Recurrent Neural Networks

Resana Machine Learning Workshop

prepared by Hosein Hasani

Summer 1399

Outline

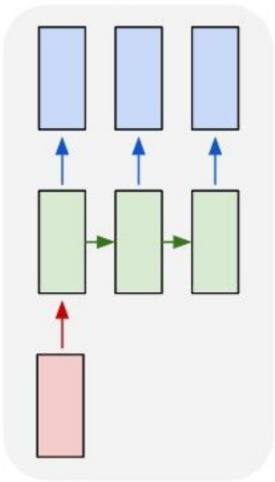
- Processing Sequential Data
- Vanilla RNN
- RNN: Computational Graph
- Example: Language Modeling
- Backpropagation Through Time
- Searching for Interpretable Cells
- Problem with Vanilla RNN Gradient Flow
- Long Short Term Memory (LSTM)
- Extended Models

Processing Sequential Data

- In many situations one must consider a series of inputs to produce an output.
- RNNs have a memory which captures information about what has been seen so far.
- In the **brain**, most of the information processing and cognitive functions are performed through complex neural networks with recurrent **feedbacks**.

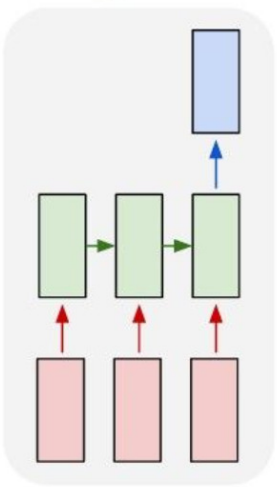
Processing Sequential Data

one to many



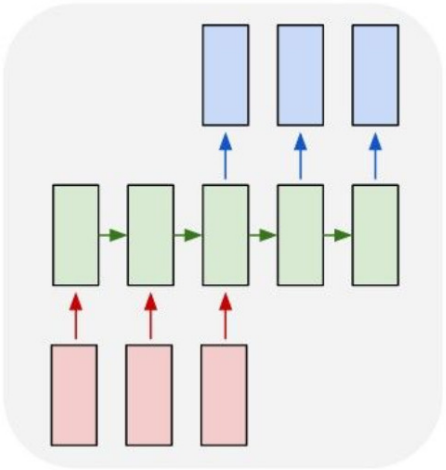
e.g. image captioning

many to one



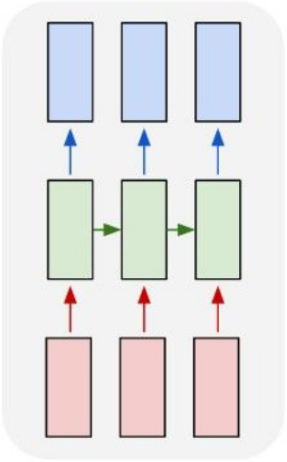
e.g. sentiment classification

many to many



e.g. machine translation

many to many



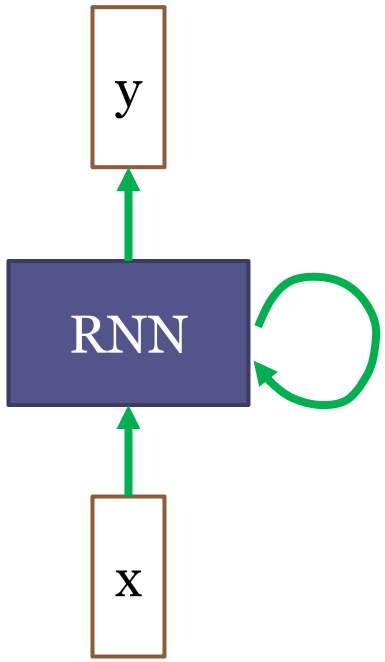
e.g. event classification in videos

Recurrent Neural Network

- Simple configuration: **Vanilla RNN**
- We can process a sequence of vectors x by applying a recurrence formula at every time step:

hidden vector

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



- Note that the same function and the same set of parameters are used at every time step

Recurrent Neural Network

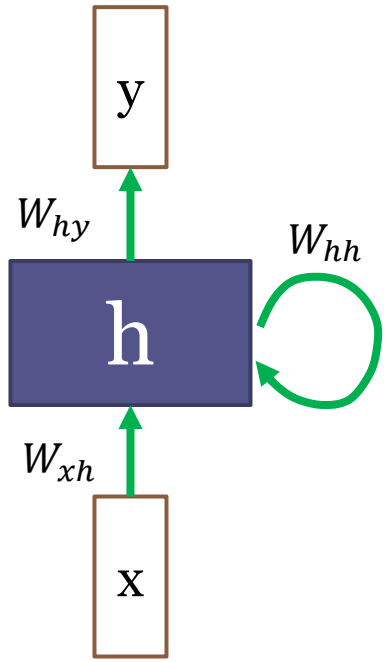
- Every time step we calculate h_t:

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

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

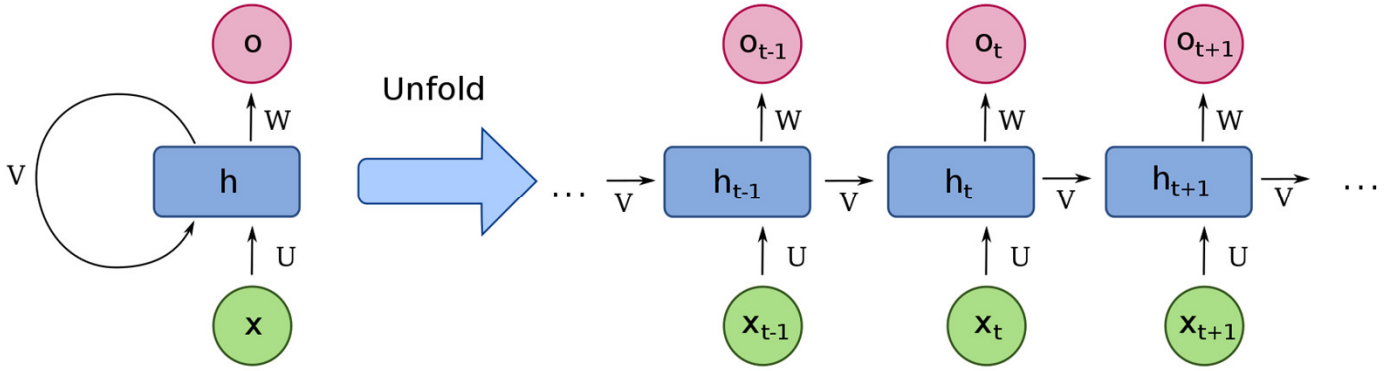
- When we need y, it's obtained by:

$$y_t = W_{hy}h_t$$



RNN: Computational Graph

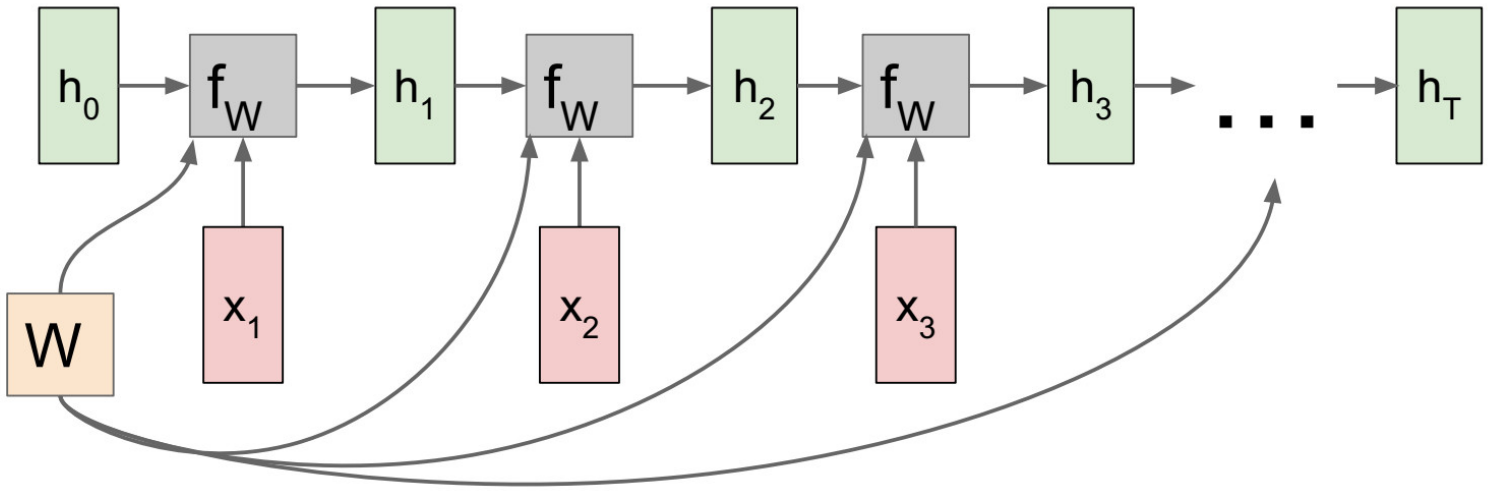
- It's useful to unfold a recurrent computation into a computational graph that has a repetitive structure, typically corresponding to a chain of events.



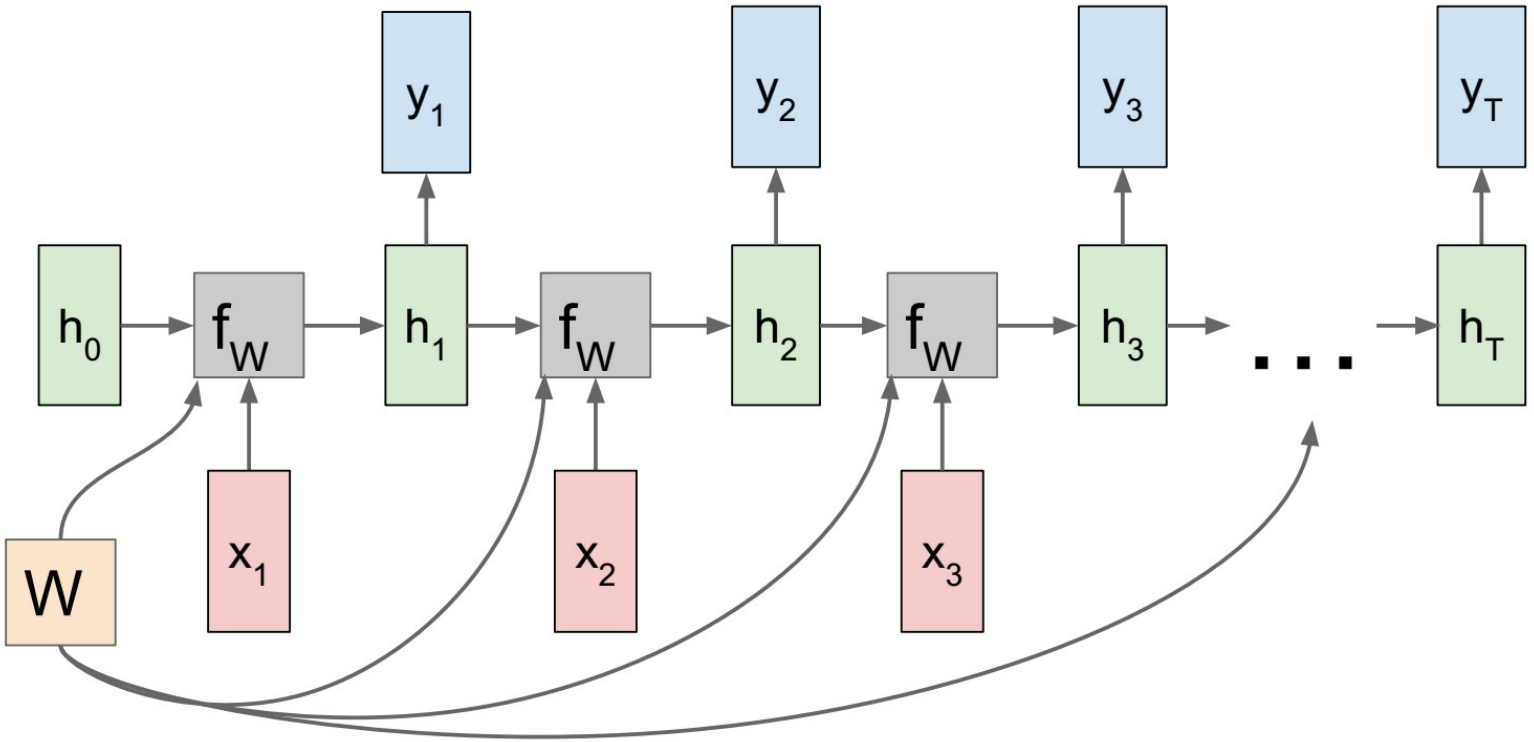
[wikipedia.org]

RNN: Computational Graph

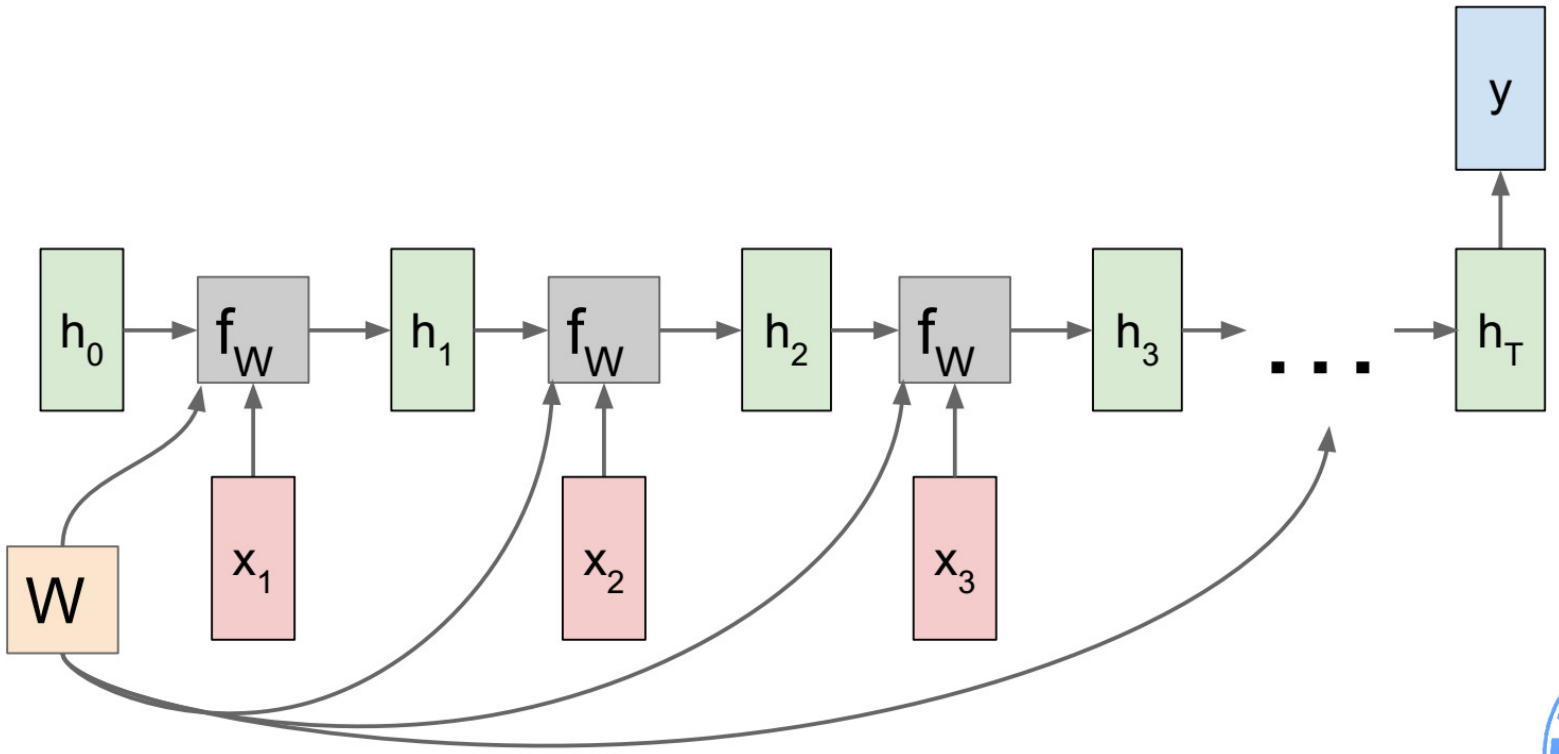
- Re-use the same weight matrix at every time-step



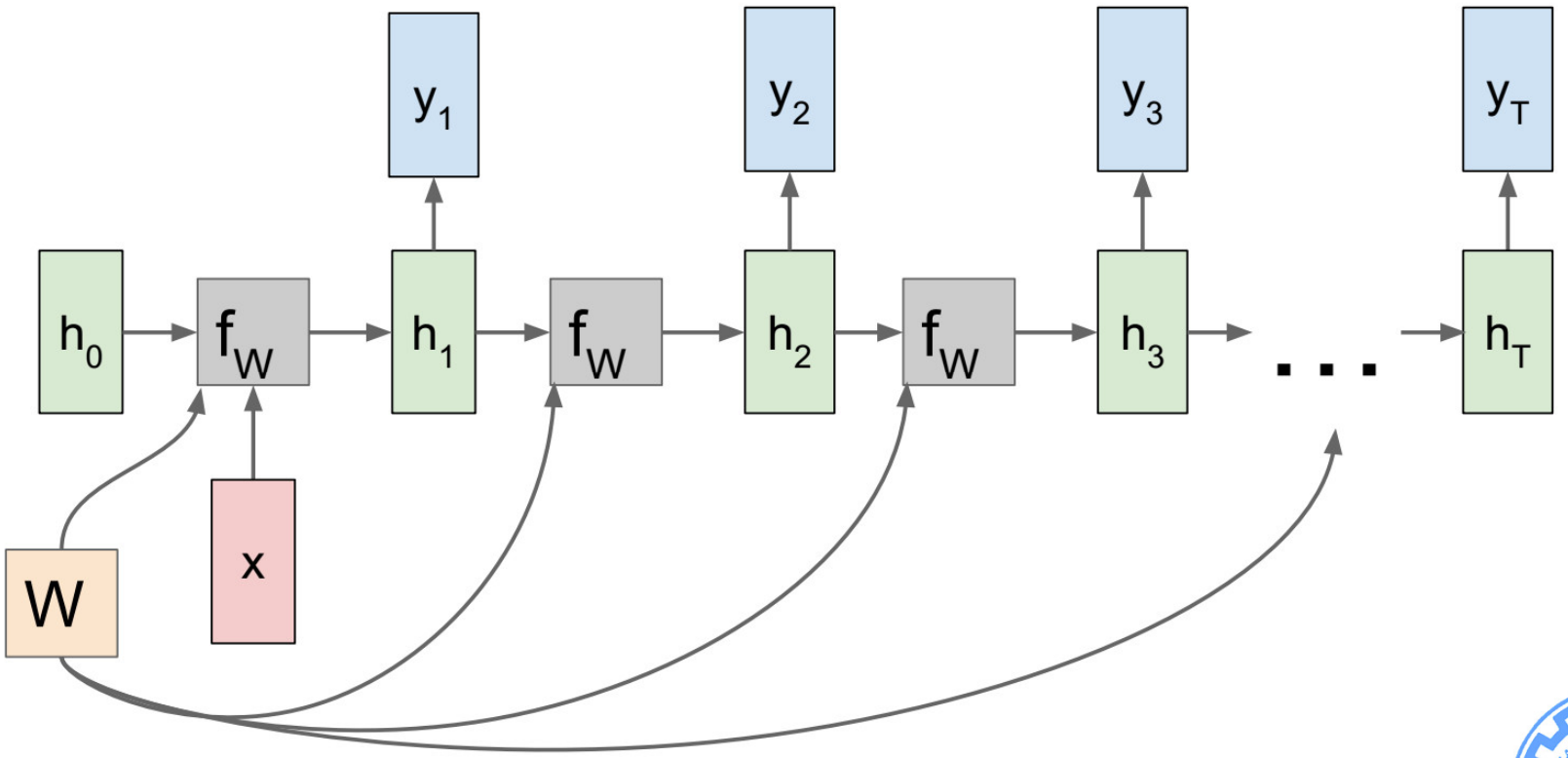
RNN: Computational Graph: Many to Many



RNN: Computational Graph: Many to One



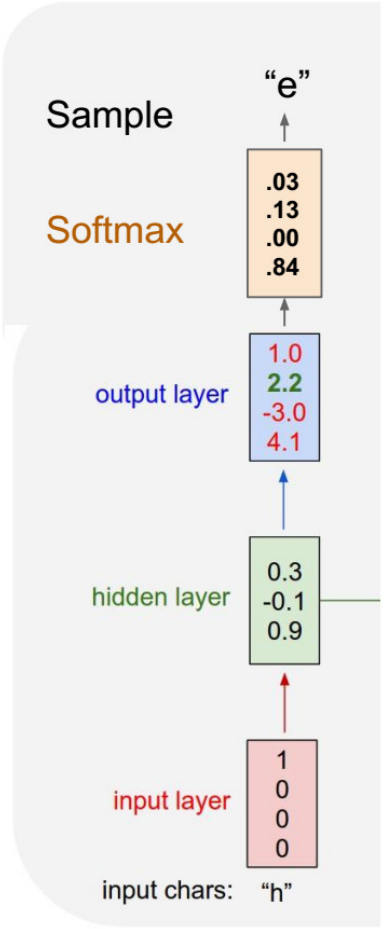
RNN: Computational Graph: One to Many



Example: Character-level Language Model

- Predicting the sequence “hello”

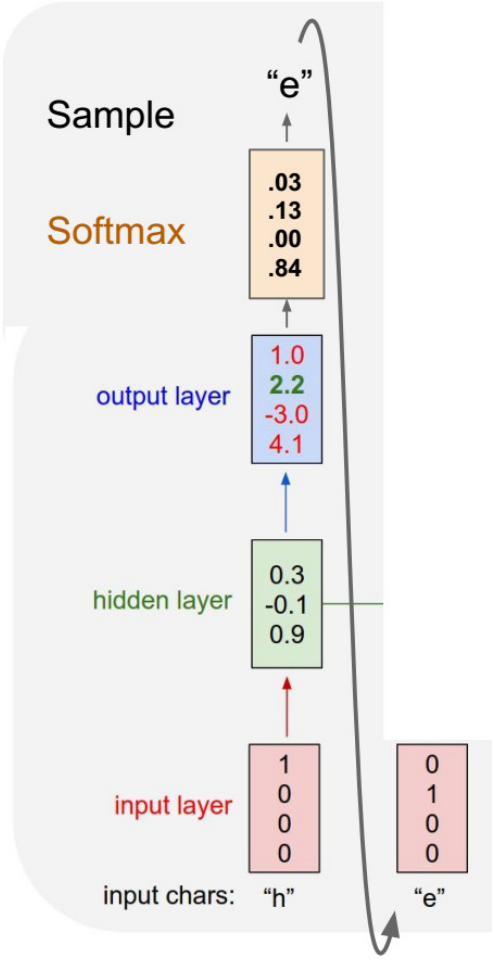
Vocabulary = [h, e, l, o]



Example: Character-level Language Model

- Predicting the sequence “hello”

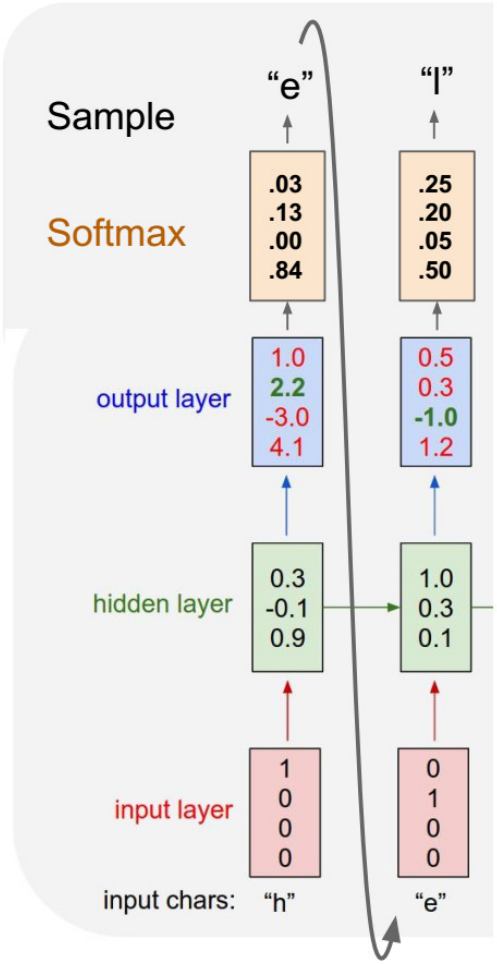
Vocabulary = [h, e, l, o]



Example: Character-level Language Model

- Predicting the sequence “hello”

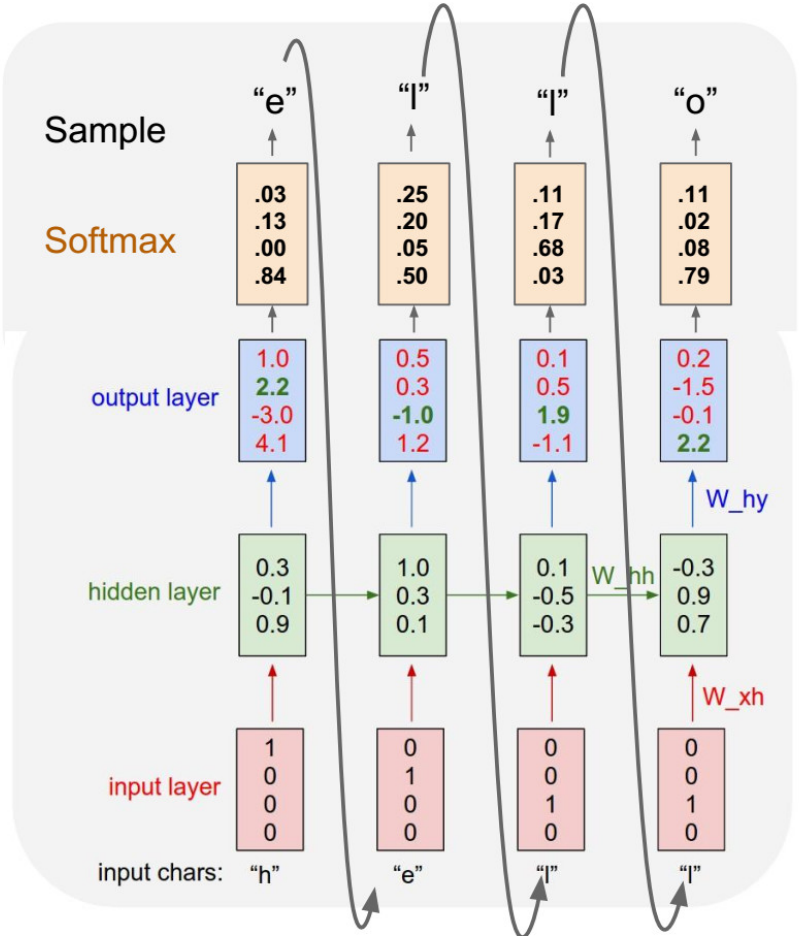
Vocabulary = [h, e, l, o]



Example: Character-level Language Model

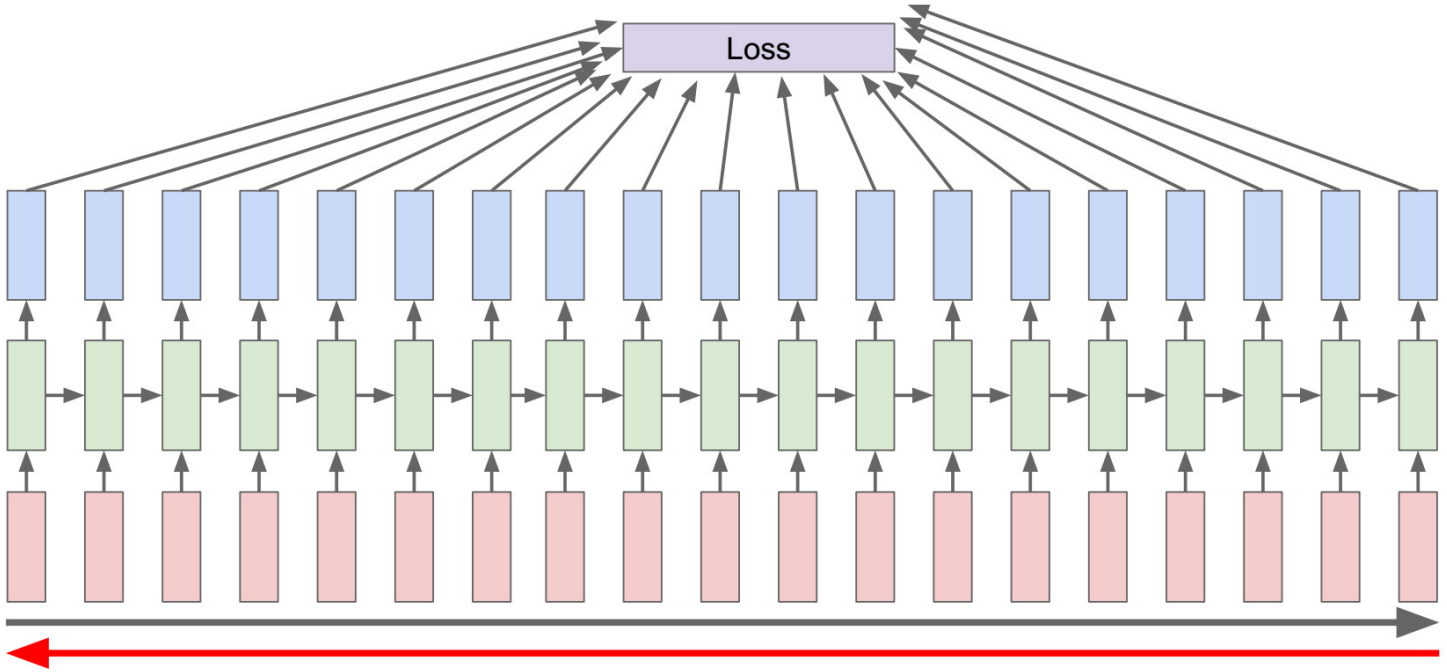
- Predicting the sequence “hello”

Vocabulary = [h, e, l, o]



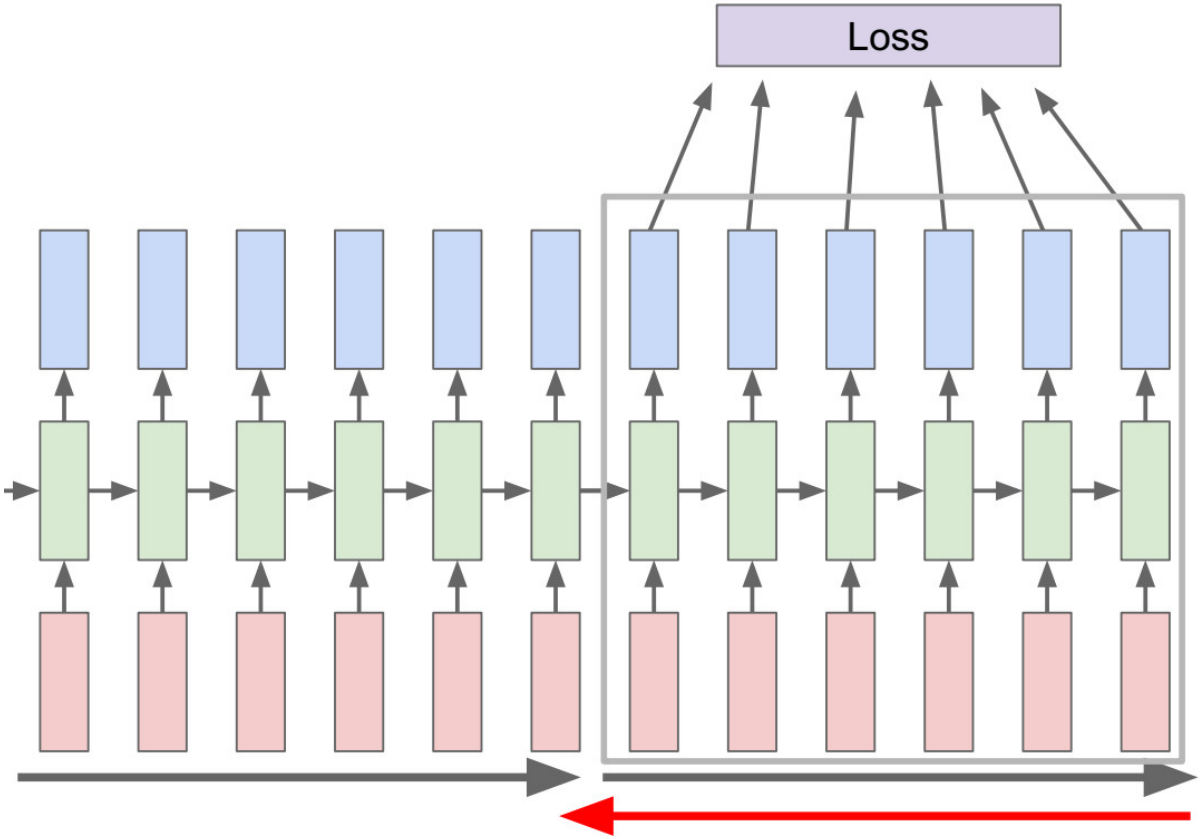
Optimization: Backpropagation Through Time

- 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.



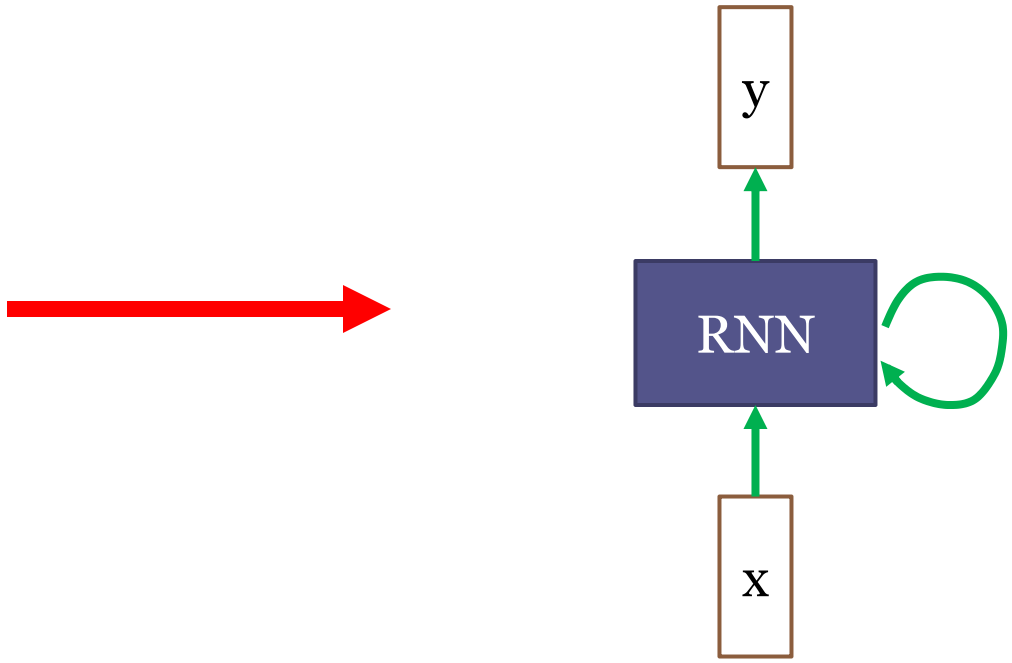
Example: Language Modeling

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 that art now the world's fresh ornament,
And only herald to the gaudy spring,
Within thine own bud buriest thy content,
And tender churl mak'st waste in niggarding:
Pity the world, or else this glutton 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 proud 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 treasure 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.



Example: Language Modeling

at first:

```
tyntd-iafhatawiaoihrdemot  lytdws  e ,tfti, astai f ogoh eoase rrranbyne 'nhthnee e
plia tklrgd t o idoe ns,smtt   h ne etie h,hregtrs nigtike,aoaenns lng
```

train more

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

train more

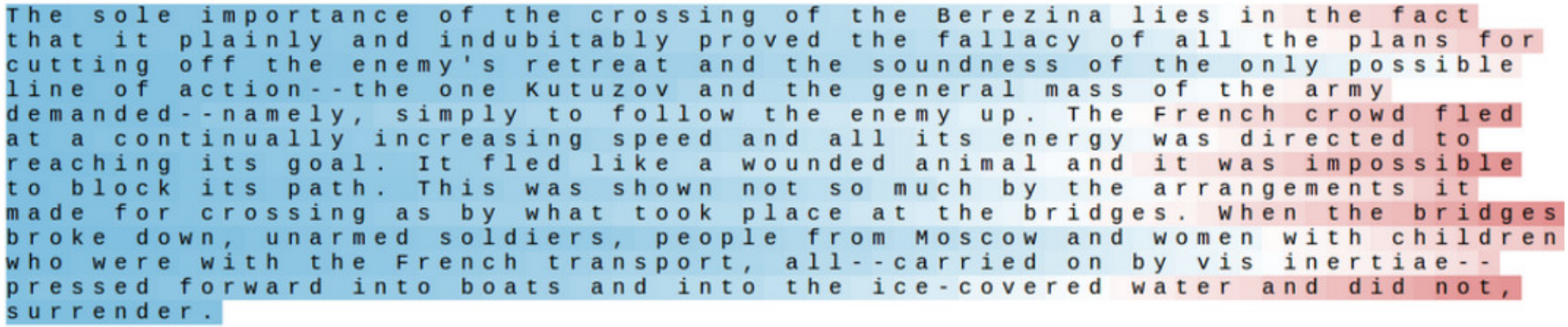
```
Aftair fall unsuch that the hall for Prince Velzonski's that me of
her hearly, and behs to so arwage fiving were to it beloge, pavu say falling misfort
how, and Gogition is so overelical and ofter.
```

train more

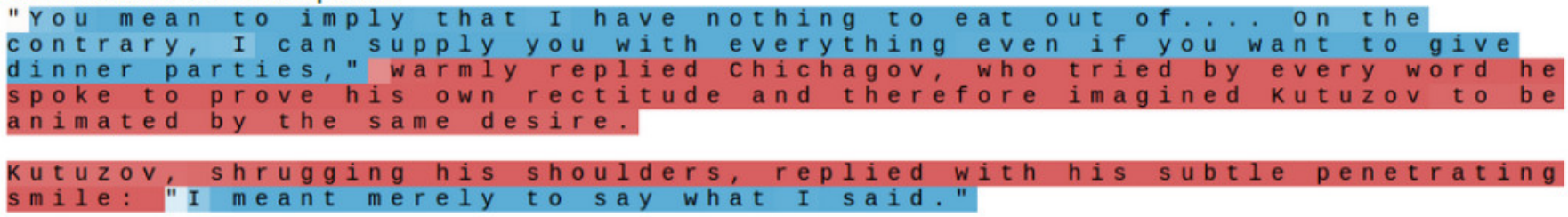
```
"Why do what that day," replied Natasha, and wishing to himself the fact the
princess, Princess Mary was easier, fed in had oftened him.
Pierre aking his soul came to the packs and drove up his father-in-law women.
```

Searching for Interpretable Cells

Cell sensitive to position in line:



Cell that turns on inside quotes:



[Karpathy et al. ICLR 2016]

Searching for Interpretable Cells

Cell that turns on inside comments and quotes:

```
/* Duplicate LSM field information.  The lsm_rule is opaque, so
 * re-initialized. */
static inline int audit_dupe_lsm_field(struct audit_field *df,
                                       struct audit_field *sf)
{
  int ret = 0;
  char *lsm_str;
  /* our own copy of lsm_str */
  lsm_str = kstrdup(sf->lsm_str, GFP_KERNEL);
  if (unlikely(!lsm_str))
    return -ENOMEM;
  df->lsm_str = lsm_str;
  /* our own (refreshed) copy of lsm_rule */
  ret = security_audit_rule_init(df->type, df->op, df->lsm_str,
                                 (void **)&df->lsm_rule);
  /* Keep currently invalid fields around in case they
   * become valid after a policy reload. */
  if (ret == -EINVAL) {
    pr_warn("audit rule for LSM \'%s\' is invalid\n",
            df->lsm_str);
    ret = 0;
  }
  return ret;
}
```

Cell that is sensitive to the depth of an expression:

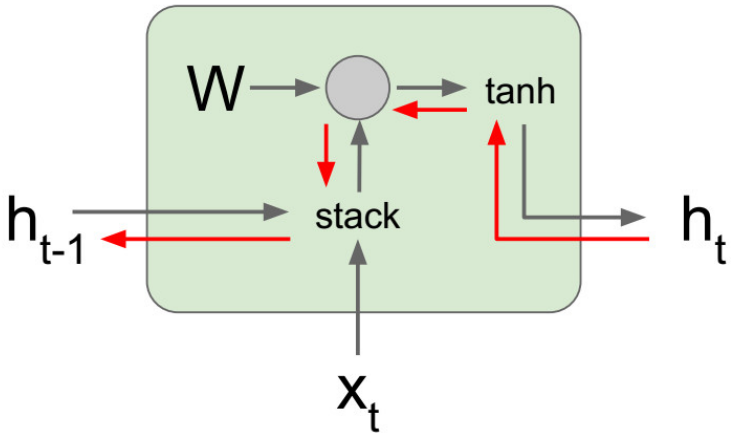
```
#ifdef CONFIG_AUDITSYSCALL
static inline int audit_match_class_bits(int class, u32 *mask)
{
  int i;
  if (classes[class]) {
    for (i = 0; i < AUDIT_BITMASK_SIZE; i++)
      if (mask[i] & classes[class][i])
        return 0;
  }
  return 1;
}
```

[Karpathy et al. ICLR 2016]

Vanilla RNN Gradient Flow

[Bengio et al., 1997]

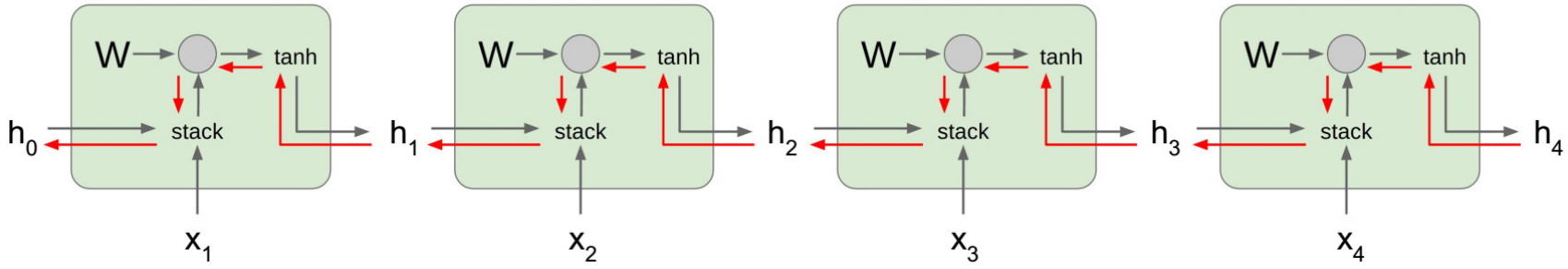
Backpropagation from h_t to h_{t-1} multiplies by W (actually W_{hh}^T)



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

Vanilla RNN Gradient Flow

[Bengio et al., 1997]



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

Long Short Term Memory (LSTM)

[Hochreiter et al., 1997]

Vanilla RNN

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

LSTM

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

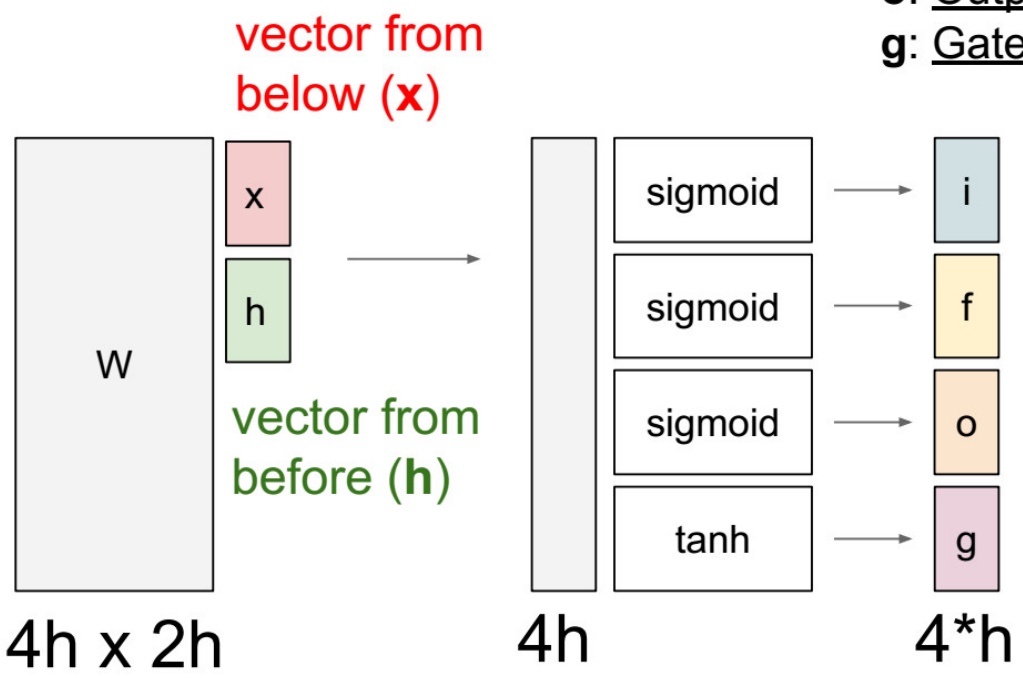
$$c_t = f \odot c_{t-1} + i \odot g$$

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

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



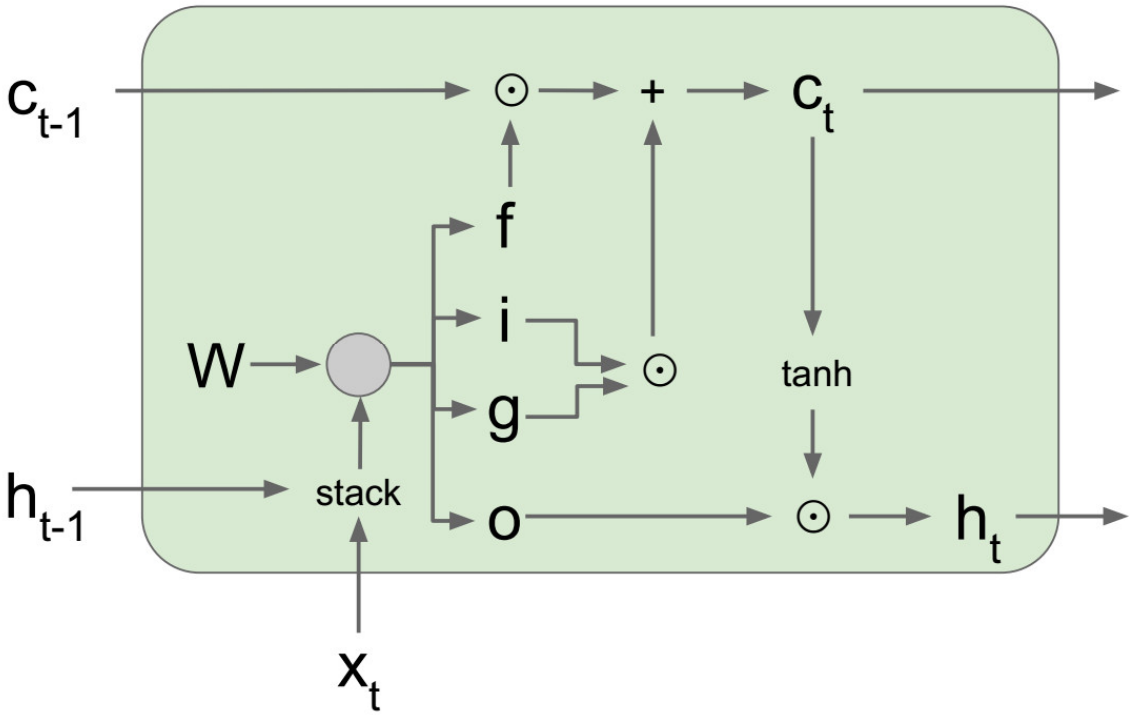
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

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

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

Long Short Term Memory (LSTM)

[Hochreiter et al., 1997]



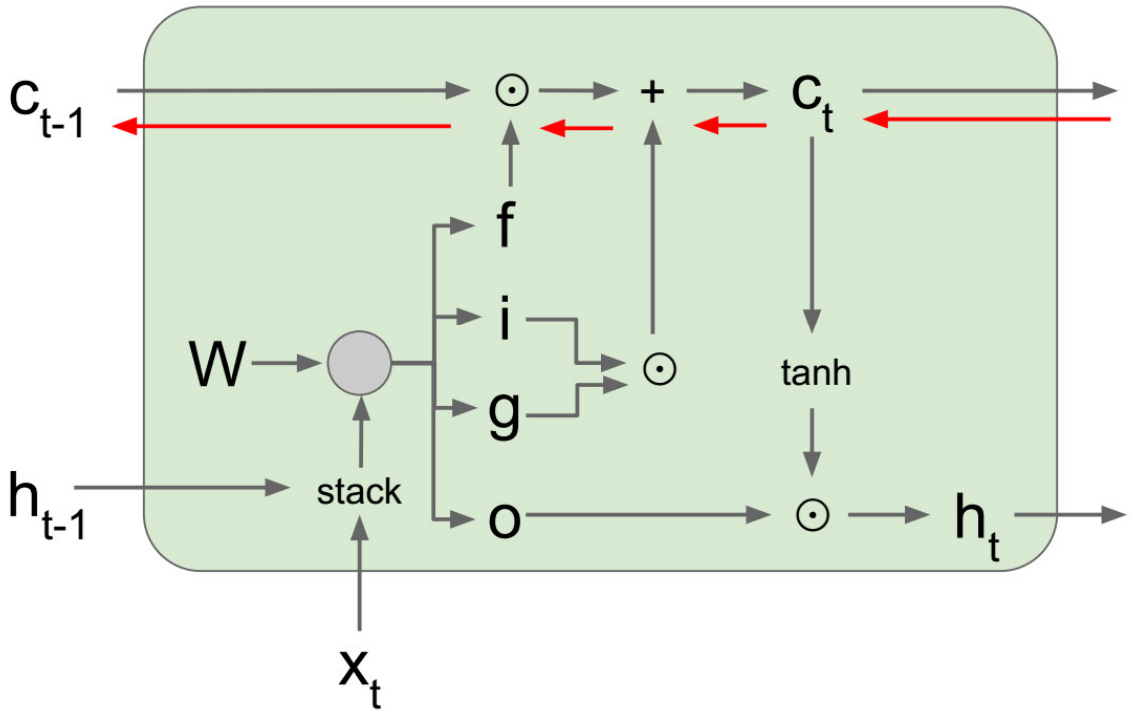
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

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

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

Long Short Term Memory (LSTM)

[Hochreiter et al., 1997]



Backpropagation from c_t to c_{t-1} only elementwise multiplication by f, no matrix multiply by W

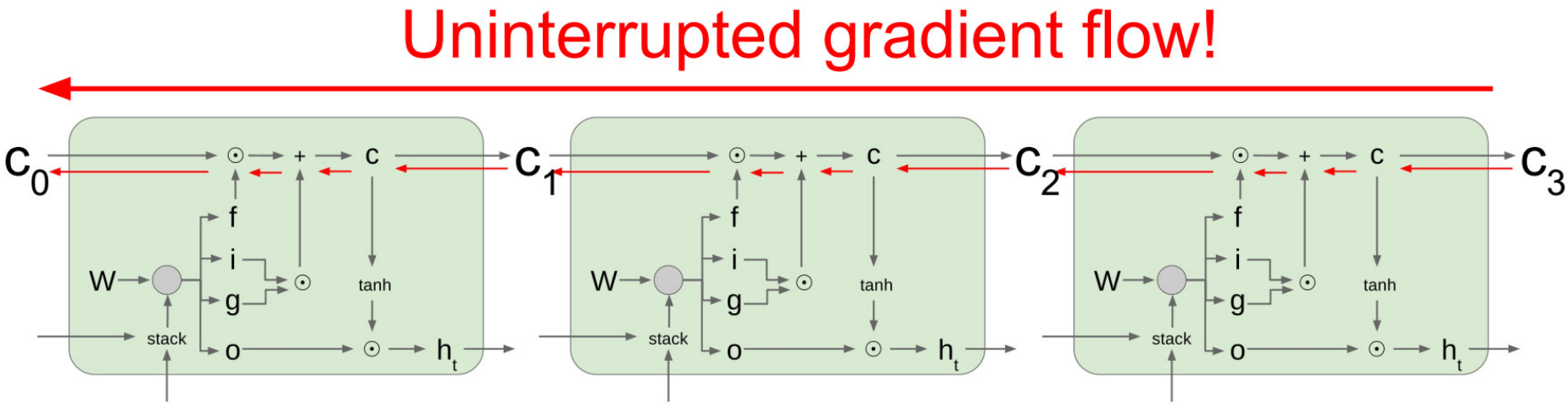
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

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

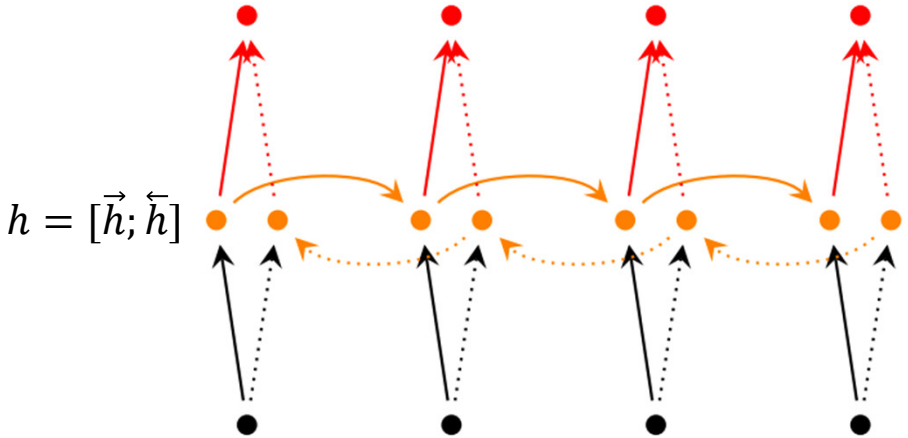
$$h_t = o \odot \tanh(c_t)$$

Long Short Term Memory (LSTM)

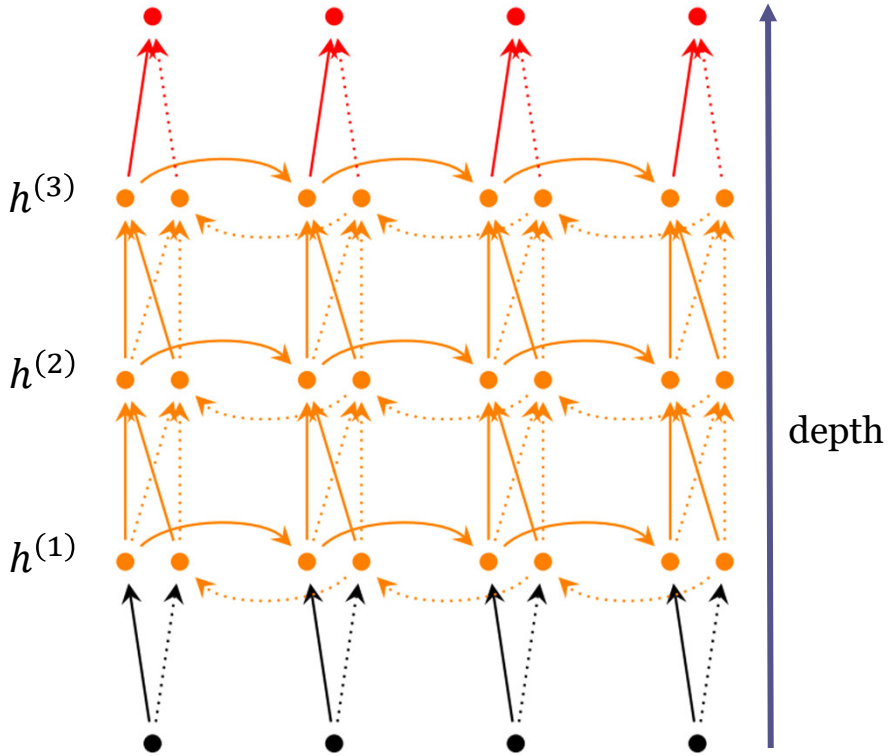
[Hochreiter et al., 1997]



Extended Models



Bidirectional



Multi layer

RNNs ...

- Excellent models for time-series analysis tasks.
- Allow a lot of flexibility in architecture design.
- Vanilla RNNs are simple but don’t work very well.
- Backward flow of gradients in RNN can explode or vanish.
- Common to use LSTM or GRU: their additive interactions improve gradient flow.

Main Resources

- http://deeplearning.cs.cmu.edu/
- http://cs231n.stanford.edu/
- Goodfellow, et al. Deep learning. (chapter 10)