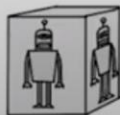
# Take-home messages

- Problems with gradients are intrinsic to RNN architecture, but can be fixed with LSTM or GRU
- Embedding is a fundamental concept: it is representation learning for RNNs
- We still can solve various complex problems using various architectures

RNN



LSTM



Bi-LSTM

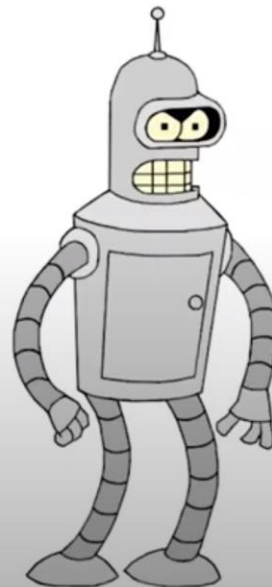
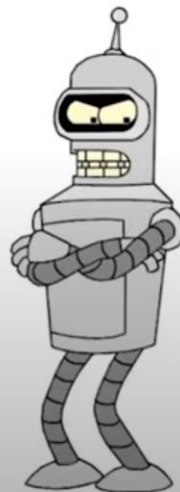


Encoder –  
Decoder



**This week**

Transformer  
Encoder –  
Attention –  
Decoder



**Next week**

# Sources

- <https://github.com/kjw0612/awesome-rnn#blogs>
- Recurrent Neural Networks | in Deep learning course by MIT 6.S191
- Coursera course on Sequence models  
<https://www.coursera.org/learn/nlp-sequence-models>
- <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>