Today

Recurrent Neural Networks

Intro

Architectures

Backprop Through Time (BPTT)

LSTMs

Visual understanding applications for CNN+RNNs
Recurrent Neural Networks

So far:
- Input: Single image
- Output: Single prediction
- Even AEs and GANs follow the same paradigm: Feed Forward

What we want to do:
- Handle input that’s a (time) sequence
- Produce an output sequence
- From sequence to sequence

How??
Recurrent Neural Networks

Stack the neurons on top of each other.

Neurons have 2 outputs:
- Internal state (h) transfer to next neuron
- Prediction (y), depends on state (h)

Within the neuron the activation function becomes more complex.
Recurrent Neural Networks

Unrolled RNN

\[
\begin{align*}
&f(Wx+b) \quad f(Wx+b) \quad f(Wx+b) \quad f(Wx+b) \quad f(Wx+b) \quad f(Wx+b) \\
&y \\
&x \quad x \quad x \quad x \quad x \quad x \\
&t = 0 \quad t = 1 \quad t = 2 \quad t = 3 \quad t = 5 \quad t = 6
\end{align*}
\]
Recurrent Neural Networks

$t = 0$

Single input, sequence output
E.g. Image captioning (sentence)
Recurrent Neural Networks

Sequence input, single output
E.g. *Action classification, sentiment analysis*

\[
\begin{align*}
x_{t=0} ightarrow f(Wx+b) ightarrow x_{t=1} ightarrow f(Wx+b) ightarrow x_{t=2} ightarrow f(Wx+b) ightarrow x_{t=3} ightarrow f(Wx+b) ightarrow x_{t=5} ightarrow f(Wx+b) ightarrow x_{t=6} ightarrow f(Wx+b) ightarrow y
\end{align*}
\]
Recurrent Neural Networks

Sequence to sequence
E.g. Machine Translation, Video Captioning
Recurrent Neural Networks

Sequences when there are no sequences.

Random walk through an image.

[Karpathy]
Recurrent Neural Networks

Simple Recurrent Cell.

State update formula:

\[ h_i = f(W, x, h_{i-1}) \]

E.g.

\[ h_i = \tanh(W_h h_{i-1} + W_x x_i) \]

\[ y_i = W_y h_i \]

W is *shared* throughout the network.
Recurrent Neural Networks

Compute graph

\[ f(W,x,h) \]

\[ h_{i-1} \rightarrow f(W,x,h) \rightarrow h_i \rightarrow f(W,x,h) \rightarrow h_{i+1} \]

\[ L_{i-1} \rightarrow y \rightarrow L_i \rightarrow y \rightarrow L_{i+1} \]

\[ L \]

\[ x \]
Recurrent Neural Networks

Sequence to single:
“Encode” the sequence to a single vector
Recurrent Neural Networks

Single to sequence:
“Decode” a vector to a sequence
Recurrent Neural Networks

Sequence-to-sequence

Recurrent Encoder-Decoder
Recurrent Neural Networks

Training:

Inputs and outputs known, can compute loss and backprop.

\[ f(W_h, W_x, h, x) = \text{tanh}(...) \]

[Karpathy]

What to do in test time?
Recurrent Neural Networks

Test time:

Feed one character at a time.

Sample the output distribution (softmax).

[Karpathy]
Recurrent Neural Networks

Test time:

Feed one character at a time.

Sample the output distribution (softmax).

[Karpathy]
Recurrent Neural Networks

Test time:
Feed one character at a time.

Sample the output distribution (softmax).

[Karpathy]
Backpropagation

Feed Forward

Back Propagate Through Time (BPTT)
Backpropagation

Truncated BPTT
Backpropagation

Truncated BPTT

Feed Forward

BPTT
Examples

"Tmont thithey" fomesscerliund
Keushey. Thom here
sheulke, anmerenith ol sivh I lalterthend Bleipile shuwy fil on aseterlome
cooniogenncc Phe lism thond hon at. MeiDimorotin in ther thize."

we counter. He stutn co des. His stanted out one ofler that concossions and was
to gearang reay Jotrets and with fre colt oft paitt thin wall. Which das stimn

Aftair fall unsuch that the hall for Prince Velzonski's that me of
her hearly, and behs to so arwage fiving were to it beloge, pavu say falling misfort
how, and Gogition is so overelical and ofter.
"Why do what that day," replied Natasha, and wishing to himself the fact the princess, Princess Mary was easier, fed in had oftened him. Pierre aking his soul came to the packs and drove up his father-in-law women.
Examples

C code...

```c
#include <asm/io.h>
#include <asm/prom.h>
#include <asm/e820.h>
#include <asm/system_info.h>
#include <asm/setew.h>
#include <asm/pgproto.h>

#define REG_PG  vesa_slot_addr_pack
#define PFM_NOCOMP  AFSR(0, load)
#define STACK_DDR(type)  (func)

#define SWAP_ALLOCATE(nr)  (e)
#define emulate_sign() arch_get_unaligned_child()
#define access_rw(tst)  asm volatile("movd %%esp, %0, %3" : "r" (0));

if (__type & DO_READ)

static void stat_pc_SEC crud_read_mostly offsetof(struct seq_argsqueue, \
    pC[1]);

static void

os_prefix(unsigned long sys)
{
    #ifdef CONFIG_PREEMPT
        PUT_PARAM_RAID(2, sel) = get_state_state();
        set_pid_sum((unsigned long)state, current_state_str(),
            (unsigned long)-1->lr_full; low;
    }
```
Examples

For $\bigoplus_{i=1}^{n} \mathcal{L}_{n_i}$, where $\mathcal{L}_{n_i} = 0$, 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

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

and the compare in the fibre product covering we have to prove the lemma generated by $\prod_{i} Z \times_X U \to V$. Consider the maps $M$ along the set of points $\text{Sch}(p)$ 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 $\text{Sh}(G)$ such that $\text{Spec}(R') \to S$ is smooth or an

$$U = \bigcup U_i \times_X S_i,$$

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

To prove study we see that $\mathcal{F}[j]$ is a covering of $\mathcal{X}$, and $T_i$ is an object of $\mathcal{F}[j]$. Let $F_i \subset \mathcal{F}$ be a pre-sheaf of $\mathcal{O}_X$-modules on $C$ as a $\mathcal{F}$-module. In particular $\mathcal{F} = U/\mathcal{F}$ we have to show that $\mathcal{F}[j] = \Gamma(T_i, \mathcal{O}_{S_i} - \mathcal{I}_i^{j+1} \mathcal{F})$ is a unique morphism of algebraic stacks. Note that

$$\mathcal{F}[j] = \Gamma(S, \mathcal{O}_{S_i}) \to (U, \text{Spec}(A))$$

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 schemes. □

The result for prove any open covering follows from the less of Example ???. It may replace $S$ by $X_{\text{open, etale}}$. Which gives an open subspace of $X$ as equal to $S_{\text{open}}$, see Descent, Lemma ???. Namely, by Lemma ?? we see that $R$ is geometrically regular over $S$.

Lemma 0.1. Assume (3) and (9) 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

$$\mathcal{F}(A) = \Gamma(X, \mathcal{O}_X).$$

When in this case of to show that $\mathcal{Q} \to \mathcal{C}_X$ is stable under the following result in the second conditions of (1), and (9). This finishes the proof. By Definition ??? (without element is when the closed subschemes are catenary. If $T$ is separative 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 suffice to check that 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 $f : X \to U$. Let $U \cap U_i = \bigcup_{i=1}^{n} U_i$ be the scheme $X$ over $S$ at the schemes $X_i \to X$ and $U = \lim X_i$. □

The following lemma surjective destrocomposes of this implies that $\mathcal{F}_x = \mathcal{F}_x = \mathcal{F}_{X_i, x}$. □

Lemma 0.2. Let $X$ be a locally Noetherian scheme over $S$, $E_0 = \mathcal{F}[j]$. Set $\mathcal{I} = \mathcal{I}_1 \subset \mathcal{I}_2$. Since $D = \mathcal{D} \subset \mathcal{E}$ are nonzero over $i_q \leq p$ is a subset of $\mathcal{D}_q \cap \mathcal{C}_2$ works.

Lemma 0.3. In Situation ???. Hence we may assume $q = 0$. □

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

$$D(\mathcal{O}_X) = \mathcal{O}_X(D)$$

where $K$ is an $F$-algebra where $\delta_{p+1}$ is a scheme over $S$. □
Questions?
Recurrent Neural Networks

Problem: Long sequences, big gaps

[Colah]
Recurrent Neural Networks

Problem: Long sequences, big gaps

What’s the underlying problem?

[Colah]
Gradient Flow

Vanilla RNN cell
Gradient Flow
Vanishing / Exploding Gradients

Repeated factors of $W$ and $\tanh$.

Exploding gradient: We can “clip”
Vanishing gradient: **What to do??**

Hint: What can “survive” the derivative?

$$h_t = f(x_t, h_{t-1})$$

$$\frac{\partial z}{\partial h_{t-1}} = \frac{\partial f(x_t, h_{t-1})}{\partial h_{t-1}} \frac{\partial z}{\partial h_t}$$
Long Short Term Memory Cells

LSTM RNN cell

Addition survives the derivative better.

[Hochreiter et al., 1997]
[Colah]
Long Short Term Memory Cells

Gradient “highway” to prevent them from dying.

Cell state $C$ has no multiplication by $W$!

But a series of “gates” that control what stays and what disappears (forgotten) from $C$. 

[Hochreiter et al., 1997]
LSTMs Step-by-Step

What information to “forget” from the state?

“Forget gate”

\[
ft = \sigma (W_f \cdot [h_{t-1}, x_t] + b_f)
\]
LSTMs Step-by-Step

What information to add to the state?

"Input gate"

\[ i_t = \sigma (W_i \cdot [h_{t-1}, x_t] + b_i) \]
\[ \tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \]
LSTMs Step-by-Step

Update the state.

\[ C_t = f_t \times C_{t-1} + i_t \times \tilde{C}_t \]
LSTMs Step-by-Step

Output, with some selection over the raw state.

“Future planning”: predict what comes next

\[ o_t = \sigma \left( W_o \left[ h_{t-1}, x_t \right] + b_o \right) \]

\[ h_t = o_t \ast \tanh \left( C_t \right) \]
Gated Recurrent Unit (GRU)

Merge states (c,h), merge forget and input gate.

Simpler and faster.

\[
\begin{align*}
    z_t &= \sigma (W_z \cdot [h_{t-1}, x_t]) \\
    r_t &= \sigma (W_r \cdot [h_{t-1}, x_t]) \\
    \tilde{h}_t &= \tanh (W \cdot [r_t \ast h_{t-1}, x_t]) \\
    h_t &= (1 - z_t) \ast h_{t-1} + z_t \ast \tilde{h}_t
\end{align*}
\]
Questions?
What do the Neurons Like?

A neuron that likes URLs.

---

Green-Blue mapping: value of the LSTM h state after tanh with C.

5 Red lines: predictions of next character.

[Karpathy]
What do the Neurons Like?

A neuron that likes markdown links.

Green-Blue mapping: value of the LSTM $h$ state after $\tanh$ with $C$.

5 Red lines: predictions of next character.

[Karpathy]
A neuron that encodes **position** in the link.

**Green-Blue**

mapping: value of the LSTM h state after tanh with C.

5 **Red** lines: predictions of next character.

[Karpathy]
What do the Neurons Like?

A neuron that encodes **position** in the line, sentence or if-statement.

The bottom one: No easy interpretation.

**Red-Blue** mapping: value of the LSTM h state after tanh with C.

---

Cell sensitive to position in line:

*The state importance of the crossing of the merazine 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 were destroyed, people cried 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 seem 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 outraged by the same desire.

Kutuzov, shrugging his shoulders, replied with his subtle penetrating eyes. I meant merely to say what I said.*

Cell that robustly activates inside if-statements:

```
static int dequeue_signalstruct suspending (pending, mask);
int sig = next_signal(pending, mask);
if (current->notifier)
  sigmember(current->notifier_mask, sig);
if (!current->notifier_data) (clear_thread_flag(if_stopnoticing);
  return 0;
);)
```

```
select_signal(sig, pending, info);
return sig;
```

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

```
// pack a filter string representing from user-space buffer
char *str = malloc┏string length + 1); char *data, *len)
char *temp;
if (len == NUL) len = strlen(temp);
return ERH(x,...)
```

...defines the longest valid length.

---

[Karpathy]
Questions?
Image Captioning

a cat laying on top of a brown couch.
the cat lies on a leather couch next to a closed laptop and a dictionary.
the kitty is catnapping on the small couch.
a tiger cat on a love seat with a laptop and a dictionary.
a cat laying on a couch between a book and a laptop.

a black cat has its head in a white toilet
a black cat with it's head inside of a toilet bowl.
a black cat balances on the rim of a toilet seat.
a cat is on the toilet and drinking from it.
a cat in the bathroom drinking toilet water.
Image Captioning

Idea use pretrained image features to bootstrap language generation.

**Convolutional Net**
Visual Feature Extraction

**Recurrent Net**
Language “Decoder”
Add visual features to output activation:

\[ b_v = W_{hi}[\text{CNN}_{\theta_c}(I)] \]  
\[ h_t = f(W_{hx}x_t + W_{hh}h_{t-1} + b_h + \mathbb{1}(t = 1) \odot b_v) \]  

In the equations above, \( W_{hi}, W_{hx}, W_{hh}, W_{oh}, x_t \) and \( b_h, b_v \) are learnable parameters, and \( \text{CNN}_{\theta_c}(I) \) is the last layer of a CNN.

[Karpathy & Li]
RNN cell outputs a distribution over the space of words. Sample for the next word:

\[ y_t = \text{softmax}(W_oh_t + b_o). \]
Image Captioning

[Image Captioning Diagram]

[References: Karpathy & Li]
And so on until <$END$> is generated.

How to train? What’s the RNN flavor? See the paper… :)

[Karpathy & Li]
Results

A cat sitting on a suitcase on the floor
A cat is sitting on a tree branch
A dog is running in the grass with a frisbee
A white teddy bear sitting in the grass
Two people walking on the beach with surfboards
A tennis player in action on the court
Two giraffes standing in a grassy field
A man riding a dirt bike on a dirt track
Results

A woman is holding a cat in her hand

A woman standing on a beach holding a surfboard

A bird is perched on a tree branch

A man in a baseball uniform throwing a ball

A person holding a computer mouse on a desk
Captioning With Attention

1. Input Image
2. Convolutional Feature Extraction
3. RNN with attention over the image
4. Word by word generation

[Image of a bird flying over a body of water]

Xu et al.
Captioning With Attention

Figure 4. Examples of attending to the correct object (*white* indicates the attended regions, *underlines* indicated the corresponding word)

- A woman is throwing a *frisbee* in a park.
- A *dog* is standing on a hardwood floor.
- A *stop* sign is on a road with a mountain in the background.
- A little *girl* sitting on a bed with a teddy bear.
- A group of *people* sitting on a boat in the water.
- A giraffe standing in a forest with trees in the background.

[Xu et al.]
Visual Question Answering

What is the mustache made of?

AI System

bananas

[VQA challenge]
VQA with Attention

LSTM: $v_t$ and $C$ are VGG features

\[
\begin{align*}
\mathbf{i}_t &= \sigma(W_{vi}v_t + W_{hi}h_{t-1} + W_{ri}r_t + b_i) \\
\mathbf{f}_t &= \sigma(W_{vf}v_t + W_{hf}h_{t-1} + W_{rf}r_t + b_f) \\
\mathbf{o}_t &= \sigma(W_{vo}v_t + W_{ho}h_{t-1} + W_{ro}r_t + b_o) \\
\mathbf{g}_t &= \phi(W_{vg}v_t + W_{hg}h_{t-1} + W_{rg}r_t + b_g) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \\
\mathbf{h}_t &= \mathbf{o}_t \odot \phi(\mathbf{c}_t) \\
\mathbf{e}_t &= w_a^T \tanh(W_{he}h_{t-1} + W_{ce}C(I)) + b_a \\
\mathbf{a}_t &= \text{softmax}(\mathbf{e}_t) \\
\mathbf{r}_t &= \mathbf{a}_t^T C(I)
\end{align*}
\]

Q: Which is the brown bread?

A1:  A2:  A3:  A4:
Results

Far from human level...

<table>
<thead>
<tr>
<th>Method</th>
<th>What</th>
<th>Where</th>
<th>When</th>
<th>Who</th>
<th>Why</th>
<th>How</th>
<th>Which</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human (Question)</td>
<td>0.356</td>
<td>0.322</td>
<td>0.393</td>
<td>0.342</td>
<td>0.439</td>
<td>0.337</td>
<td>-</td>
<td>0.353</td>
</tr>
<tr>
<td>Human (Question + Image)</td>
<td>0.965</td>
<td>0.957</td>
<td>0.944</td>
<td>0.965</td>
<td>0.927</td>
<td>0.942</td>
<td>0.973</td>
<td>0.966</td>
</tr>
<tr>
<td>Logistic Regression (Question)</td>
<td>0.420</td>
<td>0.375</td>
<td>0.666</td>
<td>0.510</td>
<td>0.354</td>
<td>0.458</td>
<td>0.354</td>
<td>0.383</td>
</tr>
<tr>
<td>Logistic Regression (Image)</td>
<td>0.408</td>
<td>0.426</td>
<td>0.438</td>
<td>0.415</td>
<td>0.337</td>
<td>0.303</td>
<td>0.256</td>
<td>0.305</td>
</tr>
<tr>
<td>Logistic Regression (Question + Image)</td>
<td>0.429</td>
<td>0.454</td>
<td>0.621</td>
<td>0.501</td>
<td>0.343</td>
<td>0.356</td>
<td>0.307</td>
<td>0.352</td>
</tr>
<tr>
<td>LSTM (Question)</td>
<td>0.430</td>
<td>0.414</td>
<td>0.693</td>
<td>0.538</td>
<td>0.491</td>
<td>0.484</td>
<td>-</td>
<td>0.462</td>
</tr>
<tr>
<td>LSTM (Image)</td>
<td>0.422</td>
<td>0.497</td>
<td>0.660</td>
<td>0.523</td>
<td>0.475</td>
<td>0.468</td>
<td>0.299</td>
<td>0.359</td>
</tr>
<tr>
<td>LSTM (Question + Image) [28]</td>
<td>0.489</td>
<td>0.544</td>
<td>0.713</td>
<td>0.581</td>
<td>0.513</td>
<td>0.503</td>
<td>0.521</td>
<td>0.521</td>
</tr>
<tr>
<td>Ours, LSTM-Att (Question + Image)</td>
<td>0.515</td>
<td>0.570</td>
<td>0.750</td>
<td>0.595</td>
<td>0.555</td>
<td>0.498</td>
<td>0.561</td>
<td>0.556</td>
</tr>
</tbody>
</table>
Wrap-Up

Today we saw:

RNNs: Intro, architectures, backprop

LSTMs

Visual understanding applications