Recurrent neural networks

Ekaterina Lobacheva

lobacheva.tjulja@gmail.com

JetBrains

Saint Petersburg, 2016

Outline

- RNN: motivation and definition
- Training: backpropagation through time
- Vanishing and exploding gradients
- LSTM, GRU, uRNN
- Bidirectional RNN

- Examples
- Tips and tricks

Motivation

Sequence input:

- Sentiment analysis

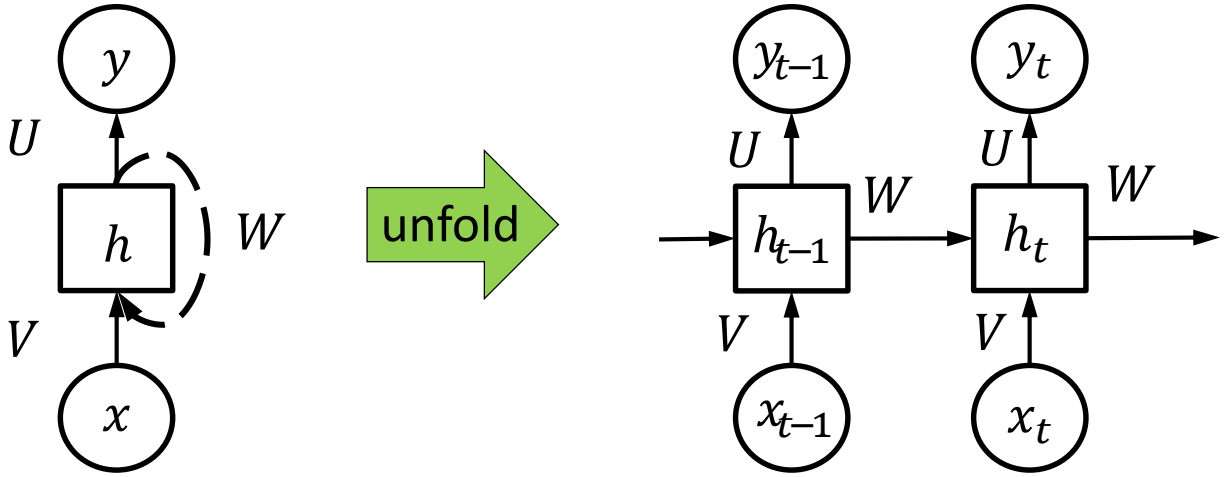
Sequence output:

- Image captioning

Sequence input and output:

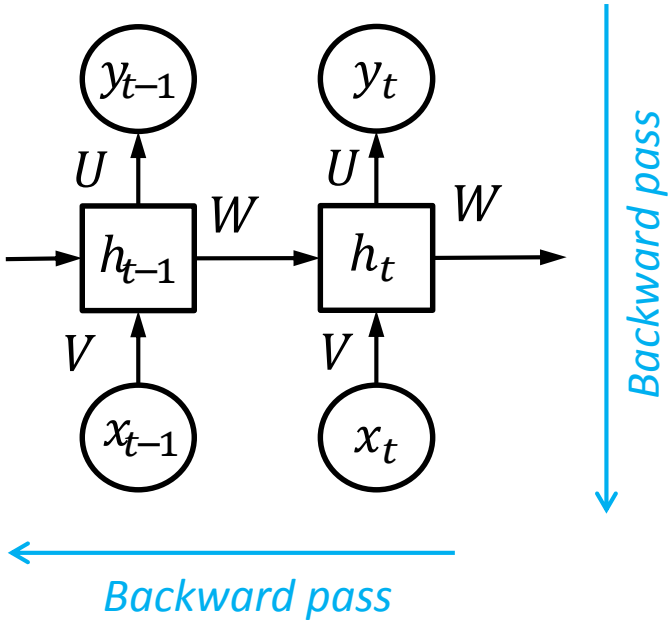
- POS tagging
- Language model
- Handwriting generation
- Speech to text / text to speech
- Machine Translation

Recurrent neural network



$$h_t = g(Vx_t + Wh_{t-1} + b_h)$$
$$y_t = f(Uh_t + b_y)$$

Backpropagation through time



$$h_t = g(Vx_t + Wh_{t-1} + b_h)$$
$$y_t = f(Uh_t + b_y)$$

Loss function:

$$F(y, a) = \sum_{t=1}^{T} F_t(y_t, a_t)$$

$$\frac{\partial F_\tau}{\partial h_t} = \frac{\partial F_\tau}{\partial h_\tau} \prod_{k=t}^{\tau-1} \frac{\partial h_{k+1}}{\partial h_k} = \frac{\partial F_\tau}{\partial h_\tau} \prod_{k=t}^{T-1} diag\big(g'(\ldots)\big)W$$

exploding or vanishing gradients

no long-range dependencies

RNN: modifications

- Gradient clipping (Mikolov, 2012; Pascanu et al., 2012)
- Gated models:
  - LSTM (Hochreiter and Schmidhuber, 1997)
  - GRU (Cho et al., 2014)
  - SCRN (Mikolov et al., 2015)

- Orthogonal and unitary matrices in RNN (Saxe et al., 2014; Le et al., 2015; Arjovsky and Shah and Bengio, 2016)

- Echo State Networks (Jaeger and Haas, 2004; Jaeger, 2012)
- Second-order optimization (Martens, 2010; Martens & Sutskever, 2011)
- Regularization (Pascanu et al., 2012)
- Careful initialization (Sutskever et al., 2013)

Gradient clipping

Algorithm 1 Pseudo-code for norm clipping

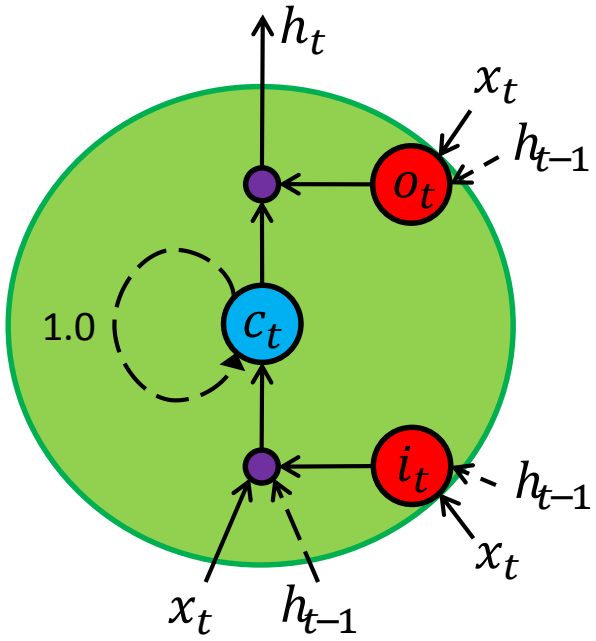
$\hat{\mathbf{g}} \leftarrow \frac{\partial \mathcal{E}}{\partial \theta}$
if $\|\hat{\mathbf{g}}\| \geq threshold$ **then**
  $\hat{\mathbf{g}} \leftarrow \frac{threshold}{\|\hat{\mathbf{g}}\|}\hat{\mathbf{g}}$
end if

threshold: average norm over a sufficiently large number of updates

[Pascanu et al., 2012]

Long short term memory:

Version 0



Gate values in [0,1]

i_t, o_t - input/output gates

c_t - memory

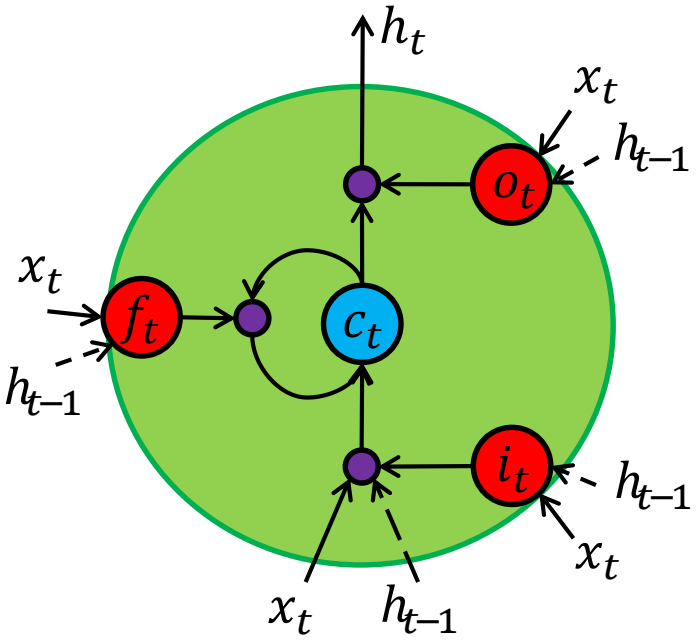
$$i_t = \sigma(V_i x_t + W_i h_{t-1} + b_i)$$
$$o_t = \sigma(V_o x_t + W_o h_{t-1} + b_o)$$

$$c_t = c_{t-1} + i_t \cdot g(V_c x_t + W_c h_{t-1} + b_c)$$
$$h_t = o_t \cdot g(c_t)$$

$$\frac{\partial h_{k+1}}{\partial h_k} \Rightarrow \frac{\partial c_{k+1}}{\partial c_k} = 1 \Rightarrow$$ Gradient doesn't vanish

Long short term memory:

Version 1



Gate values in [0,1]

i_t, o_t, f_t - input/output/forget gates

c_t - memory

$$i_t = \sigma(V_i x_t + W_i h_{t-1} + b_i)$$
$$f_t = \sigma(V_f x_t + W_f h_{t-1} + b_f)$$
$$o_t = \sigma(V_o x_t + W_o h_{t-1} + b_o)$$

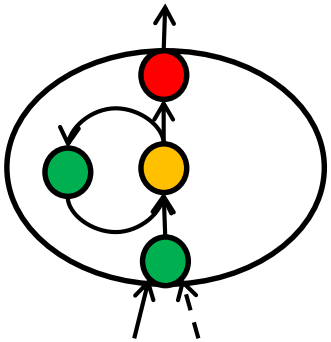
$$c_t = f_t \cdot c_{t-1} + i_t \cdot g(V_c x_t + W_c h_{t-1} + b_c)$$
$$h_t = o_t \cdot g(c_t)$$

$$\frac{\partial h_{k+1}}{\partial h_k} \Rightarrow \frac{\partial c_{k+1}}{\partial c_k} = f_{k+1} \Rightarrow$$ High initial b_f

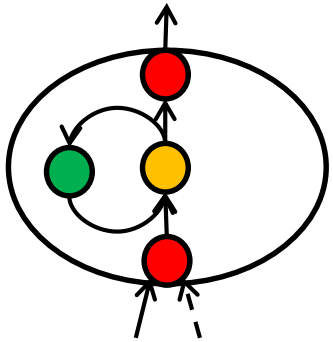
Long short term memory:

Examples

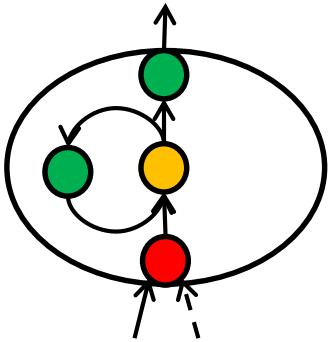
Captures info



Keeps info



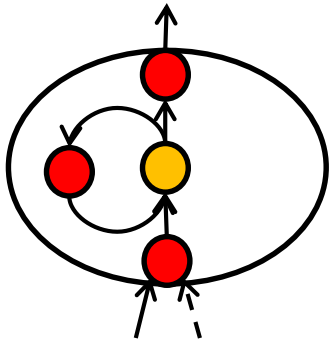
Releases info



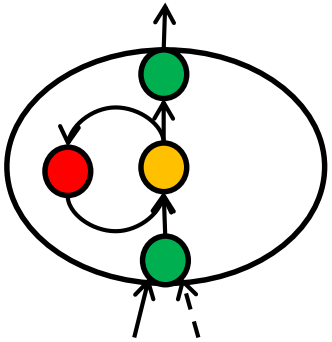
- gate is close

- gate is open

Erases info



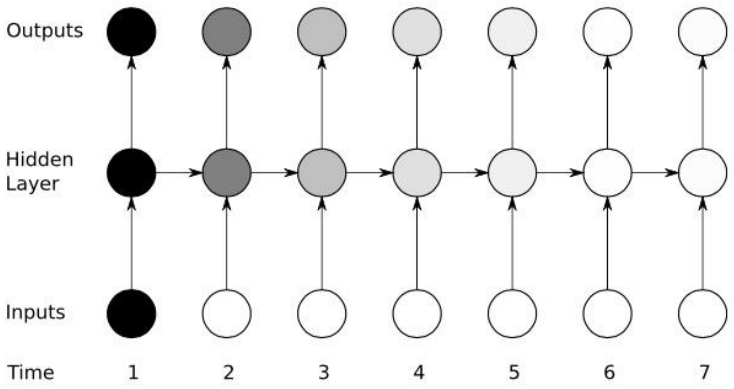
= RNN



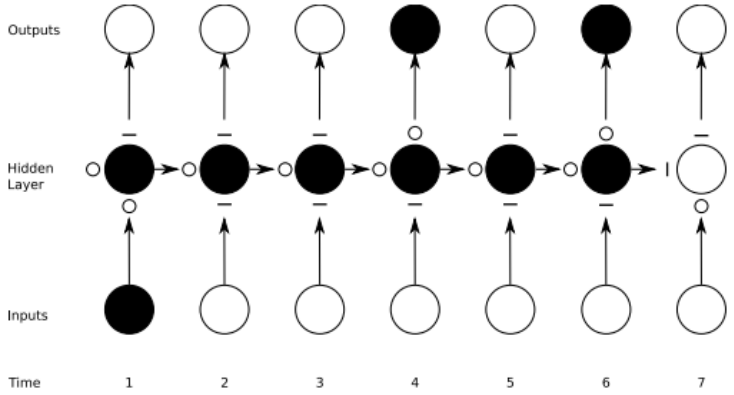
Long short term memory:

Examples

RNN



LSTM

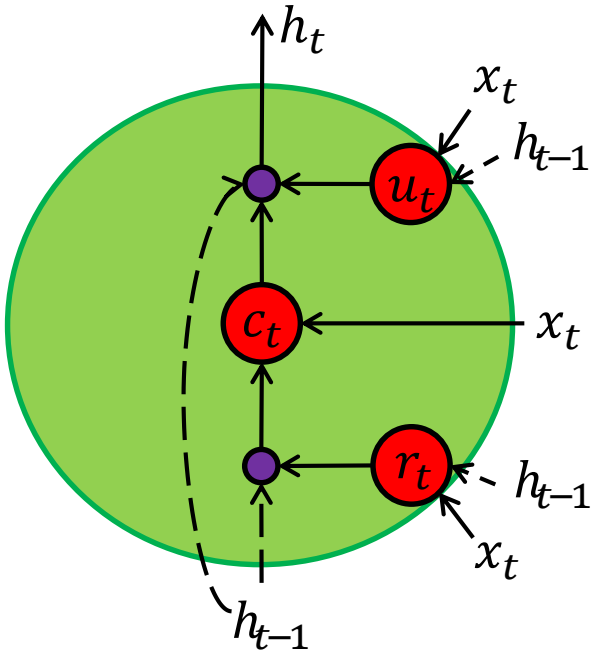


– - gate is close

○ - gate is open

[Graves, 2012]

Gated Recurrent Unit



Gate values in [0,1]

r_t, u_t - reset/update gates

$$u_t = \sigma(V_u x_t + W_u h_{t-1} + b_u)$$
$$r_t = \sigma(V_r x_t + W_r h_{t-1} + b_r)$$

$$c_t = g(V_c x_t + W_c(h_{t-1} \cdot r_t))$$
$$h_t = (1 - u_t) \cdot c_t + u_t \cdot h_{t-1}$$

$$\frac{\partial h_{k+1}}{\partial h_k} = u_{k+1} + (1 - u_{k+1}) \cdot \frac{\partial c_{k+1}}{\partial h_k}$$ ⟹ High initial b_u

Orthogonal and unitary matrices

$$\frac{\partial F_T}{\partial h_t} = \frac{\partial F_T}{\partial h_T}\prod_{k=t}^{T-1}\frac{\partial h_{k+1}}{\partial h_k} = \frac{\partial F_T}{\partial h_T}\prod_{k=t}^{T-1} diag\big(g'(\dots)\big)W$$

$$\left\|\frac{\partial F_T}{\partial h_t}\right\| = \left\|\frac{\partial F_T}{\partial h_T}\prod_{k=t}^{T-1} DW\right\| \leq \left\|\frac{\partial F_T}{\partial h_T}\right\|\prod_{k=t}^{T-1}\|DW\| =$$

W → Orthogonal or unitary → $= \left\|\frac{\partial F_T}{\partial h_T}\right\|\prod_{k=t}^{T-1}\|D\| =$

D → ReLU → $= \left\|\frac{\partial F_T}{\partial h_T}\right\|$

Orthogonal and unitary matrices

Initialize recurrent weights with the identity matrix

[Le et al., 2015]

Regularization:

$$\Omega = \sum_k \Omega_k = \sum_k \left(\frac{\left\| \frac{\partial \mathcal{E}}{\partial \mathbf{x}_{k+1}} \frac{\partial \mathbf{x}_{k+1}}{\partial \mathbf{x}_k} \right\|}{\left\| \frac{\partial \mathcal{E}}{\partial \mathbf{x}_{k+1}} \right\|} - 1 \right)^2$$

[Pascanu et al., 2012]

uRNN

$$W = D_3 R_2 F^{-1} D_2 \Pi R_1 F D_1$$

- $\mathbf{D}$, a diagonal matrix with $\mathbf{D}_{j,j} = e^{iw_j}$, with parameters $w_j \in \mathbb{R}$,
- $\mathbf{R} = \mathbf{I} - 2\frac{vv^*}{\|v\|^2}$, a reflection matrix in the complex vector $v \in \mathbb{C}^n$,
- $\mathbf{\Pi}$, a fixed random index permutation matrix, and
- $\mathcal{F}$ and $\mathcal{F}^{-1}$, the Fourier and inverse Fourier transforms.

Complex:

- hidden units,
- in-to-hidden
- hidden-to-hidden

$$\Rightarrow \quad o_t = f\left(U \begin{pmatrix} Re(h_t) \\ Im(h_t) \end{pmatrix} + b_o\right)$$

$$modReLU(z) = \begin{cases} (|z| + b)\frac{z}{|z|} & if\ |z| + b \geq 0 \\ 0 & if\ |z| + b < 0 \end{cases}$$

[Arjovsky et al., 2016]

uRNN

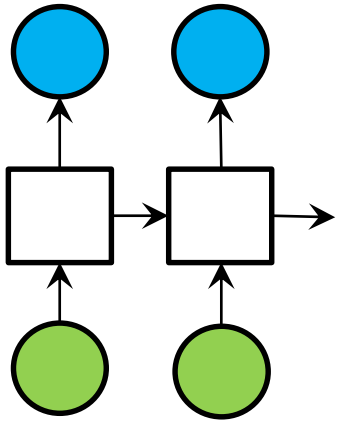
Pros:

- No vanishing or exploding gradients
- Memory: $O(n)$, time: $O(n\ logn)$
- Good parametrization: $O(n)$ parameters $\rightarrow$ more hidden units
- Very long dependencies

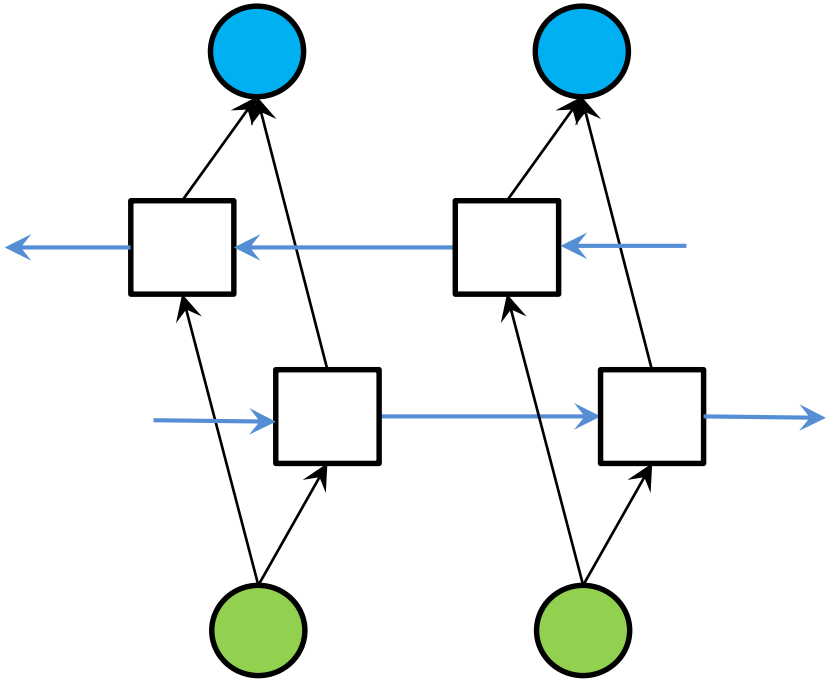
Cons:

- LSTM has stronger local dependencies

Bidirectional RNN



RNN



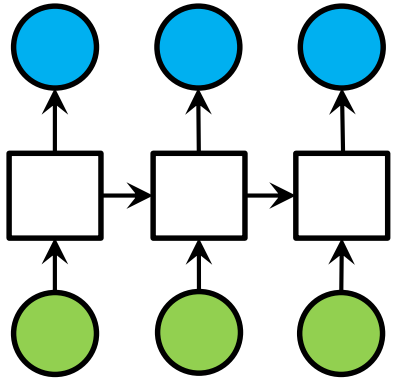
Bidirectional RNN

Examples

Sequence to sequence

Synced sequence input and output:

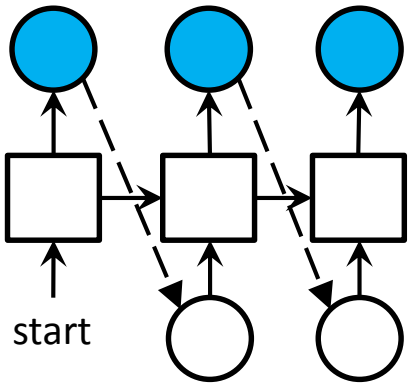
- POS tagging
- Video frames classification



Text generation

Next symbol/word

Current symbol/word



Text generation

PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

Second Lord:
They would be ruled after this chamber, and
my fair nues begun out of the fact, to be conveyed,
Whose noble souls I'll have the heart of the wars.

Clown:
Come, sir, I will make did behold your worship.

VIOLA:
I'll drink it.

Andrej Karpathy blog

Text generation

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 $X_{étale}$ we have

$$\mathcal{O}_X(\mathcal{F}) = \{morph_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

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

Lemma 0.2. *This is an integer $\mathcal{Z}$ 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 $\mathcal{U} \subset \mathcal{X}$ 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

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

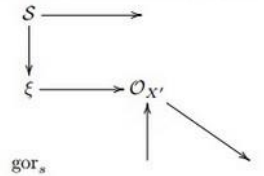
be a morphism of algebraic spaces over S and Y.

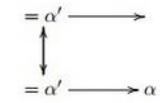
Proof. Let X be a nonzero scheme of X. Let X be an algebraic space. Let $\mathcal{F}$ be a quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent

(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. □

This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram





$\mathrm{Spec}(K_\psi)$ Mor_{Sets} $\mathrm{d}(\mathcal{O}_{\mathcal{X}_{X/k}}, \mathcal{G})$

is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type f_*. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

□

Proof. We have see that $X = \mathrm{Spec}(R)$ and $\mathcal{F}$ is a finite type representable by algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then the cohomology of X is an open neighbourhood of U. □

Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas **??**.
A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The functor $\mathcal{F}$ is a "field

$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \quad \text{-1}(\mathcal{O}_{X_{étale}}) \longrightarrow \mathcal{O}_{X_t}^{-1}\mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\eta}^{\overline{v}})$$

is an isomorphism of covering of $\mathcal{O}_{X_i}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.
The property $\mathcal{F}$ is a disjoint union of Proposition **??** and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type over S.
If $\mathcal{F}$ is a scheme theoretic image points. □

If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_\lambda}$ is a closed immersion, see Lemma **??**. This is a sequence of $\mathcal{F}$ is a similar morphism.

[Andrej Karpathy blog]

Text generation

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:

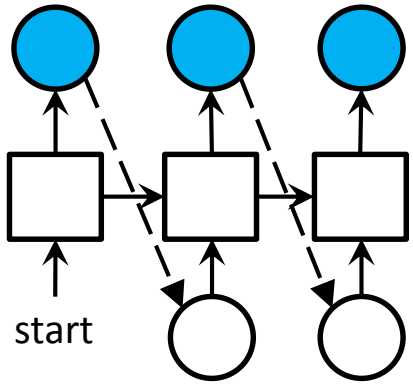
```
/* Unpack a filter field's string representation from user-space
 * buffer. */
char *audit_unpack_string(void **bufp, size_t *remain, size_t len)
{
  char *str;
  if (!*bufp || (len == 0) || (len > *remain))
    return ERR_PTR(-EINVAL);
  /* Of the currently implemented string fields, PATH_MAX
 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

[Andrej Karpathy blog]

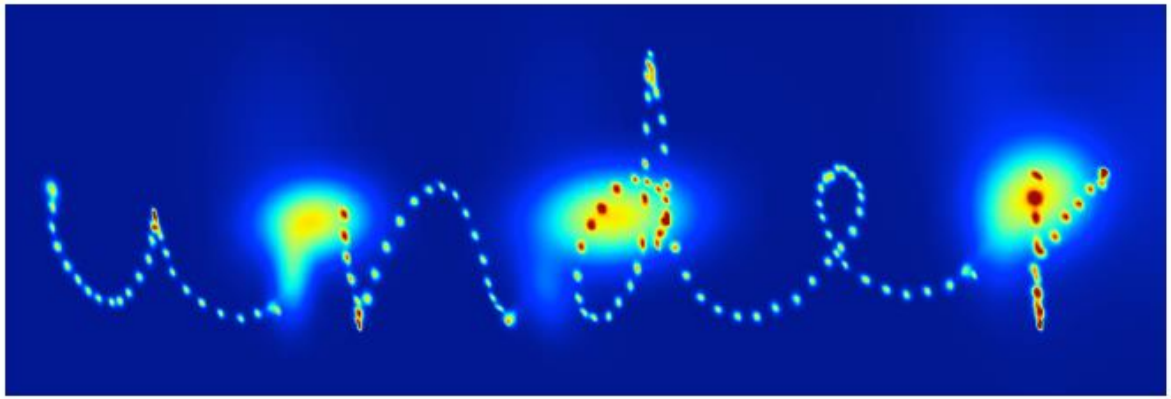
Handwriting generation:

handwriting -> handwriting

Next pen position (we predict parameters):
x1, x2 - mixture of bivariate Gaussians
x3 - Bernoulli distribution

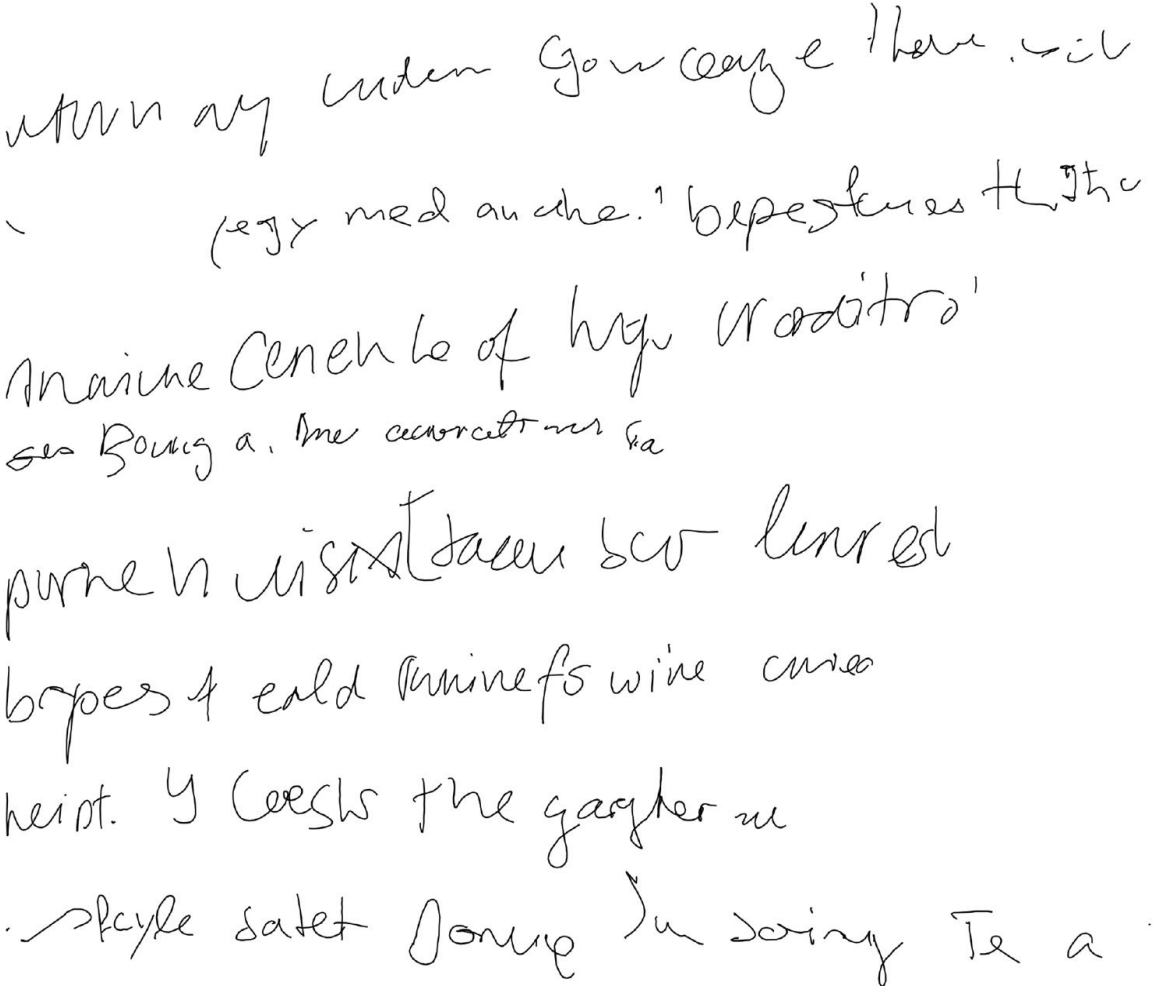


Current pen position:
x1, x2 – pen offset
x3 – is it end of the stroke



Handwriting generation:

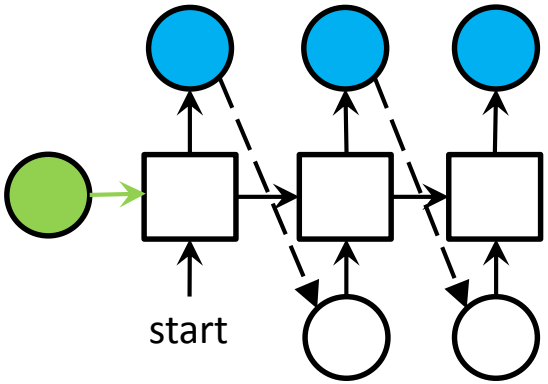
example

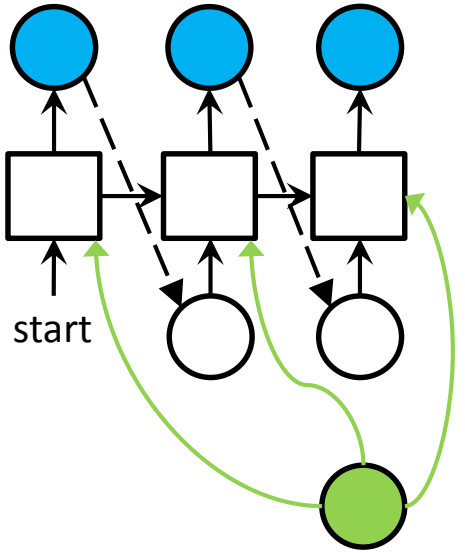


Sequence output

Sequence generation:

- Handwriting synthesis
- Image captioning





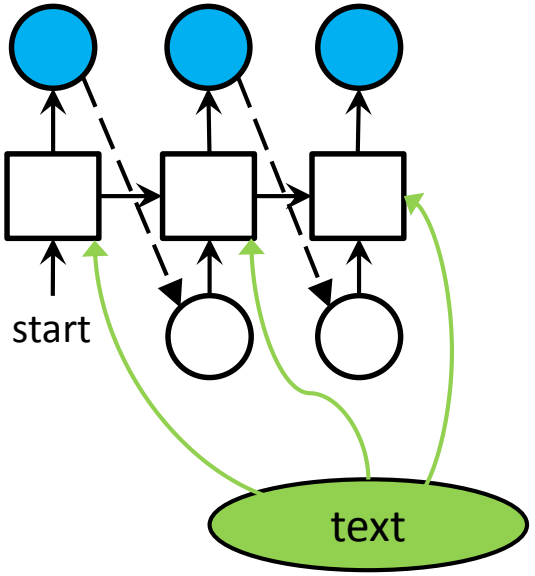
Handwriting synthesis:

text -> handwriting

Next pen position

Current pen position

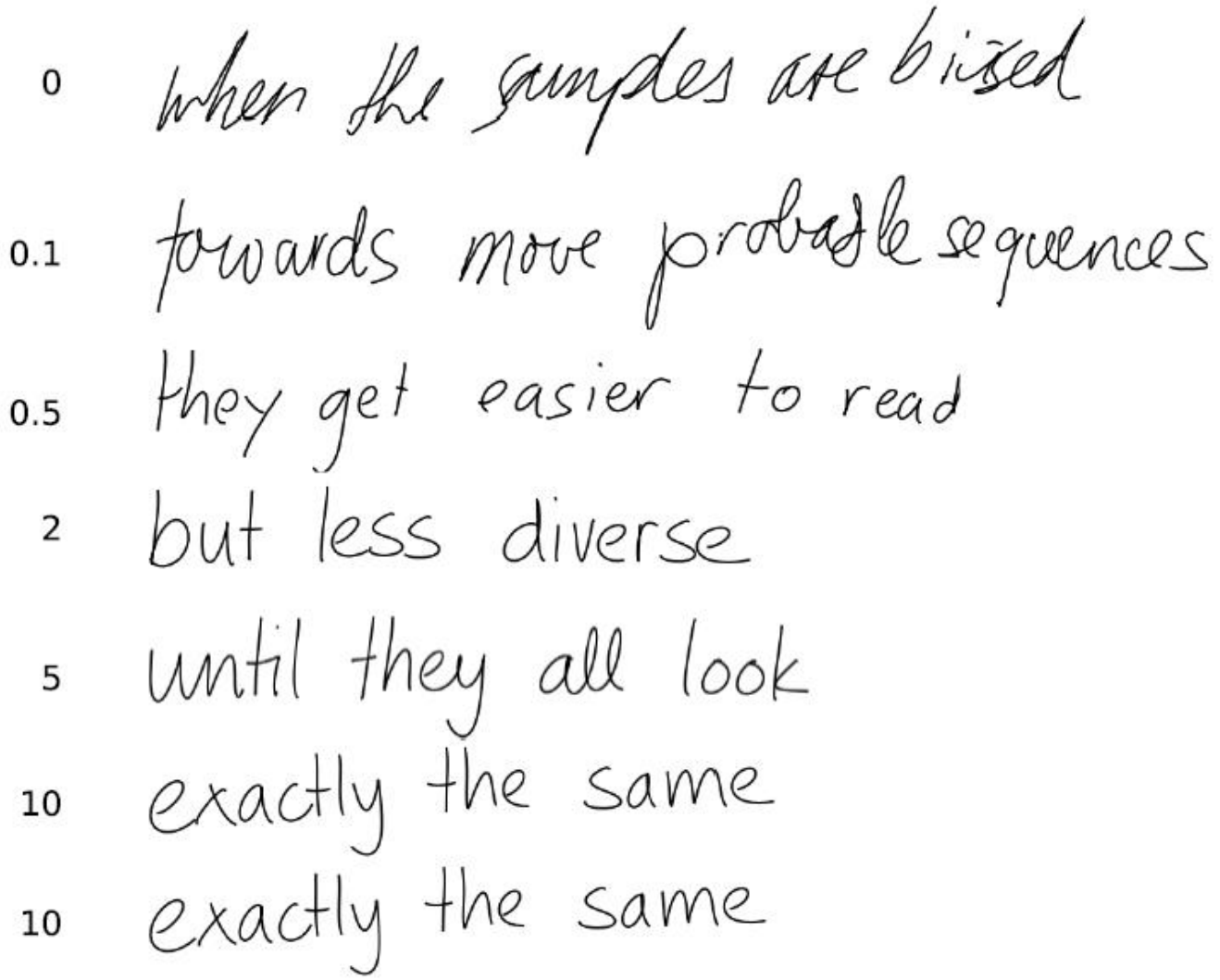
Which letter we write now



Demo

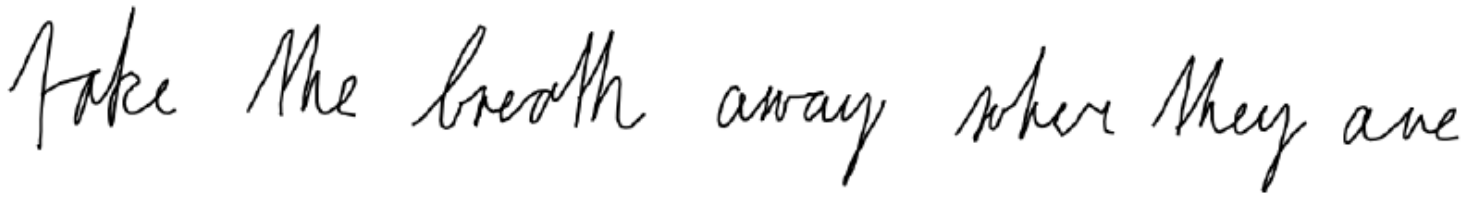
Handwriting synthesis:

biased sampling



Handwriting synthesis:

primed sampling

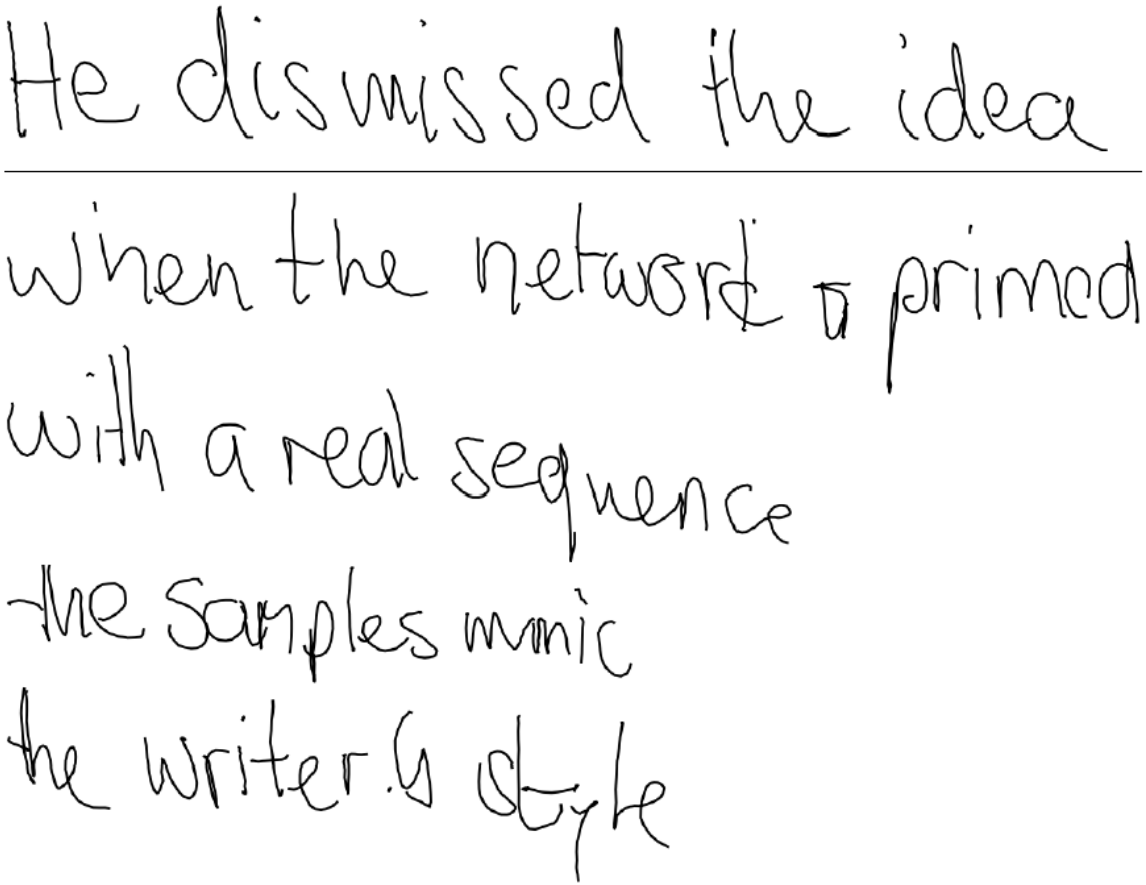


---

when the network is primed
with a real sequence
the samples mimic
the writer's style

Handwriting synthesis:

primed sampling



Handwriting synthesis:

primed and biased sampling

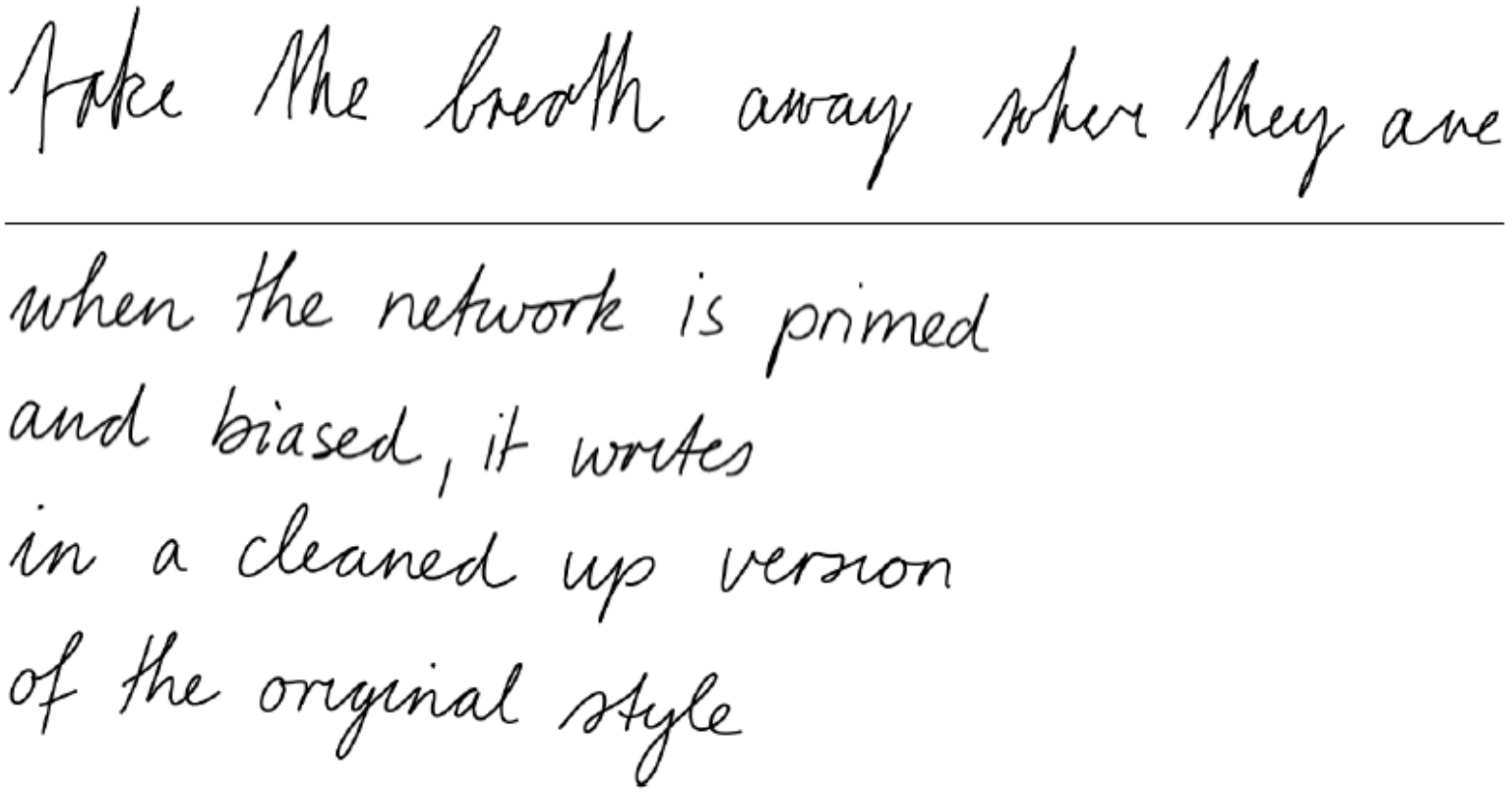
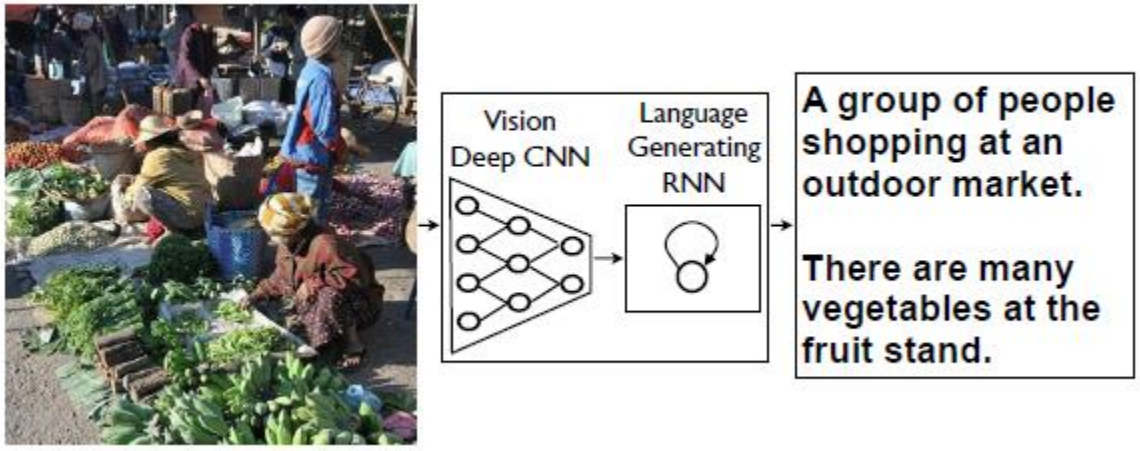


Image Caption Generation



Demo (images)

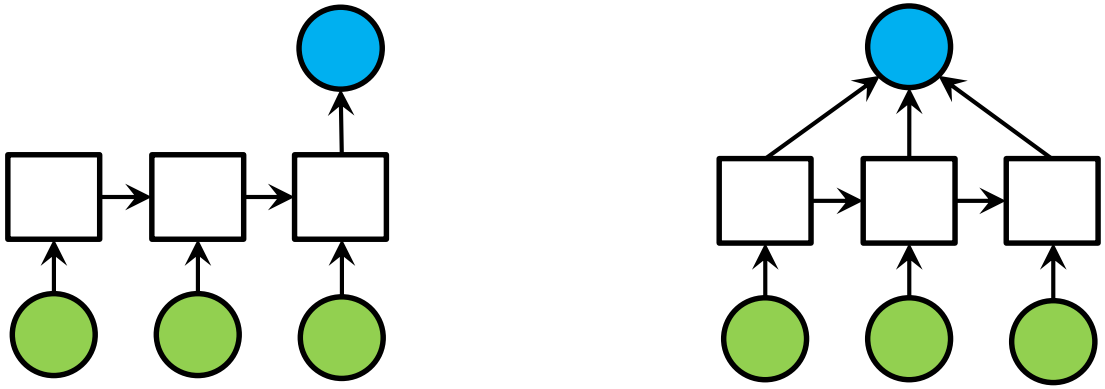
Demo (top images for test texts)

Demo (more sophisticated model)

Sequence input

Sequence classification:

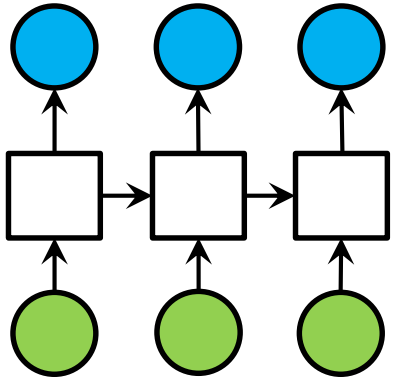
- Sentiment analysis



Sequence to sequence

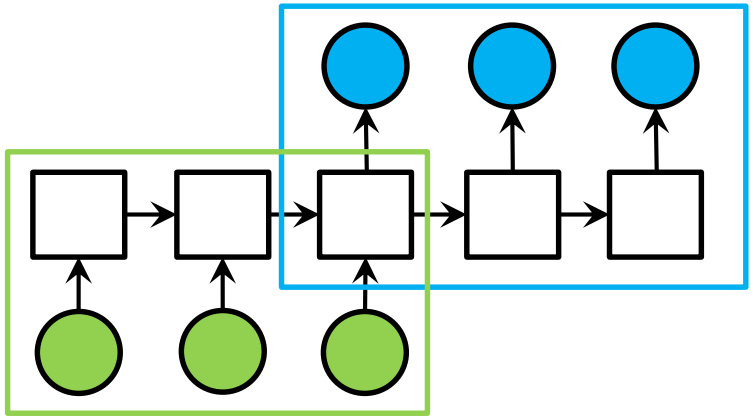
- Handwriting to text / text to handwriting
- Speech to text / text to speech

Input and output have different length!



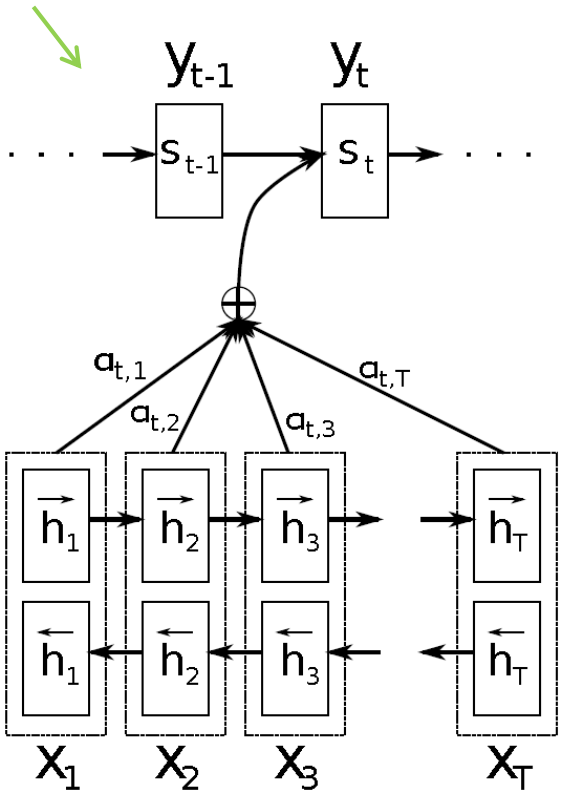
- Machine Translation

Demo with bidirectional RNN



Translation with attention

decoder RNN



$$s_i = f(s_{i-1}, y_{i-1}, c_i)$$

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$$

$$e_{ij} = a(s_{i-1}, h_j)$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

bidirectional RNN

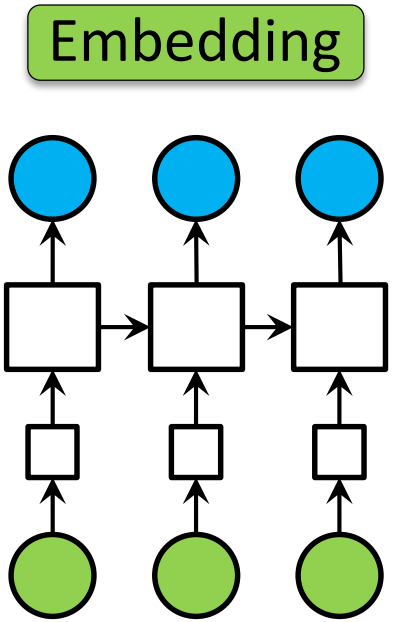
[Bahdanau et al. 2015]

Tips and tricks

Tips and tricks

Train data for text generation:

- Sequences of the same length
- Sequences of different lengths and a mask (sentences)
- Sequences of the same length and accurate initialization of hidden units



Tips and tricks

- Gradient clipping: 2 variants
- Truncated BPTT
- Numerically stable log-softmax with crossentropy

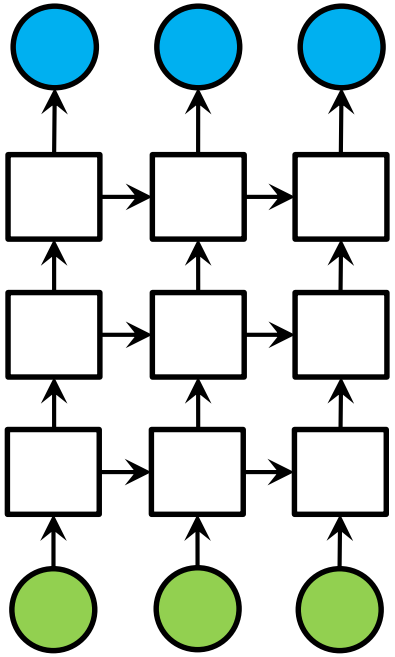
$$p_j = softmax(x)_j = \frac{\exp(x_j)}{\sum_k \exp(x_k)} \qquad L = -\sum_j a_j \log(p_j)$$

$$x_j = x_j - \max_k x_k$$

$$p_j = logsoftmax(x)_j = x_j - \log\left(\sum_k \exp(x_k)\right)$$

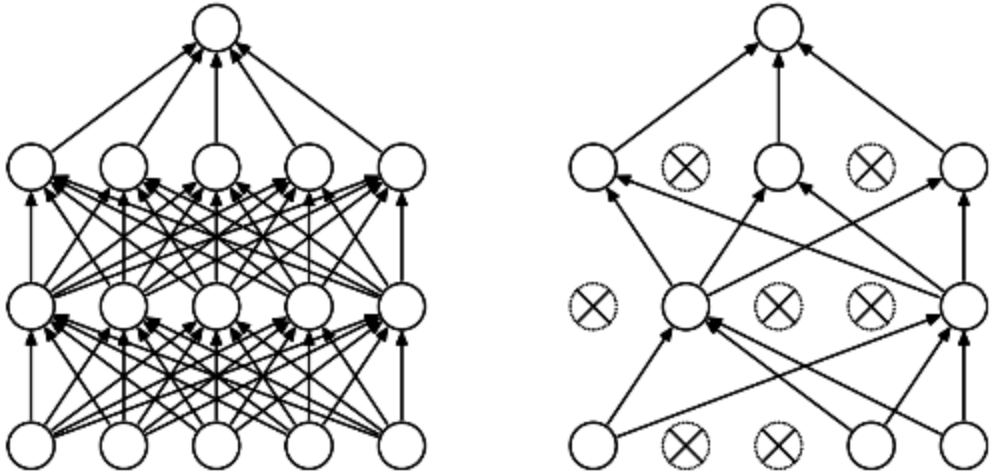
$$L = -\sum_j a_j \, p_j$$

Deep RNN



Dropout

- p – probability of dropping unit
- Train: for each case a new thinned network is sampled and trained.
- Test: net without dropout, but $w = pw$
- Net can be seen as a collection of exponential number of thinned neural networks.

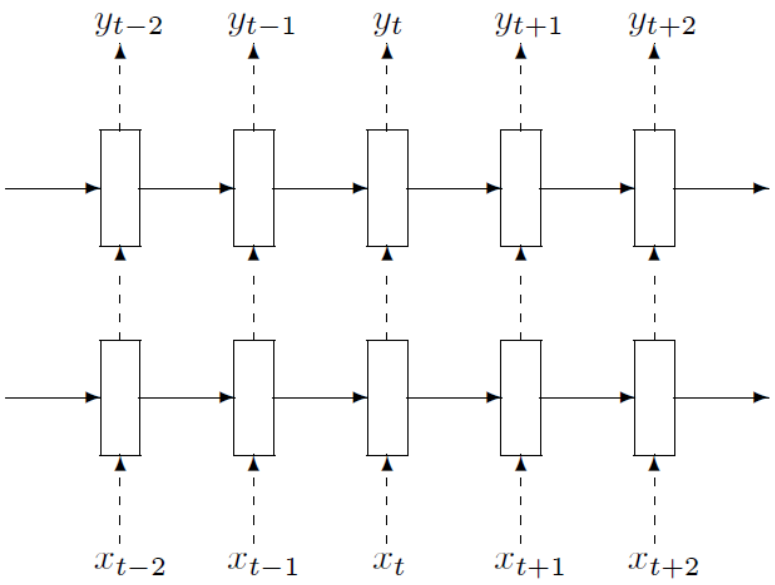


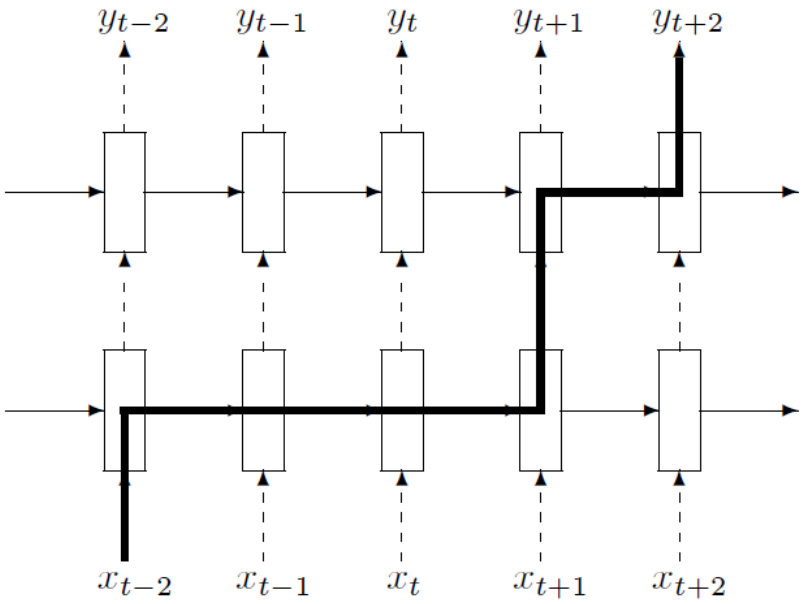
(a) Standard Neural Net (b) After applying dropout.

[Srivastava et al., 2014]

Dropout and BN for RNN

Only to non-recurrent connections!





[Zaremba et al., 2015]

[Laurent et al., 2016]

Reference

Theory

Hochreiter, Sepp, and Jurgen Schmidhuber. Long short-term memory // Neural computation 9.8: 1735-1780. 1997.

F. A. Gers, J. Schmidhuber, F. Cummins. Learning to Forget: Continual Prediction with LSTM // Tech. Rep. No. IDSIA-01-99, 1999.

F. A. Gers. Long Short-Term Memory in Recurrent Neural Networks // PhD thesis, Department of Computer Science, Swiss Federal Institute of Technology, Lausanne, EPFL, Switzerland, 2001.

Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, Jürgen Schmidhuber. LSTM: A Search Space Odyssey.

Kyunghyun Cho et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation// EMNLP, 2014.

Mike Schuster and Kuldip K. Paliwal . Bidirectional Recurrent Neural Networks // IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL. 45, NO. 11, 1997

Razvan Pascanu, Tomas Mikolov, Yoshua Bengio. On the difficulty of training Recurrent Neural Networks // ICML, 2013.

Tomas Mikolov et al. Learning Longer Memory in Recurrent Neural Networks // ICLR, 2015.

Quoc V. Le, Navdeep Jaitly, Geoffrey E. Hinton. A Simple Way to Initialize Recurrent Networks of Rectified Linear Units // arXiv, 2015.

Martin Arjovsky, Amar Shah, Yoshua Bengio. Unitary Evolution Recurrent Neural Networks // ICML, 2016.

Reference

Theory

Nitish Srivastava et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting // JMLR, 2014.

Wojciech Zaremba, Ilya Sutskever, Oriol Vinyals. Recurrent Neural Network Regularization // arXiv, 2014.

Sergey Ioffe, Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift // ICML, 2015.

César Laurent et al. Batch Normalized Recurrent Neural Networks // ICASSP, 2016.

Tim Cooijmans et al. Recurrent Batch Normalization // arXiv, 2016.

A list of resources dedicated to RNNs: Awesome Recurrent Neural Networks

Andrej Karpathy . The Unreasonable Effectiveness of Recurrent Neural Networks // blogpost.

Andrej Karpathy, Justin Johnson, Li Fei-Fei. Visualizing and Understanding Recurrent Networks // ICLR, 2016.

Reference: examples

Sequence generation

- **Character-wise text generation with Multiplicative RNN**

  Ilya Sutskever, James Martens, and Geoffrey Hinton. Generating Text with Recurrent Neural Networks // ICML 2011.

  demo, slides

- **Word-wise text generation with RNN (RNN vs n-grams)**

  Mikolov Tomá, Karafiát Martin, Burget Luká, Èernocký Jan, Khudanpur Sanjeev. Recurrent neural network based language model. // Proceedings of the 11th Annual Conference of the International Speech Communication Association (INTERSPEECH 2010).

  Mikolov Tomá. Statistical Language Models based on Neural Networks // PhD thesis, Brno University of Technology, 2012.

  lib+demo

- **Both character and word-wise text generation + handwritten generation + handwritten synthesis (all with LSTM)**

  A. Graves. Generating Sequences With Recurrent Neural Networks.

  slides, handwritten synthesis demo

Reference: examples

Sequence translation

Ilya Sutskever, Oriol Vinyals, Quoc Le. Sequence to Sequence Learning with Neural Networks // NIPS 2014

K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation // EMNLP 2014.

Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate // ICLR, 2015.

demo

Image Caption Generation

O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator // *CVPR*, 2015.

Andrej Karpathy, Li Fei-Fei. Deep Visual-Semantic Alignments for Generating Image Descriptions // CVPR, 2015.

demo (images), demo (top images for test texts)

Ryan Kiros, Ruslan Salakhutdinov, Richard Zemel. Unifying Visual-Semantic Embeddings with Multimodal Neural Language Models // TACL, 2015

demo