Topics:
- Recurrent Neural Networks
- Long Short-Term Memory

CS 4644-DL / 7643-A
ZSOLT KIRA
• **Assignment 3**
  • Due **March 14th 11:59pm EST.**
  • See [https://piazza.com/class/ky0k0ha5vgy1mk?cid=176](https://piazza.com/class/ky0k0ha5vgy1mk?cid=176)
    • (note: ignore logistics on that slide deck)
  • Do not use grace period as extension! Submit *something* on time and continue to refine it.

• **Projects**
  • Project proposal due **March 13th** (into grace period)
The Space of Architectures

- Recurrent Neural Networks
- Fully Connected Neural Networks
- Convolutional Neural Networks
- Attention-Based Networks
- Graph-Based Networks

Input Data → Predictions
Input Image → Predictions

The Space of Architectures

Georgia Tech
(Vanilla) Recurrent Neural Network

The state consists of a single "hidden" vector $h$:

$$y_t = W_{hy} h_t + b_y$$

$$h_t = \tanh(W_{hh} h_{t-1} + W_{hx} x_t)$$

$$= \tanh \left( (W_{hh} \ W_{hx}) \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} \right)$$

$$= \tanh \left( W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} \right)$$

Sometimes called a "Vanilla RNN" or an "Elman RNN" after Prof. Jeffrey Elman

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Recurrent Neural Network

We can process a sequence of vectors $x$ by applying a recurrence formula at every time step:

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

- $h_t$: new state
- $h_{t-1}$: old state
- $x_t$: input vector at some time step
- $f_W$: some function with parameters $W$
RNN: Computational Graph: Many to Many

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Sequence to Sequence: Many-to-one + one-to-many

**Many to one:** Encode input sequence in a single vector

**One to many:** Produce output sequence from single input vector

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Example:
Character-level Language Model

Vocabulary: [h,e,l,o]

Example training sequence: “hello”
Example:
Character-level Language Model

Vocabulary:
[h,e,l,o]

Example training sequence:
“hello”

\[ h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h) \]
Distributed Representations Toy Example

• Can we interpret each dimension?
Power of distributed representations!

Local

\[ \bullet \bullet \circ \bullet = VR + HR + HE = ? \]

Distributed

\[ \bullet \bullet \circ \bullet = V + H + E \approx \circ \]
Example:  
Character-level Language Model

Vocabulary:  
[h,e,l,o]

Example training sequence:  
“hello”
Example:
Character-level Language Model

Vocabulary:
[h,e,l,o]

Example training sequence:
“hello”
Test Time: Sample / Argmax / Beam Search

Example:
Character-level Language Model Sampling

Vocabulary:
[h,e,l,o]

At test-time sample characters one at a time, feed back to model
Test Time: Sample / Argmax / Beam Search

Example:
Character-level Language Model Sampling

Vocabulary:
[h,e,l,o]

At test-time sample characters one at a time, feed back to model
Test Time: Sample / Argmax / Beam Search

Example:
Character-level Language Model Sampling

Vocabulary:
[h,e,l,o]

At test-time sample characters one at a time, feed back to model
Example:
Character-level Language Model
Sampling

Vocabulary: [h,e,l,o]

At test-time sample characters one at a time, feed back to model
Let's do Monday.

Monday works for me.

Either day works for me.
Backpropagation through time

Forward through entire sequence to compute loss, then backward through entire sequence to compute gradient

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
**Truncated** Backpropagation through time

Run forward and backward through chunks of the sequence instead of whole sequence
Truncated Backpropagation through time

Carry hidden states forward in time forever, but only backpropagate for some smaller number of steps
Truncated Backpropagation through time

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
min-char-rnn.py gist: 112 lines of Python

[Link to gist: https://gist.github.com/karpathy/d4dee5668568291f086]
THE SONNETS

by William Shakespeare

From fairest creatures we desire increase,
That thereby beauty's rose might never die,
But as by sea and stock the enmity
Hic under heirs might bear his memory:
His tender heer might bear his name:
But thou, contracted to one hour's cause,
Fed in his womb with self-sustained fuel,
Making a famine where abundance lies,
Tyrannish dry how so dry now estore.
Thus art thou bred in this thy present.
And only herald to the gandy spring.
Within thine own bed hast thou to comfort;
And under child enskyd waste in ambiguous:
Pay the world, or else this glutton be.
To eat the world's due, by the grave and then.

When forty winters shall besiege thy brow,
And dig deep trenches in thy beauty's field,
Thy youth's proud livery so gaudied on,
Will be a mere/nob thirtyswell, worth held;
Thine own self, when thou hast rov'd all day:
To say, within thine own deep coolness eyes,
Wore an all-'eating shrew, and blinstein' praise:
How much more praise shouldest thou not be like:
It thou couldst answer This Fair child of mine
Shall un my count, and make my old excuse;
Proving his beauty by succession thine?
This were to bee new made when thou art old,
And see thy blood warm when thou feel'st it cold.
at first:

"Why do what that day," replied Natasha, and wishing to himself the fact the princess, Princess Mary was easier, fed in had oftend him. Pierre aking his soul came to the packs and drove up his father-in-law women.

"Tmont hithey" fomesscerliund
Keushey. Thom here
sheulke, anmerenith oI sivh I lalterthend Bleipile shuyv fil on aseterlome
canoiogennc Phe lism thond hon at. MeiDimorotion in ther thize."

train more

train more

train more
PANDARUS:
Alas, I think he shall be come approached and the day
When little strain 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 nes 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.

VIOLA:
Why, Salisbury must find his flesh and thought
That which I am not aps, not a man and in fire,
To show the reigning of the raven and the wars
To grace my hand reproach within, and not a fair are hand,
That Caesar and my goodly father’s world;
When I was heaven of presence and our fleets,
We spare with hours, but cut thy council I am great,
Murdered and by thy master’s ready there
My power to give thee but so much as hell:
Some service in the noble bondman here,
Would show him to her wine.

KING LEAR:
O, if you were a feeble sight, the courtesy of your law,
Your sight and several breath, will wear the gods
With his heads, and my hands are wonder’d at the deeds,
So drop upon your lordship’s head, and your opinion
Shall be against your honour.
The Stacks Project: open source algebraic geometry textbook

The Stacks Project

Browse chapters

<table>
<thead>
<tr>
<th>Part</th>
<th>Chapter</th>
<th>online</th>
<th>TeX source</th>
<th>view pdf</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preliminaries</td>
<td>1. Introduction</td>
<td>online</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2. Conventions</td>
<td>online</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. Set Theory</td>
<td>online</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4. Categories</td>
<td>online</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>5. Topology</td>
<td>online</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>6. Sheaves on Spaces</td>
<td>online</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>7. Sites and Sheaves</td>
<td>online</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8. Stacks</td>
<td>online</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>9. Fields</td>
<td>online</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10. Commutative Algebra</td>
<td>online</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Parts

1. Preliminaries
2. Schemes
3. Topics in Scheme Theory
4. Algebraic Spaces
5. Topics in Geometry
6. Deformation Theory
7. Algebraic Stacks
8. Miscellany

Statistics

The Stacks project now consists of
- 455910 lines of code
- 14221 tags (56 inactive tags)
- 2366 sections

Latex source

http://stacks.math.columbia.edu/
The stacks project is licensed under the GNU Free Documentation License

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Georgia Tech
For $\bigoplus_{i=1}^{\infty} L_{n_i}$, where $L_{n_i} = 0$, 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 comparably in the fibre product covering we have to prove the lemma generated by $\bigsqcup Z \times_U U \to V$. Consider the maps $M$ along the set of points of $\text{Sch}_{/\mathbb{S}}$, and $U \to U$ is the fibre category of $S$ in $U$ in Section 7 and the fact that any $U$ affine, see Morphisms, Lemma 7. Hence we obtain a scheme $S$ and any open subset $W \subset U$ in $\text{Sch}(G)$ such that $\text{Spec}(R) \to S$ 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 $\mathcal{O}_{X,s}$ is a scheme where $x, x', x'' \in S'$ such that $\mathcal{O}_{X,x'} \to \mathcal{O}_{X,x''}$ is separated. By Algebra, Lemma 7 we can define a map of complexes $\mathcal{O}_{S'}(x'/S')$ and we win. □

To prove study we see that $\mathcal{F}$ is a covering of $X'$, and $T_i$ is an object of $\mathcal{F}_{X/S}$ for $i > 0$ and $\mathcal{F}_n$ exists and let $F_i$ be a presheaf of $\mathcal{O}_X$-modules on $\mathcal{C}$ as a $\mathcal{F}$-module. In particular $\mathcal{F} = U/F$ we have to show that

$$\widetilde{\mathcal{F}} = \mathcal{F} \otimes_{\mathcal{O}_S} \mathcal{O}_{\mathcal{S}, \text{Spec}(S)} \otimes_{\mathcal{O}_S} \mathcal{F}$$

is a unique morphism of algebraic stacks. Note that

$$\text{Arrows} = \left( \text{Sch}/S \right)_{/\mathcal{F}} \otimes_{\text{Sch}/S} \left( \text{Sch}/S \right)_{/\mathcal{F}}$$

and

$$\mathcal{F} = \Gamma(S, \mathcal{O}) \to \left( U, \text{Spec}(A) \right)$$

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. □

The result for prove any open covering follows from the less of Example 7. It may replace $S$ by $X_{\text{spaces,Etale}}$ which gives an open subspace of $X$ and $T$ equal to $S_{\text{etale}}$, see Descent, Lemma 7. Namely, by Lemma 7 we see that $R$ is geometrically regular over $S$.

**Lemma 0.1.** Assume (3) and (3) by the construction in the description. Suppose $X = \lim X$ by the formal open covering $X$ and a single map $\text{Proj}_X(A) = \text{Spec}(B)$ over $U$ compatible with the complex $\text{Set}(A) = \Gamma(X, \mathcal{O}_{X,X'})$.

When in this case of to show that $\mathcal{Q} \to \mathcal{C}_{X/X}$ is stable under the following result in the second conditions of (1), and (3). This finishes the proof. By Definition 7 without element is when the closed subschemes are catenary. 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 form all sheaves of sheaves on $X$. But given a scheme $U$ and a surjective étale morphism $U \to X$. Let $U \to U = \bigsqcup_{i=1}^{\infty} U_i$ be the scheme $X$ over $S$ at the schemes $X_i \to X$ and $U = \lim X_i$.

The following lemma surjective restreicolors of this implies that $\mathcal{F}_n = \mathcal{F}_n = \mathcal{F}_{X,...,0}$.

**Lemma 0.2.** Let $X$ be a locally Noetherian scheme over $S$, $E = \mathcal{F}_{X/S}$. Set $T = T_! \subset T_0$. Since $T_0 \subset T_!$ are nonzero over $p \leq p$ is a subset of $T_0 = \mathcal{F}_{X,0}$ works.

**Lemma 0.3.** In Situation 7. Hence we may assume $p' = 0$.

**Proof.** We will use the property we see that $p$ is the next functor (7). On the other hand, by Lemma 7 we see that

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

where $K$ is an $F$-algebra where $\delta_{+1}$ is a scheme over $S$. □
Proof. Omitted.

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

$$O_{O_X} = O_X(L)$$

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

$$O_X(F) = \{\text{morph}_1 \times O_X(\mathcal{G}, F)\}$$

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

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

Proof. See Spaces, Lemma ??.

Lemma 0.3. Let $S$ be a scheme. Let $X$ be a scheme and $X$ is an affine open covering. Let $U \subseteq X$ 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

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

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 $F$ be a quasi-coherent sheaf of $O_X$-modules. The following are equivalent

1. $F$ is an algebraic space over $S$.
2. $F$ is an affine open covering.

Consider a common structure on $X$ and $X$ the functor $O_X(U)$ which is locally of finite type.

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

\[ \begin{array}{ccc}
S & \to & \cdots \\
\downarrow & & \downarrow \\
\mathcal{G} & \to & \cdots \\
\downarrow & & \downarrow \\
\cdots & & \cdots \\
\end{array} \]

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

- the composition of $F$ is a regular sequence,
- $O_X$ is a sheaf of rings.

Proof. We have to see that $X = \text{Spec}(R)$ and $F$ is a finite type representation by algebraic space. The property $F$ 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 refined above we conclude that $U$ is an open covering of $C$. The functor $F$ is a "field"

$$O_{X,S} \to F$$

is an isomorphism of covering of $O_X$. If $F$ is the unique element of $F$ such that $X$ is an isomorphism.

The property $F$ is a disjoint union of Proposition ?? and we can filtered set of presentations of a scheme $O_X$-algebra with $F$ are open of finite type over $S$.

If $F$ is a scheme theoretic image points.

If $F$ is a finite direct sum $O_{X,S}$ is a closed immersion, see Lemma ???. This is a sequence of $F$ is a similar morphism.
```c
static void do_command(struct seq_file *m, void *v)
{
    int column = 32 << (cmd[2] & 0x80);
    if (state)
        cmd = (int)(int_state ^ (in_8(&ch->ch_flags) & Cmd) ? 2 : 1);
    else
        seq = 1;
    for (i = 0; i < 16; i++) {
        if (k & (i << 1))
            pipe = (in_use & UMXTHREAD_UNCC) +
            ((count & 0x00000000ffffff8) & 0x000000f) << 8;
        if (count == 0)
            sub(pid, ppc_md.kexec_handle, 0x20000000);
        pipe_set_bytes(i, 0);
    
    /* Free our user pages pointer to place camera if all dash */
    subsystem_info = &of_changes[PAGE_SIZE];
    rek_controls(offset, idx, &offset);
    /* Now we want to deliberately put it to device */
    control_check_polarity(&context, val, 0);
    for (i = 0; i < COUNTER; i++)
        seq_puts(s, "policy ");
}
```
```c
#include <asm/io.h>
#include <asm/prom.h>
#include <asm/e820.h>
#include <asm/system_info.h>
#include <asm/seteew.h>
#include <asm/pgproto.h>

#define REG_PG vesa_slot_addr_pack
#define PFM_NOMCOMP APSR(0, load)
#define STACK_DDR(type) (func)

#define SWAP_ALLOCATE(nr) (e)
#define emulate_sigs() arch_get_unaligned_child()
#define access_rwlock(TST) asm volatile("movd %esp, %0, %3" : : "r" (0));
    if (__type & DO_READ)

static void stat_PC_SEC __read_mostly offsetof(struct seq_argsqueue, \pC)(1));

static void
as_prefix(unsigned long sys)
{
    #ifdef CONFIG_PREEMPT
        PUT_PARAM_RAID(2, sel) = get_state_state();
        set_pid_num((unsigned long)state, current_state_str(),
                    (unsigned long)-1->lr_full, low;
    }
```
Searching for interpretable cells

Karpathy, Johnson, and Fei-Fei: Visualizing and Understanding Recurrent Networks, ICLR Workshop 2016
Searching for interpretable cells

/* 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. */
}

Karpathy, Johnson, and Fei-Fei: Visualizing and Understanding Recurrent Networks, ICLR Workshop 2016
Figures copyright Karpathy, Johnson, and Fei-Fei, 2015; reproduced with permission
Searching for interpretable cells

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

quote detection cell

Karpathy, Johnson, and Fei-Fei: Visualizing and Understanding Recurrent Networks, ICLR Workshop 2016

Figures copyright Karpathy, Johnson, and Fei-Fei, 2015; reproduced with permission

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Searching for interpretable cells

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.

Karpathy, Johnson, and Fei-Fei: Visualizing and Understanding Recurrent Networks, ICLR Workshop 2016
Figures copyright Karpathy, Johnson, and Fei-Fei, 2015; reproduced with permission

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Searching for interpretable cells

```c
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)) {
                clear_thread_flag(TIF_SIGPENDING);
                return 0;
            }
        }
        collect_signal(sig, pending, info);
    }
    return sig;
}
```

if statement cell

Karpathy, Johnson, and Fei-Fei: Visualizing and Understanding Recurrent Networks, ICLR Workshop 2016
Figures copyright Karpathy, Johnson, and Fei-Fei, 2015; reproduced with permission
Searching for interpretable cells
Searching for interpretable cells

code depth cell
Multilayer RNNs

\[ h_t^l = \tanh W^l \left( \frac{h_{t-1}^l}{h_{t-1}^{l-1}} \right) \]

- \( h \in \mathbb{R}^n \)
- \( W^l [n \times 2n] \)
Vanilla RNN Gradient Flow

\[ h_t = \tanh(W_{hh}h_{t-1} + W_{hx}x_t) \]
\[ = \tanh \left( (W_{hh} \quad W_{hx}) \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} \right) \]
\[ = \tanh \left( W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} \right) \]
Vanilla RNN Gradient Flow

Backpropagation from $h_t$ to $h_{t-1}$ multiplies by $W$ (actually $W_{hh}$)

$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t)$$
$$= \tanh \left( (W_{hh} \  W_{hx}) \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} \right)$$
$$= \tanh \left( W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} \right)$$

Bengio et al, “Learning long-term dependencies with gradient descent is difficult”, IEEE Transactions on Neural Networks, 1994

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Vanilla RNN Gradient Flow

Computing gradient of $h_0$ involves many factors of $W$ (and repeated tanh)

Bengio et al, "Learning long-term dependencies with gradient descent is difficult", IEEE Transactions on Neural Networks, 1994

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Vanilla RNN Gradient Flow

Computing gradient of $h_0$ involves many factors of $W$ (and repeated tanh)

Largest singular value $> 1$: **Exploding gradients**

Largest singular value $< 1$: **Vanishing gradients**

Bengio et al. “Learning long-term dependencies with gradient descent is difficult”, IEEE Transactions on Neural Networks, 1994

Vanilla RNN Gradient Flow

Computing gradient of $h_0$ involves many factors of $W$ (and repeated tanh)

Largest singular value > 1: **Exploding gradients**

Largest singular value < 1: **Vanishing gradients**

Gradient clipping: Scale gradient if its norm is too big

```
grad_norm = np.sum(grad * grad)
if grad_norm > threshold:
    grad *= (threshold / grad_norm)
```

Bengio et al, “Learning long-term dependencies with gradient descent is difficult”, IEEE Transactions on Neural Networks, 1994

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Vanilla RNN Gradient Flow

Computing gradient of $h_0$ involves many factors of $W$ (and repeated tanh)

Largest singular value $> 1$: **Exploding gradients**

Largest singular value $< 1$: **Vanishing gradients**

Change RNN architecture

Bengio et al, "Learning long-term dependencies with gradient descent is difficult", IEEE Transactions on Neural Networks, 1994
Long Short Term Memory (LSTM)

Vanilla RNN

$$h_t = \tanh \left( W \left( h_{t-1} \right) \right)$$

LSTM

$$\begin{bmatrix} i \\ f \\ o \\ g \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{bmatrix} W \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

Hochreiter and Schmidhuber, “Long Short Term Memory”, Neural Computation 1997
Meet LSTMs

Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)
LSTMs Intuition: Memory

- Cell State / Memory
LSTMs Intuition: Forget Gate

• Should we continue to remember this “bit” of information or not?

\[ f_t = \sigma (W_f \cdot [h_{t-1}, x_t] + b_f) \]
LSTMs Intuition: Input Gate

- Should we update this “bit” of information or not?
  - If so, with what?

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

- Forget that + memorize this

\[ C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \]
LSTMs Intuition: Output Gate

- Should we output this “bit” of information to “deeper” layers?

\[ o_t = \sigma (W_o [h_{t-1}, x_t] + b_o) \]
\[ h_t = o_t \times \tanh(C_t) \]
LSTMs Intuition: Additive Updates

Backpropagation from $c_t$ to $c_{t-1}$ only elementwise multiplication by $f$, no matrix multiply by $W$
LSTMs Intuition: Additive Updates

Uninterrupted gradient flow!
LSTMs Intuition: Additive Updates

Uninterrupted gradient flow!

Similar to ResNet!
LSTMs

• A pretty sophisticated cell

(Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/))
LSTM Variants: Gated Recurrent Units

• Changes:
  – No explicit memory; memory = hidden output
  – $Z = \text{memorize new and forget old}$

\[
\begin{align*}
  z_t &= \sigma (W_z \cdot [h_{t-1}, x_t]) \\
  r_t &= \sigma (W_r \cdot [h_{t-1}, x_t]) \\
  \tilde{h}_t &= \tanh (W \cdot [r_t \cdot h_{t-1}, x_t]) \\
  h_t &= (1 - z_t) \cdot h_{t-1} + z_t \cdot \tilde{h}_t
\end{align*}
\]
Other RNN Variants

[An Empirical Exploration of Recurrent Network Architectures, Jozefowicz et al., 2015]