CS 4803 / 7643: Deep Learning

Topics:
  – Recurrent Neural Networks (Cont.)

Zsolt Kira
Georgia Tech
Administrative

- HW3 out
  - Due date pushed back to March 15th to allow you more time

- Next lecture: FB lecture on attention/transformers
Sequences in Input or Output?

• It’s a spectrum…

- **Input:** No sequence
- **Output:** No sequence
  - Example: "standard" classification / regression problems

- **Input:** No sequence
- **Output:** Sequence
  - Example: Im2Caption

- **Input:** Sequence
- **Output:** No sequence
  - Example: sentence classification, multiple-choice question answering

- **Input:** Sequence
- **Output:** Sequence
  - Example: machine translation, video classification, video captioning, open-ended question answering

Image Credit: Andrej Karpathy

(C) Dhruv Batra
2 Key Ideas

• Parameter Sharing
  – in computation graphs = adding gradients

• “Unrolling”
  – in computation graphs with parameter sharing

• Parameter sharing + Unrolling
  – Allows modeling arbitrary sequence lengths!
  – Keeps numbers of parameters in check
Recurrent Neural Network

We can process a sequence of vectors $\mathbf{x}$ by applying a **recurrence formula** at every time step:

$$ h_t = f_W(h_{t-1}, \mathbf{x}_t) $$

- $h_t$ is the **new state**
- $h_{t-1}$ is the **old state**
- $\mathbf{x}_t$ is the **input vector at some time step**
- $f_W$ is **some function** with parameters $W$
Recurrent Neural Network

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

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

Notice: the same function and the same set of parameters are used at every time step.
(Vanilla) Recurrent Neural Network

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

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

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

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

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

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

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

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

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

Re-use the same weight matrix at every time-step

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
RNN: Computational Graph: Many to Many

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
RNN: Computational Graph: Many to Many

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
RNN: Computational Graph: Many to Many

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
RNN: Computational Graph: Many to One

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
RNN: Computational Graph: One 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
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”
Example:
Character-level Language Model

Vocabulary: [h,e,l,o]

Example training sequence: “hello”
Training Time: MLE / “Teacher Forcing”

Example:
Character-level Language Model

Vocabulary: [h,e,l,o]

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

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

Example:
Character-level Language Model Sampling

Vocabulary: [h,e,l,o]

At test-time sample characters one at a time, feed back to model

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

Vocabulary:
[h,e,l,o]

At test-time sample characters one at a time, feed back to model

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

Vocabulary: [h,e,l,o]

At test-time sample characters one at a time, feed back to model

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Lets do Monday.

Monday works for me.

Either day works for me.

Reply

Reply all

Forward
Backpropagation through time

Forward through entire sequence to compute loss, then backward through entire sequence to compute gradient
**Truncated** Backpropagation through time

Run forward and backward through chunks of the sequence instead of whole sequence

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

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

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Truncated Backpropagation through time
THE SONNETS
by William Shakespeare

From fairest creatures we desire increase,
That thereby beauty's rose might never die,
But as the riper should by time decease,
His tender heir might bear his memory:
But thou, contracted to thine own bright eyes,
Feed'st thy light's flame with self-substantial fuel,
Making a famine where abundance lies,
Thyself thy foe, to thy sweet self too cruel:
Thou art now the world's fresh ornament,
And only herald to the gaudy spring.
Within thine own bud hast thou the content,
And tender chari'lt make waste in rigouring:
Ply the world, or else this gluton be,
To eat the world's due, by the grave and thee.

When forty winters shall besiege thy brow,
And dig deep trenches in thy beauty's field,
Thy youth's sad livery so gazed on now,
Will be a tatter'd weed of small worth held;
Then being asked, where all thy beauty lies,
Where all the treasures of thy lusty days;
To say, within thine own deep sunken eyes,
Were an all-eating shame, and thriftless praise.
How much more praise deserv'd thy beauty's use,
If thou couldst answer 'This fair child of mine
Shall sum my count, and make my old excuse,'
Proving his beauty by succession thine!
This were to be new made when thou art old,
And see thy blood warm when thou feel'st it cold.
at first:

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

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.
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 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.

VIOLA:
Why, Salisbury must find his flesh and thought
That which I am not aps, not a man and in fire,
To show the reining 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

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

Proof. Proof of (1). It also start we get
\[
S = \text{Spec}(R) = U \times_U U \times_U U
\]
and the comparicly in the fibre product covering we have to prove the lemma generated by $\prod Z \times_U U \rightarrow V$. Consider the maps $M$ along the set of points $Schfpf$ and $U \rightarrow U$ 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 $\text{Spec}(R') \rightarrow S$ is smooth or an
\[
U = \bigcup U_i \times_{S_i} S_i
\]
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 $\mathcal{O}_{X,x'} \rightarrow \mathcal{O}_{X,x}$ is separated. By Algebra, Lemma ?? we can define a map of complexes $G_{S'}(x'/S')$ and we win. □

To prove study we see that $F_i$ is a covering of $X'$, and $T_i$ is an object of $F_{X/S}$ for $i > 0$ and $F_p$ exists and let $F_1$ be a presheaf of $\mathcal{O}_X$-modules on $C$ as a $F$-module. In particular $F = U/F$ we have to show that
\[
M^* = T^* \otimes_{\text{Spec}(k)} \mathcal{O}_{S,x} \rightarrow i_X^* F
\]
is a unique morphism of algebraic stacks. Note that
\[
\text{Arrows} = \left( \text{Sch}/S \right)_{\text{fpf}} \otimes \left( \text{Sch}/S \right)_{\text{fpf}}
\]
and
\[
V = \Gamma(S, \mathcal{O}) \hookrightarrow (U, \text{Spec}(A))
\]
is an open subset of $X$. Thus $U$ is affine. This is a continuous map of $X$ is the inverse, the groupoid scheme $S$.

Proof. See discussion of sheaves of sets. □

The result for prove any open covering follows from the less of Example ???. It may replace $S$ by $X_e space st etale$ which gives an open subspace of $X$ and $T$ equal to $S_{Zar}$, see Descent, Lemma ???. Namely, by Lemma ?? 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)
\[
\text{Set}(A) = \Gamma(X, \mathcal{O}_X, \mathcal{O}_X).
\]
When in this case of to show that $\mathcal{O} \rightarrow C_{Z,X}$ is stable under the following result in the second conditions of (1), and (3). This finishes the proof. By Definition ?? (without element is when the close 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 $Z \subset X$ 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
\[
(1) f \text{ is locally of finite type. Since } S = \text{Spec}(R) \text{ and } Y = \text{Spec}(R).
\]
Proof. This is form all sheaves of sheaves on $X$. But given a scheme $U$ and a surjective étale morphism $U \rightarrow X$. Let $U \cap U = \bigcup_{i=1}^{n} U_i$ be the scheme $X$ over $S$ at the schemes $X_i \rightarrow X$ and $U = \lim X_i$. □

The following lemma surjective restrecomposes of this implies that $F_{x_0} = F_{x_0} = F_{X,...,0}$.

**Lemma 0.2.** Let $X$ be a locally Noetherian scheme over $S$, $E = F_{X/S}$. Set $T = \mathcal{O}_X \otimes_{\mathcal{O}_S} \mathcal{O}_T$. Since $T \subset T'$ are nonzero over $i_0 \leq p$ is a subset of $\mathcal{O}_{X_0} \otimes A_2$ 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
\[
\text{D}(\mathcal{O}_X(D)) = \mathcal{O}_X(D)
\]
where $K$ is an $F$-algebra where $\delta_{n+1}$ 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 sheaf of $\mathcal{O}$-modules. We have to show that

$$O_{\mathcal{O}_X} = O_X(\mathcal{L})$$

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

$$O_X(F) = \{\text{morph}_\text{1} \times_{O_X} (G, F)\}$$

where $G$ defines an isomorphism $F \to F$ of $\mathcal{O}$-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 $U \subset X$ 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 $F$ be a quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent

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

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

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

$$\text{Spec}(K_x) \quad \text{Mor}_{\text{Sets}} \quad 	ext{d}(O_{X_{x}})$$

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

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

Proof. We have see that $X = \text{Spec}(R)$ and $F$ is a finite type representable 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 $G$ is a finite presentation, see Lemmas ??.
A reduced above we conclude that $U$ is an open covering of $C$. The functor $F$ is a “field

$$O_{X_{x}} \to F_{x} - 1(\text{Spec}_{x}) \to \text{Spec}_{x} O_{X_{x}}(F_{x})$$

is an isomorphism of covering of $O_{X_{x}}$. 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 opens of finite type over $S$.
If $F$ is a scheme theoretic image points.

If $F$ is a finite direct sum $O_{X_{x}}$ is a closed immersion, see Lemma ??.

This is a sequence of $F$ is a similar morphism.
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 & (1 << 1))
            pipe = (in_use & UMXTHREAD_UNCCA) +
                ((count & 0x0000000000ff00ff) & 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 ");
}
/*
 * Copyright (c) 2006-2010, Intel Mobile Communications. All rights reserved.
 *
 * This program is free software; you can redistribute it and/or modify it
 * under the terms of the GNU General Public License version 2 as published by
 * the Free Software Foundation.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software Foundation,
 * Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
 */

#include <linux/kexec.h>
#include <linux/errno.h>
#include <linux/io.h>
#include <linux/platform_device.h>
#include <linux/multi.h>
#include <linux/ckevent.h>

#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>
```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_sigs() 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 __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;
  }
```
Searching for interpretable cells

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

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

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.

line length tracking 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

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

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

```c
#ifdef CONFIG_AUDITSYSCALL
static inline int audit_match_class_bits(int class, u32 *mask)
{
    int i;
    if (classes[class]) {
        for (i = 0; i < AUDIT_BITMASK_SIZE; i++)
            if (mask[i] & classes[class][i])
                return 0;
    }
    return 1;
}
#endif
```

code depth 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
Plan for Today

• Recurrent Neural Networks (RNNs)
  – Example Problem: (Character-level) Language modeling
  – Learning: (Truncated) BackProp Through Time (BPTT)
  – Visualizing RNNs
  – Inference: Beam Search
  – Problems with gradients in “vanilla” RNNs
  – LSTMs (and other RNN variants)
  – Example: Image Captioning
  – Multilayer RNNs
Beam Search

- Proceed from left to right
- Maintain N partial captions
- Expand each caption with possible next words
- Discard all but the top N new partial translations
  - Maintain score for each, e.g. product of probabilities

https://geekysawesome.blogspot.com/2016/10/using-beam-search-to-generate-most.html
Vanilla RNN Gradient Flow

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

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

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

\[
W
\]

\[
\begin{align*}
h_t &= \tanh(W_{hh} h_{t-1} + W_{xh} x_t) \\
     &= \tanh \left( (W_{hh} \quad W_{hx}) \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} \right) \\
     &= \tanh \left( W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} \right)
\end{align*}
\]

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

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

Gradient clipping: Scale gradient if its norm is too big

Largest singular value < 1: Vanishing gradients

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

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n
Long Short Term Memory (LSTM)
[Hochreiter et al., 1997]

i: Input gate, whether to write to cell
f: Forget gate, Whether to erase cell
o: Output gate, How much to reveal cell
g: Gate gate (?), How much to write to cell

\[
\begin{pmatrix}
    i \\
    f \\
    o \\
    g
\end{pmatrix}
= \begin{pmatrix}
    \sigma \\
    \sigma \\
    \sigma \\
    \tanh
\end{pmatrix} \cdot W \cdot
\begin{pmatrix}
    h_{t-1} \\
    x_t
\end{pmatrix}
\]

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

\[
h_t = o \odot \tanh(c_t)
\]

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

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

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

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

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

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

- Forget that + memorize this

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

(C) Dhruv Batra  
Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)
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!

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

- A pretty sophisticated cell

Image Credit: Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/)
LSTM Variants #1: Peephole Connections

• Let gates see the cell state / memory

\[
\begin{align*}
    f_t &= \sigma (W_f \cdot [C_{t-1}, h_{t-1}, x_t] + b_f) \\
    i_t &= \sigma (W_i \cdot [C_{t-1}, h_{t-1}, x_t] + b_i) \\
    o_t &= \sigma (W_o \cdot [C_t, h_{t-1}, x_t] + b_o) 
\end{align*}
\]
LSTM Variants #2: Coupled Gates

- Only memorize new if forgetting old

\[
C_t = f_t \ast C_{t-1} + (1 - f_t) \ast \tilde{C}_t
\]
LSTM Variants #3: Gated Recurrent Units

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

\[
\begin{align*}
  z_t &= \sigma (W_z \cdot [h_{t-1}, x_t]) \\
  r_t &= \sigma (W_r \cdot [h_{t-1}, x_t]) \\
  \tilde{h}_t &= \tanh (W \cdot [r_t \ast h_{t-1}, x_t]) \\
  h_t &= (1 - z_t) \ast h_{t-1} + z_t \ast \tilde{h}_t
\end{align*}
\]
Long Short Term Memory (LSTM)

**Vanilla RNN**

\[
th_t = \tanh \left( W \left( \frac{h_{t-1}}{x_t} \right) \right)
\]

**LSTM**

\[
\begin{align*}
\left( \begin{array}{c}
i \\
f \\
o \\
g
\end{array} \right) &= \left( \begin{array}{c}
\sigma \\
\sigma \\
\sigma \\
\tanh
\end{array} \right) W \left( \frac{h_{t-1}}{x_t} \right) \\
ct &= f \odot ct_{t-1} + i \odot g \\
h_t &= o \odot \tanh(ct)
\end{align*}
\]

Hochreiter and Schmidhuber, “Long Short Term Memory”, Neural Computation 1997
Other RNN Variants

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

\[
\text{MUT1:}
\begin{align*}
z &= \text{sigm}(W_{xz}x_t + b_z) \\
r &= \text{sigm}(W_{xr}x_t + W_{hr}h_t + b_r) \\
h_{t+1} &= \tanh(W_{hh}(r \odot h_t) + \tanh(x_t) + b_h) \odot z \\
&+ h_t \odot (1 - z)
\end{align*}
\]

\[
\text{MUT2:}
\begin{align*}
z &= \text{sigm}(W_{xz}x_t + W_{hz}h_t + b_z) \\
r &= \text{sigm}(x_t + W_{hr}h_t + b_r) \\
h_{t+1} &= \tanh(W_{hh}(r \odot h_t) + W_{xh}x_t + b_h) \odot z \\
&+ h_t \odot (1 - z)
\end{align*}
\]

\[
\text{MUT3:}
\begin{align*}
z &= \text{sigm}(W_{xz}x_t + W_{hz} \tanh(h_t) + b_z) \\
r &= \text{sigm}(W_{xr}x_t + W_{hr}h_t + b_r) \\
h_{t+1} &= \tanh(W_{hh}(r \odot h_t) + W_{xh}x_t + b_h) \odot z \\
&+ h_t \odot (1 - z)
\end{align*}
\]
Multilayer RNNs

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

\( h \in \mathbb{R}^n \)
\( W^l \in [n \times 2n] \)
Summary

- RNNs allow a lot of flexibility in architecture design
- Vanilla RNNs are simple but don’t work very well
- Common to use LSTM or GRU: their additive interactions improve gradient flow
- Backward flow of gradients in RNN can explode or vanish. Exploding is controlled with gradient clipping. Vanishing is controlled with additive interactions (LSTM)
- Better/simpler architectures are a hot topic of current research
- Better understanding (both theoretical and empirical) is needed.