RNNs in TensorFlow

CS 20SI:
TensorFlow for Deep Learning Research
Lecture 11
2/22/2017

Beat the World's Best at Super Smash Bros

Beating the World's Best at Super Smash Bros. Melee with Deep Reinforcement Learning (Firoiu et al., 2017) - MIT

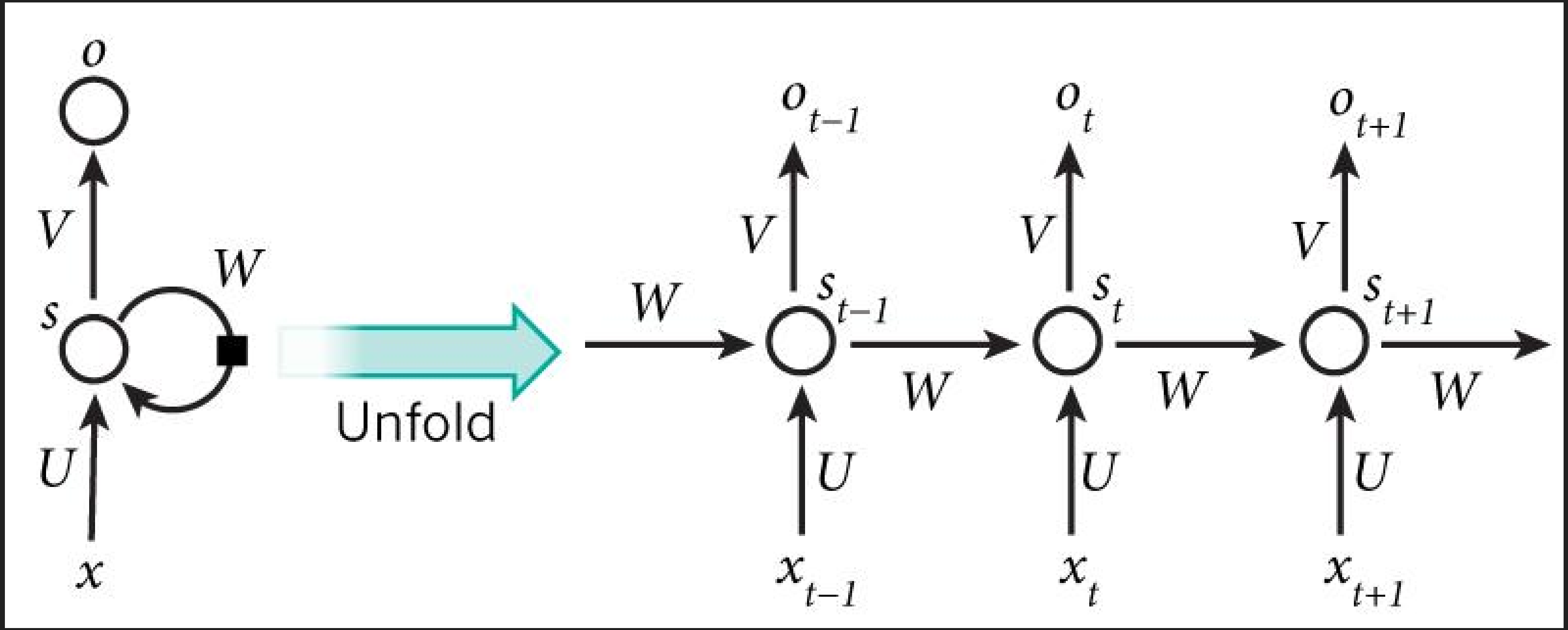
Agenda

All about RNNs

Implementation tricks & treats

Live demo of Language Modeling

From feed-forward to RNNs



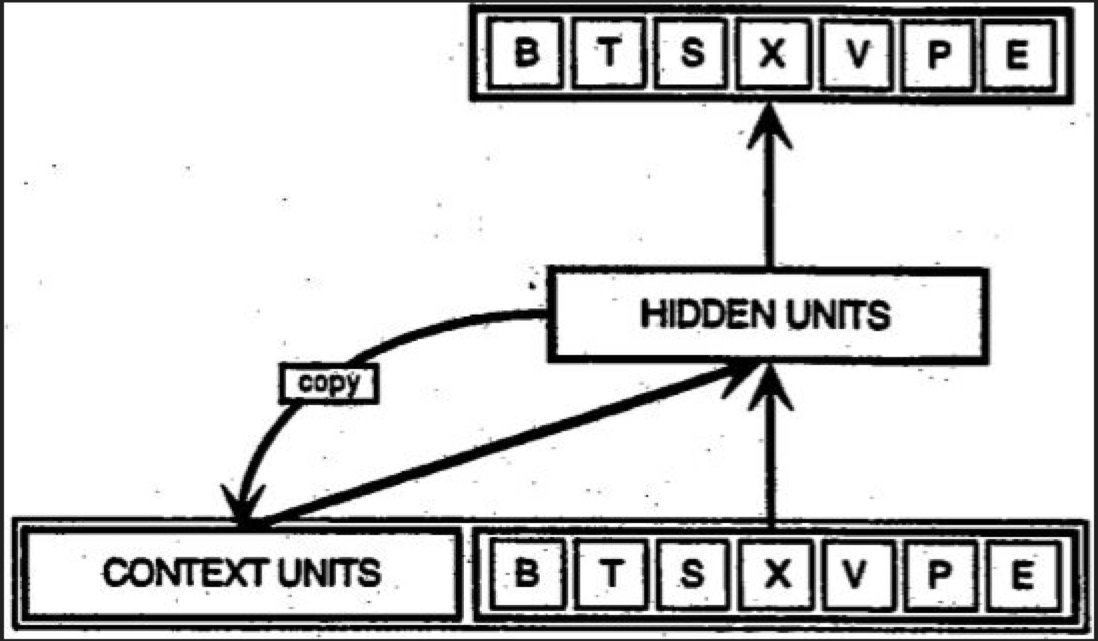
Graph by Nature

From feed-forward to RNNs

- RNNs take advantage of sequential information of data (texts, genomes, spoken words, etc.)
- Directed cycles
- All steps share weights to reduce the total number of parameters
- Form the backbone of NLP
- Can also be used for images

Simple Recurrent Neural Network (SRNN)

Introduced by Jeffrey Elman in 1990. Also known as Elman Network



Elman, Jeffrey L. "Finding structure in time." Cognitive science 14.2 (1990): 179-211.

Simple RNNs are Simple

Elman and Jordan networks are also known as "simple recurrent networks" (SRN).

Elman network[10]

$$h_t = \sigma_h(W_h x_t + U_h h_{t-1} + b_h)$$
$$y_t = \sigma_y(W_y h_t + b_y)$$

Jordan network[11]

$$h_t = \sigma_h(W_h x_t + U_h y_{t-1} + b_h)$$
$$y_t = \sigma_y(W_y h_t + b_y)$$

Variables and functions

- x_t : input vector
- h_t : hidden layer vector
- y_t : output vector
- W, U and b: parameter matrices and vector
- σ_h and σ_y: Activation functions

From Wikipedia

RNNs in the context of NLP

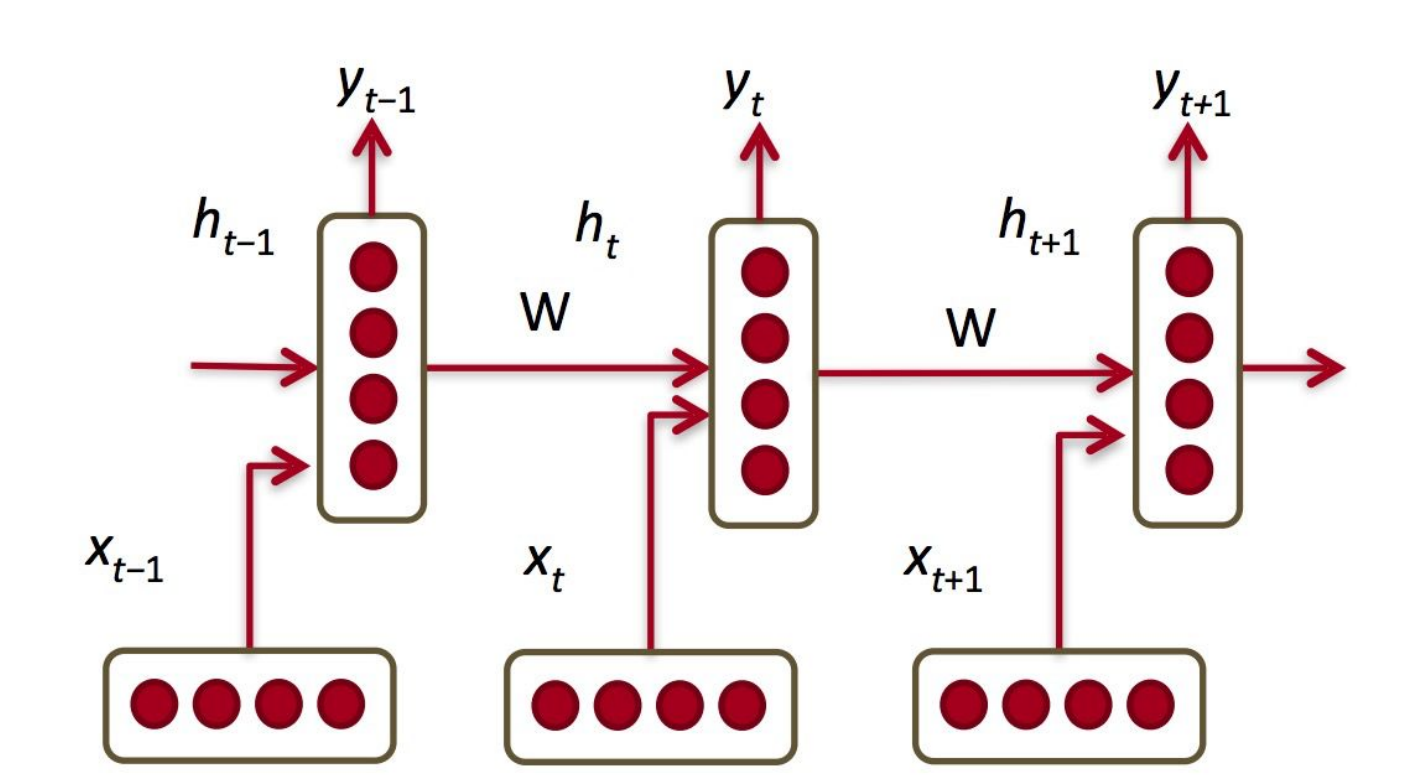


Diagram from CS 224D Slides

The problem with RNNs

- In practice, RNNs aren’t very good at capturing long-term dependencies

“I grew up in France… I speak fluent ???”
-> Needs information from way back

The rise of LSTMs

Long Short Term Memory

The rise of LSTMs

- Control how much of new input to take, how much of the previous hidden state to forget
- Closer to how humans process information
- The idea is not new. Hochreiter and Schmidhuber published the paper in 1997*

*Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.

The rise of LSTMs

$$i^{(t)} = \sigma(W^{(i)}x^{(t)} + U^{(i)}h^{(t-1)}) \qquad \text{(Input gate)}$$

$$f^{(t)} = \sigma(W^{(f)}x^{(t)} + U^{(f)}h^{(t-1)}) \qquad \text{(Forget gate)}$$

$$o^{(t)} = \sigma(W^{(o)}x^{(t)} + U^{(o)}h^{(t-1)}) \qquad \text{(Output/Exposure gate)}$$

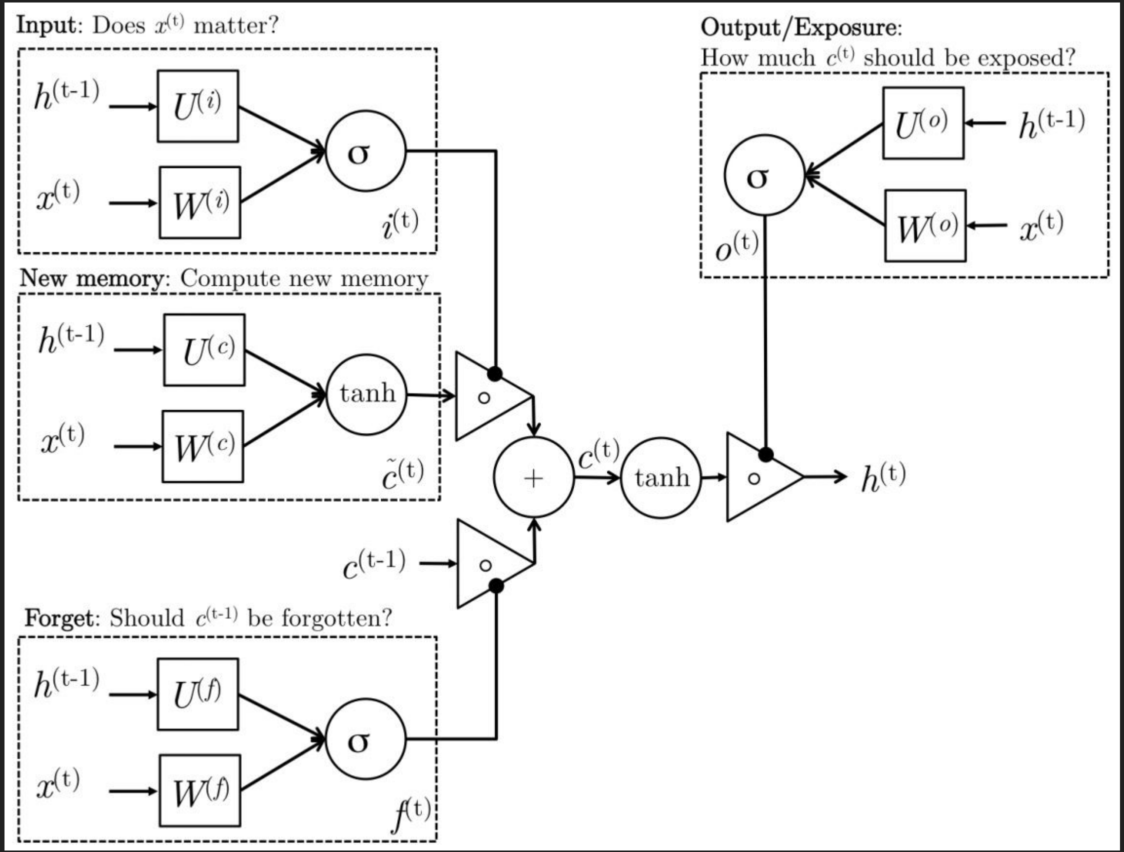
$$\tilde{c}^{(t)} = \tanh(W^{(c)}x^{(t)} + U^{(c)}h^{(t-1)}) \qquad \text{(New memory cell)}$$

$$c^{(t)} = f^{(t)} \circ \tilde{c}^{(t-1)} + i^{(t)} \circ \tilde{c}^{(t)} \qquad \text{(Final memory cell)}$$

$$h^{(t)} = o^{(t)} \circ \tanh(c^{(t)})$$

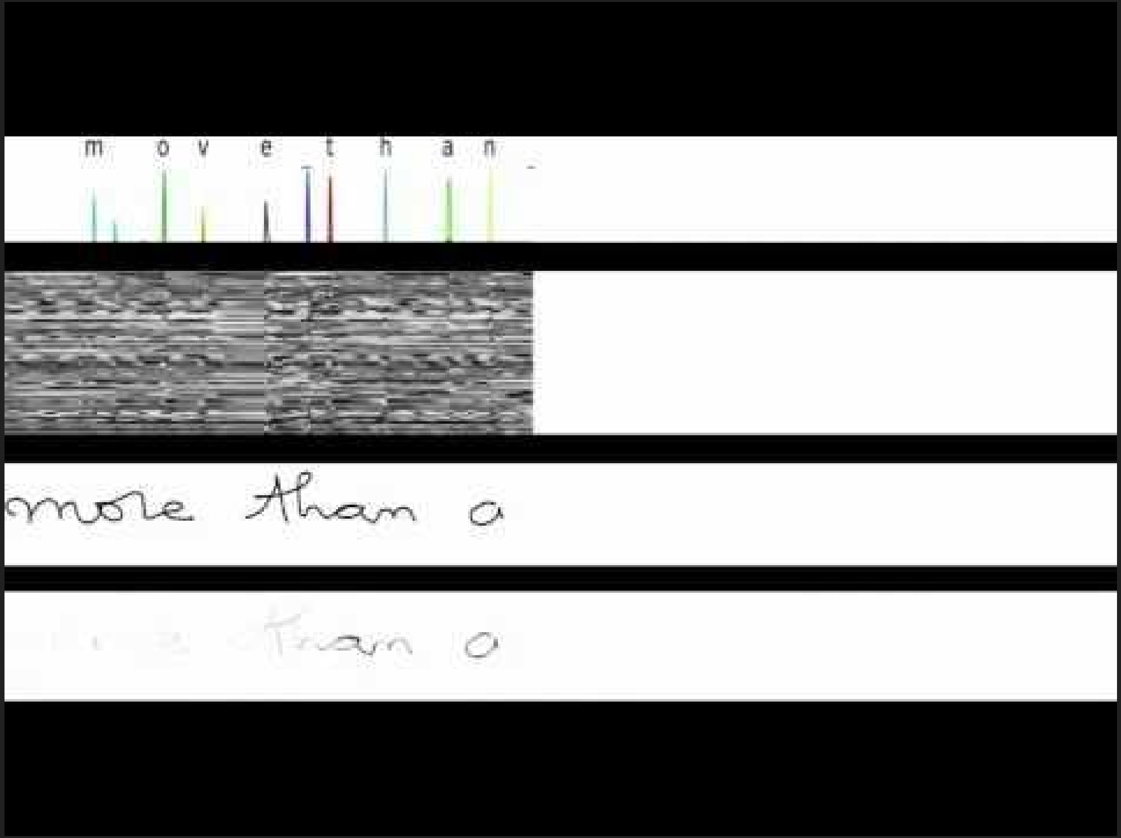
From CS 224D Lecture Note

The rise of LSTMs



From CS 224D Lecture Note

The rise of LSTMs



Visualization of LSTM in action by Alex Graves (DeepMind)

LSTMs vs GRUs

People find LSTMs work well, but unnecessarily complicated, so they introduced GRUs

GRUs (Gated Recurrent Units)

Two most widely used gated recurrent units

Gated Recurrent Unit
[Cho et al., EMNLP2014;
Chung, Gulcehre, Cho, Bengio, DLUFL2014]

$$h_t = \boxed{u_t \odot \tilde{h}_t + (1 - u_t) \odot h_{t-1}}$$
$$\tilde{h} = \tanh(W[x_t] + U(r_t \odot h_{t-1}) + b)$$
$$u_t = \sigma(W_u[x_t] + U_u h_{t-1} + b_u)$$
$$r_t = \sigma(W_r[x_t] + U_r h_{t-1} + b_r)$$

Long Short-Term Memory
[Hochreiter & Schmidhuber, NC1999;
Gers, Thesis2001]

$$h_t = o_t \odot \tanh(c_t)$$
$$c_t = \boxed{f_t \odot c_{t-1} + i_t \odot \tilde{c}_t}$$
$$\tilde{c}_t = \tanh(W_c[x_t] + U_c h_{t-1} + b_c)$$
$$o_t = \sigma(W_o[x_t] + U_o h_{t-1} + b_o)$$
$$i_t = \sigma(W_i[x_t] + U_i h_{t-1} + b_i)$$
$$f_t = \sigma(W_f[x_t] + U_f h_{t-1} + b_f)$$

From CS 224D Lecture Note

GRUs (Gated Recurrent Units)

- Computationally less expensive
- Performance on par with LSTMs*

*Chung, Junyoung, et al. "Empirical evaluation of gated recurrent neural networks on sequence modeling." arXiv preprint arXiv:1412.3555 (2014).

What can RNNs do?

Language Modeling

- Allows us to measure how likely a sentence is
- Important input for Machine Translation (since high-probability sentences are typically correct)
- Can generate new text

Character-level Language Modeling

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.

Shakespeare Generator
Andrej Karpathy's blog

Character-level Language Modeling

```
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
  int error;
  if (fd == MARN_EPT) {
    /*
     * The kernel blank will coeld it to userspace.
     */
    if (ss->segment < mem_total)
      unblock_graph_and_set_blocked();
    else
      ret = 1;
    goto bail;
  }
  segaddr = in_SB(in.addr);
  selector = seg / 16;
  setup_works = true;
  for (i = 0; i < blocks; i++) {
    seq = buf[i++];
    bpf = bd->bd.next + i * search;
    if (fd) {
      current = blocked;
    }
  }
  rw->name = "Getjbbregs";
  bprm_self_clearl(&iv->version);
  regs->new = blocks[(BPF_STATS << info->historidac)] | PFMR_CLOBATHINC_SECONDS << 12;
  return segtable;
}
```

Linux Source Code Generator
Andrej Karpathy's [blog]

Character-level Language Modeling

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

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

$$S = \mathrm{Spec}(R) = U \times_X U \times_X U$$

and the comparicoly in the fibre product covering we have to prove the lemma generated by $\coprod Z \times_U U \to V$. Consider the maps M along the set of points Sch_{fppf} and $U \to U$ is the fibre category of S in U in Section, **??** and the fact that any U affine, see Morphisms, Lemma **??**. Hence we obtain a scheme S and any open subset $W \subset U$ in $Sh(G)$ such that $\mathrm{Spec}(R') \to S$ is smooth or an

$$U = \bigcup U_i \times_{S_i} U_i$$

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

To prove study we see that $\mathcal{F}|_U$ is a covering of $\mathcal{X}'$, 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 $\mathcal{O}_X$-modules on $\mathcal{C}$ as a $\mathcal{F}$-module. In particular $\mathcal{F} = U/\mathcal{F}$ we have to show that

$$\widetilde{M}^\bullet = \mathcal{I}^\bullet \otimes_{\mathrm{Spec}(k)} \mathcal{O}_{S,s} - i_X^{-1}\mathcal{F})$$

is a unique morphism of algebraic stacks. Note that

$$\mathrm{Arrows} = (Sch/S)^{opp}_{fppf}, (Sch/S)_{fppf}$$

and

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

Fake Math Doc Generator
Andrej Karpathy's blog

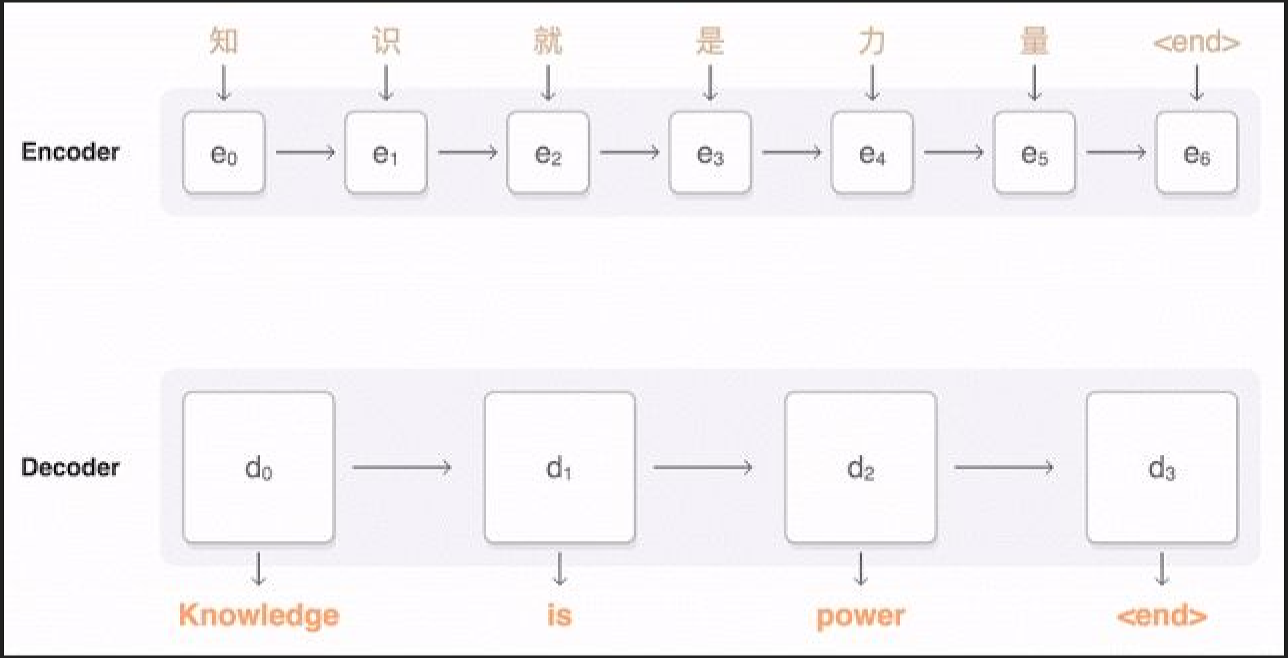
Character-level Language Modeling

Deep learning neural network architectures can be used to best developing a new architectures contros of the training and max model parametrinal Networks (RNNs) outperform deep learning algorithm is easy to out unclears and can be used to train samples on the state-of-the-art RNN more effective Lorred can be used to best developing a new architectures contros of the training and max model and state-of-the-art deep learning algorithms to a similar pooling relevants. The space of a parameter to optimized hierarchy the state-of-the-art deep learning algorithms to a simple analytical pooling relevants. The space of algorithm is easy to outions of the network are allowed at training and many dectional representations are allow develop a groppose a network by a simple model interact that training algorithms to be the activities to maximul setting, ...

Fake Arvix Abstracts Generator

We'll build this!!!!

Machine Translation



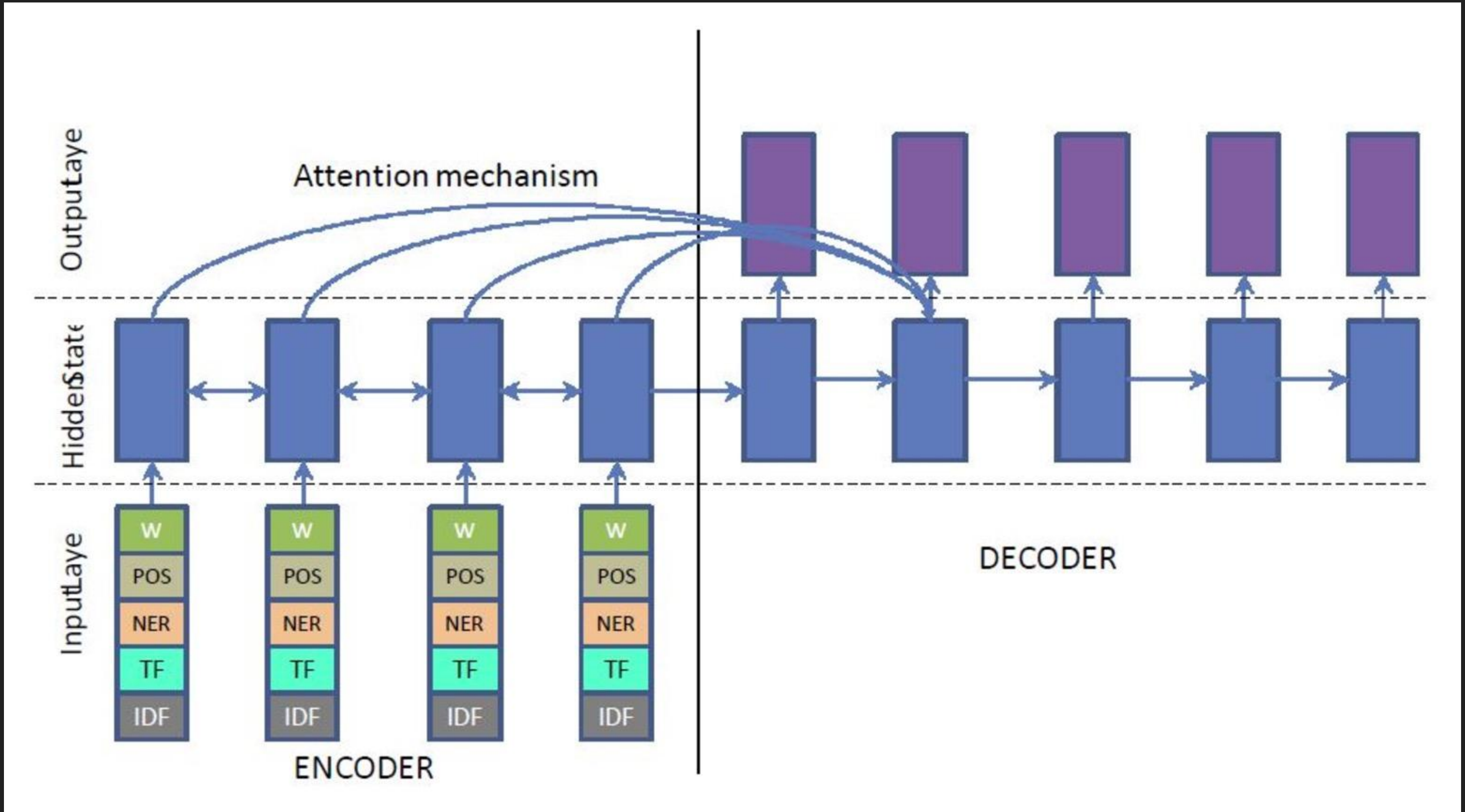
Google Neural Machine Translation
(Google Research's blog)

Machine Translation

Input sentence:	*Translation (PBMT):*	*Translation (GNMT):*	*Translation (human):*
李克強此行將啟動中加總理年度對話機制，與加拿大總理杜魯多舉行兩國總理首次年度對話。	Li Keqiang premier added this line to start the annual dialogue mechanism with the Canadian Prime Minister Trudeau two prime ministers held its first annual session.	Li Keqiang will start the annual dialogue mechanism with Prime Minister Trudeau of Canada and hold the first annual dialogue between the two premiers.	Li Keqiang will initiate the annual dialogue mechanism between premiers of China and Canada during this visit, and hold the first annual dialogue with Premier Trudeau of Canada.

Google Neural Machine Translation
(Google Research’s blog)

Text Summarization



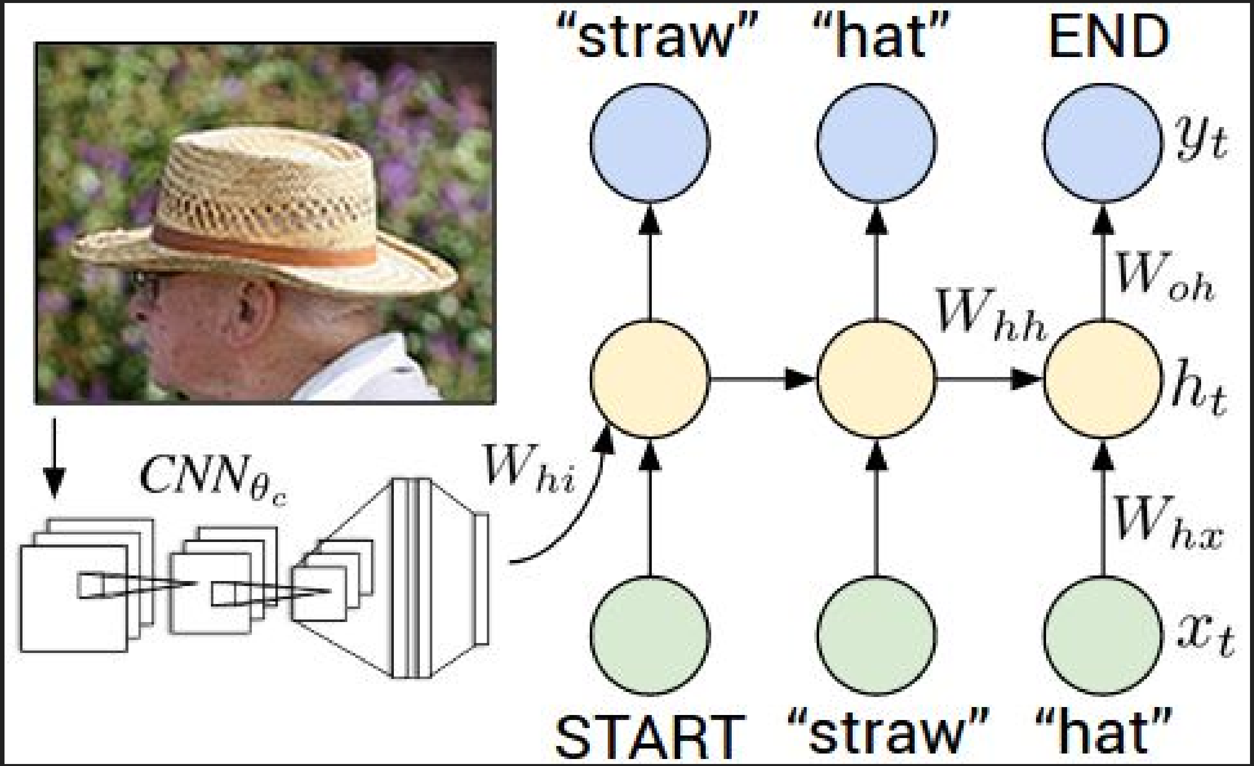
Nallapati, Ramesh, et al. "Abstractive text summarization using sequence-to-sequence rnns and beyond." arXiv preprint arXiv:1602.06023 (2016).

Text Summarization

Source Document
(@entity0) wanted : film director , must be eager to shoot footage of golden lassos and invisible jets . <eos> @entity0 confirms that @entity5 is leaving the upcoming " @entity9 " movie (the hollywood reporter first broke the story) . <eos> @entity5 was announced as director of the movie in november . <eos> @entity0 obtained a statement from @entity13 that says , " given creative differences , @entity13 and @entity5 have decided not to move forward with plans to develop and direct ’ @entity9 ’ together . <eos> " (@entity0 and @entity13 are both owned by @entity16 . <eos>) the movie , starring @entity18 in the title role of the @entity21 princess , is still set for release on june 00 , 0000 . <eos> it ’s the first theatrical movie centering around the most popular female superhero . <eos> @entity18 will appear beforehand in " @entity25 v. @entity26 : @entity27 , " due out march 00 , 0000 . <eos> in the meantime , @entity13 will need to find someone new for the director ’s chair . <eos>
Ground truth Summary
@entity5 is no longer set to direct the first " @entity9 " theatrical movie <eos> @entity5 left the project over " creative differences " <eos> movie is currently set for 0000
words-lvt2k
@entity0 confirms that @entity5 is leaving the upcoming " @entity9 " movie <eos> @entity13 and @entity5 have decided not to move forward with plans to develop <eos> @entity0 confirms that @entity5 is leaving the upcoming " @entity9 " movie

Nallapati, Ramesh, et al. "Abstractive text summarization using sequence-to-sequence rnns and beyond." arXiv preprint arXiv:1602.06023 (2016).

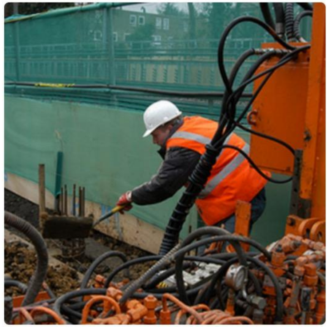
Image Captioning



Karpathy, Andrej, and Li Fei-Fei. "Deep visual-semantic alignments for generating image descriptions." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

Image Captioning

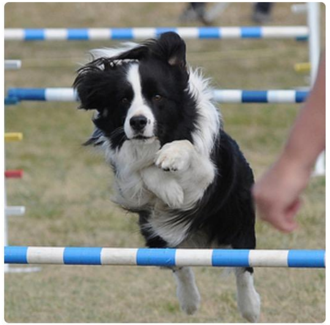
"man in black shirt is playing guitar."



"construction worker in orange safety vest is working on road."

"two young girls are playing with lego toy."

"girl in pink dress is jumping in air."

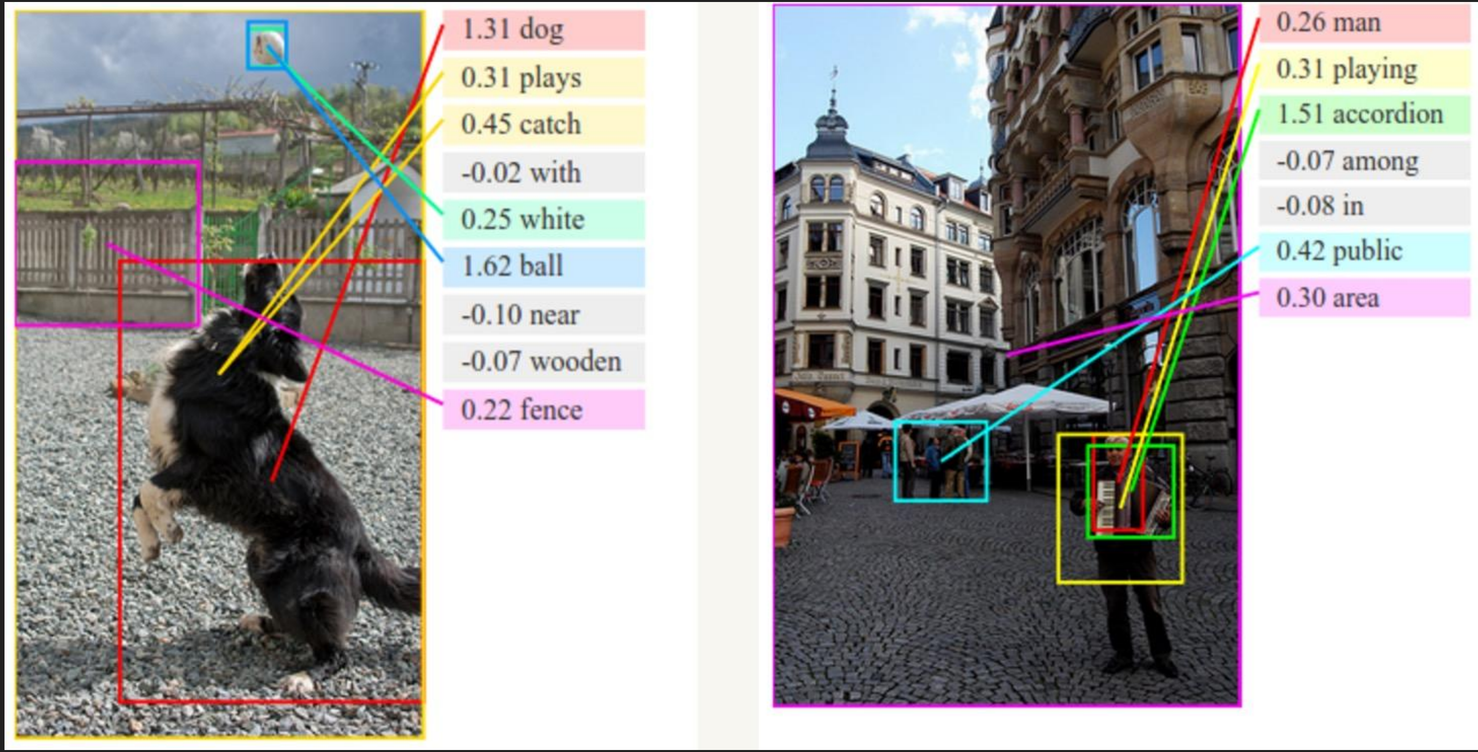


"black and white dog jumps over bar."

"young girl in pink shirt is swinging on swing."

Karpathy, Andrej, and Li Fei-Fei. "Deep visual-semantic alignments for generating image descriptions." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

Image Captioning



Karpathy, Andrej, and Li Fei-Fei. "Deep visual-semantic alignments for generating image descriptions." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

RNNs in TensorFlow

Cell Support (tf.nn.rnn_cell)

- BasicRNNCell: The most basic RNN cell.
- RNNCell: Abstract object representing an RNN cell.
- BasicLSTMCell: Basic LSTM recurrent network cell.
- LSTMCell: LSTM recurrent network cell.
- GRUCell: Gated Recurrent Unit cell

Construct Cells (tf.nn.rnn_cell)

```
cell = tf.nn.rnn_cell.GRUCell(hidden_size)
```

Stack multiple cells

```
cell = tf.nn.rnn_cell.GRUCell(hidden_size)

rnn_cells = tf.nn.rnn_cell.MultiRNNCell([cell] * num_layers)
```

Construct Recurrent Neural Network

- `tf.nn.dynamic_rnn`: uses a tf.While loop to dynamically construct the graph when it is executed. Graph creation is faster and you can feed batches of variable size.
- `tf.nn.bidirectional_dynamic_rnn`: dynamic_rnn with bidirectional

Stack multiple cells

```
cell = tf.nn.rnn_cell.GRUCell(hidden_size)

rnn_cells = tf.nn.rnn_cell.MultiRNNCell([cell] * num_layers)

output, out_state = tf.nn.dynamic_rnn(cell, seq, length, initial_state)
```

Any problem with this?

Stack multiple cells

```
cell = tf.nn.rnn_cell.GRUCell(hidden_size)

rnn_cells = tf.nn.rnn_cell.MultiRNNCell([cell] * num_layers)

output, out_state = tf.nn.dynamic_rnn(cell, seq, length, initial_state)
```

Most sequences are not of the same length

Dealing with variable sequence length

Pad all sequences with zero vectors and all labels with zero label (to make them of the same length)

Most current models can’t deal with sequences of length larger than 120 tokens, so there is usually a fixed max_length and we truncate the sequences to that max_length

Dealing with variable sequence length

Pad all sequences with zero vectors and all labels with zero label (to make them of the same length)

Most current models can’t deal with sequences of length larger than 120 tokens, so there is usually a fixed max_length and we truncate the sequences to that max_length

Problem?

Padded/truncated sequence length

The padded labels change the total loss, which affects the gradients

Padded/truncated sequence length

Approach 1:

- Maintain a mask (True for real, False for padded tokens)
- Run your model on both the real/padded tokens (model will predict labels for the padded tokens as well)
- Only take into account the loss caused by the real elements

```
full_loss = tf.nn.softmax_cross_entropy_with_logits(preds, labels)
loss = tf.reduce_mean(tf.boolean_mask(full_loss, mask))
```

Padded/truncated sequence length

Approach 2:

- Let your model know the real sequence length so it only predict the labels for the real tokens

```
cell = tf.nn.rnn_cell.GRUCell(hidden_size)
rnn_cells = tf.nn.rnn_cell.MultiRNNCell([cell] * num_layers)
tf.reduce_sum(tf.reduce_max(tf.sign(seq), 2), 1)
output, out_state = tf.nn.dynamic_rnn(cell, seq, length, initial_state)
```

How to deal with common problems when training RNNS

Vanishing Gradients

Use different activation units:

- `tf.nn.relu`
- `tf.nn.relu6`
- `tf.nn.crelu`
- `tf.nn.elu`

In addition to:

- `tf.nn.softplus`
- `tf.nn.softsign`
- `tf.nn.bias_add`
- `tf.sigmoid`
- `tf.tanh`

Exploding Gradients

Clip gradients with tf.clip_by_global_norm

```
gradients = tf.gradients(cost, tf.trainable_variables())

clipped_gradients, _ = tf.clip_by_global_norm(gradients, max_grad_norm)

optimizer = tf.train.AdamOptimizer(learning_rate)
train_op = optimizer.apply_gradients(zip(gradients, trainables))
```

Exploding Gradients

Clip gradients with tf.clip_by_global_norm

```
gradients = tf.gradients(cost, tf.trainable_variables())
# take gradients of cosst w.r.t. all trainable variables

clipped_gradients, _ = tf.clip_by_global_norm(gradients, max_grad_norm)


optimizer = tf.train.AdamOptimizer(learning_rate)
train_op = optimizer.apply_gradients(zip(gradients, trainables))
```

Exploding Gradients

Clip gradients with tf.clip_by_global_norm

```
gradients = tf.gradients(cost, tf.trainable_variables())
# take gradients of cosst w.r.t. all trainable variables

clipped_gradients, _ = tf.clip_by_global_norm(gradients, max_grad_norm)
# clip the gradients by a pre-defined max norm

optimizer = tf.train.AdamOptimizer(learning_rate)
train_op = optimizer.apply_gradients(zip(gradients, trainables))
```

Exploding Gradients

Clip gradients with tf.clip_by_global_norm

```
gradients = tf.gradients(cost, tf.trainable_variables())
# take gradients of cosst w.r.t. all trainable variables

clipped_gradients, _ = tf.clip_by_global_norm(gradients, max_grad_norm)
# clip the gradients by a pre-defined max norm

optimizer = tf.train.AdamOptimizer(learning_rate)
train_op = optimizer.apply_gradients(zip(gradients, trainables))
# add the clipped gradients to the optimizer
```

Anneal the learning rate

Optimizers accept both scalars and tensors as learning rate

```
learning_rate = tf.train.exponential_decay(init_lr,
                                           global_step,
                                           decay_steps,
                                           decay_rate,
                                           staircase=True)
optimizer = tf.train.AdamOptimizer(learning_rate)
```

Overfitting

Use dropout through tf.nn.dropout or DropoutWrapper for cells

- `tf.nn.dropout`

```
hidden_layer = tf.nn.dropout(hidden_layer, keep_prob)
```

- `DropoutWrapper`

```
cell = tf.nn.rnn_cell.GRUCell(hidden_size)
cell = tf.nn.rnn_cell.DropoutWrapper(cell,
                                     output_keep_prob=keep_prob)
```

Language Modeling

Neural Language Modeling

- Allows us to measure how likely a sentence is
- Important input for Machine Translation (since high-probability sentences are typically correct)
- Can generate new text

Language Modeling: Main approaches

- Word-level: n-grams
- Character-level
- Subword-level: somewhere in between the two above

Language Modeling: N-grams

- The traditional approach up until very recently
- Train a model to predict the next word based on previous n-grams

What can be the problems?

Language Modeling: N-grams

- The traditional approach up until very recently
- Train a model to predict the next word based on previous n-grams
- Huge vocabulary
- Can’t generalize to OOV (out of vocabulary)
- Requires a lot of memory

Language Modeling: Character-level

- Introduced in the early 2010s
- Both input and output are characters

Pros and cons?

Language Modeling: Character-level

- Introduced in the early 2010s
- Both input and output are characters

Pros:

- Very small vocabulary
- Doesn't require word embeddings
- Faster to train

Cons:

- Low fluency (many words can be gibberish)

Language Modeling: Hybrid

- Word-level by default, switching to character-level for unknown tokens

Language Modeling: Subword-Level

- Input and output are subwords
- Keep W most frequent words
- Keep S most frequent syllables
- Split the rest into characters
- Seem to perform better than both word-level and character-level models*

```
new company dreamworks interactive
new company dre+ am+ wo+ rks: in+ te+ ra+ cti+ ve:
```

Mikolov, Tomáš, et al. "Subword language modeling with neural networks." preprint (http://www. fit. vutbr. cz/imikolov/rnnlm/char. pdf) (2012).

Demo:
Character-level Language Modeling

Generate fake Arvix abstracts

- Dataset: 7200 abstracts of Arvix papers about neural networks

“Heuristic optimisers which search for an optimal configuration of variables relative to an objective function often get stuck in local optima where the algorithm is unable to find further improvement. The standard approach to circumvent this problem involves periodically restarting the algorithm from random initial configurations when no further improvement can be found. We propose a method of partial reinitialization, whereby, in an attempt to find a better solution, only sub-sets of variables are re-initialised rather than the whole configuration. Much of the information gained from previous runs is hence retained. This leads to significant improvements in the quality of the solution found in a given time for a variety of optimisation problems in machine learning.”

Generate fake Arvix abstracts

- Evaluation: no scientific way to evaluate

"Deep learning neural network architectures can be used to best developing a new architectures contros of the training and max model parametrinal Networks (RNNs) outperform deep learning algorithm is easy to out unclears and can be used to train samples on the state-of-the-art RNN more effective Lorred can be used to best developing a new architectures contros of the training and max model and state-of-the-art deep learning algorithms to a similar pooling relevants. The space of a parameter to optimized hierarchy the state-of-the-art deep learning algorithms to a simple analytical pooling relevants. The space of algorithm is easy to outions of the network are allowed at training and many dectional representations are allow develop a groppose a network by a simple model interact that training algorithms to be the activities to maximul setting, ..."

GitHub

data/arvix_abstracts.txt

examples/11_char_nn_gist.py

Next class

Guest lecture by Lukasz Kaiser

Feedback: huyenn@stanford.edu

Thanks!