Recurrent Neural Networks

Changyou Chen

Department of Computer Science and Engineering
Universitpy at Buffalo, SUNY
changyou@buffalo.edu

March 14, 2019

Academic Integrity

1. Zero tolerance on cheating/plagiarism:
   - Fail the course on any homework assignment/lab, project, or exam even for first attempt, & report to the department.
   - May lose TAship/RAship and be expelled from the program.
   - Consult the CSE Department and UB Policies for details on other consequences of academic misconduct.
2. Group study/discussion is encouraged, but the submission must be your own work.
3. No exchanges of source codes, even if for viewing only.
4. May take only a small, non-critical portion of the codes from the Internet (this may not be allowed in other CSE courses), but only if you disclose the source (otherwise, considered cheating).
5. Bottomline: TAs and instructor are also very good at, with the help of effective tools (e.g. MOSS) to detect cheating).
6. DO NOT violate AI polices unless you can live with the consequence (as you will most likely get caught).

Ways to Fail[1]

1. Start your project the weekend before it's due.
2. Avoid your TAs and their office hours.
3. Stay quiet in class when your professor says something you don't understand.
4. Wait until the deadline to submit for the first time.

[1] https://odin.cse.buffalo.edu/teaching/cse-562/2017sp/slides/2017-01-31-Intro.html#/5

Recurrent Neural Networks[2]

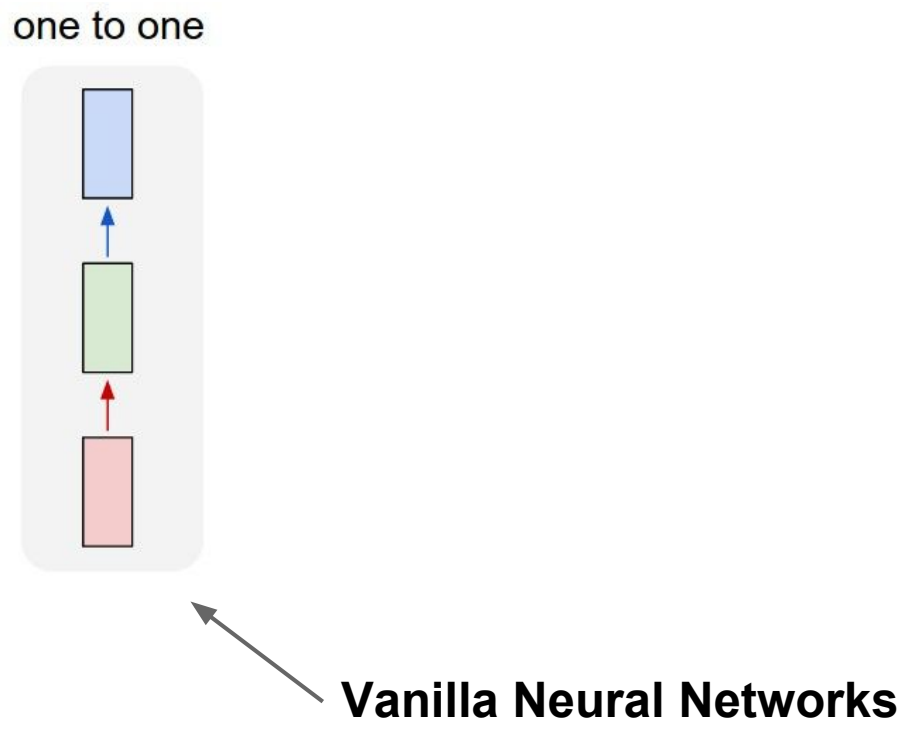
Recurrent neural networks are extensions of feed forward neural networks such that

- it can process sequential data efficiently.
- weights of the neural networks between different time are shared:
  - In CNN, weights between different regions are shared.

[2] Partially adapted from http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture10.pdf

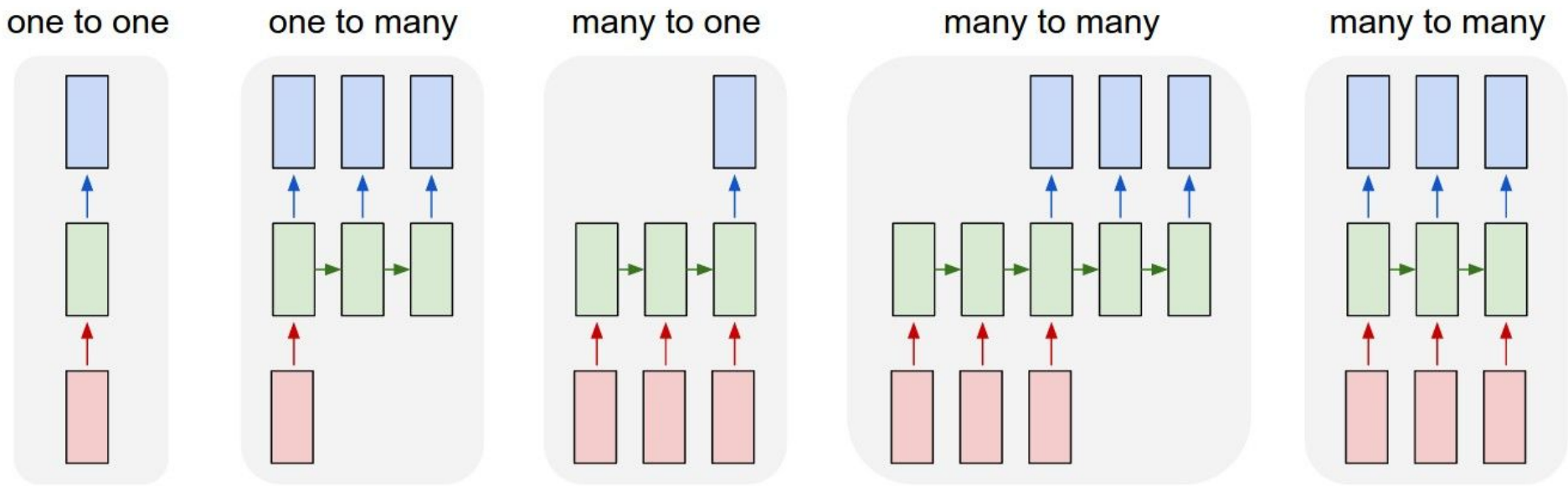
“Vanilla” Neural Network

pink: input; green: hidden; blue: output



Recurrent Neural Networks: Process Sequences

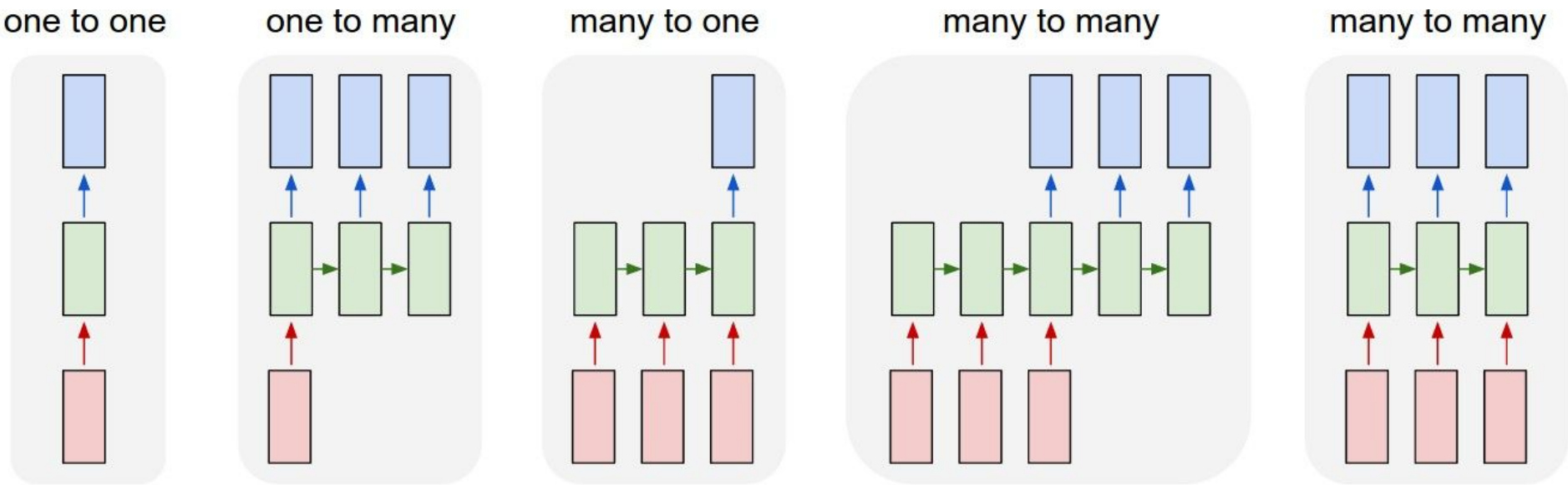
pink: input; green: hidden; blue: output



e.g. **Image Captioning**
image -> sequence of words

Recurrent Neural Networks: Process Sequences

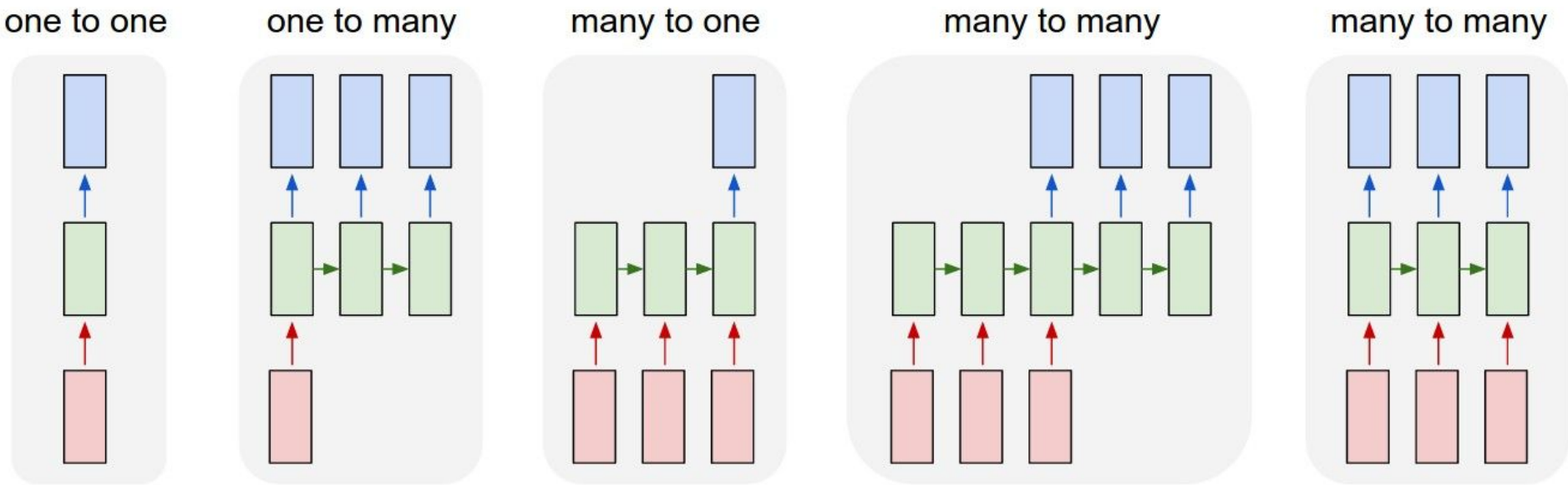
pink: input; green: hidden; blue: output



e.g. **Sentiment Classification**
sequence of words -> sentiment

Recurrent Neural Networks: Process Sequences

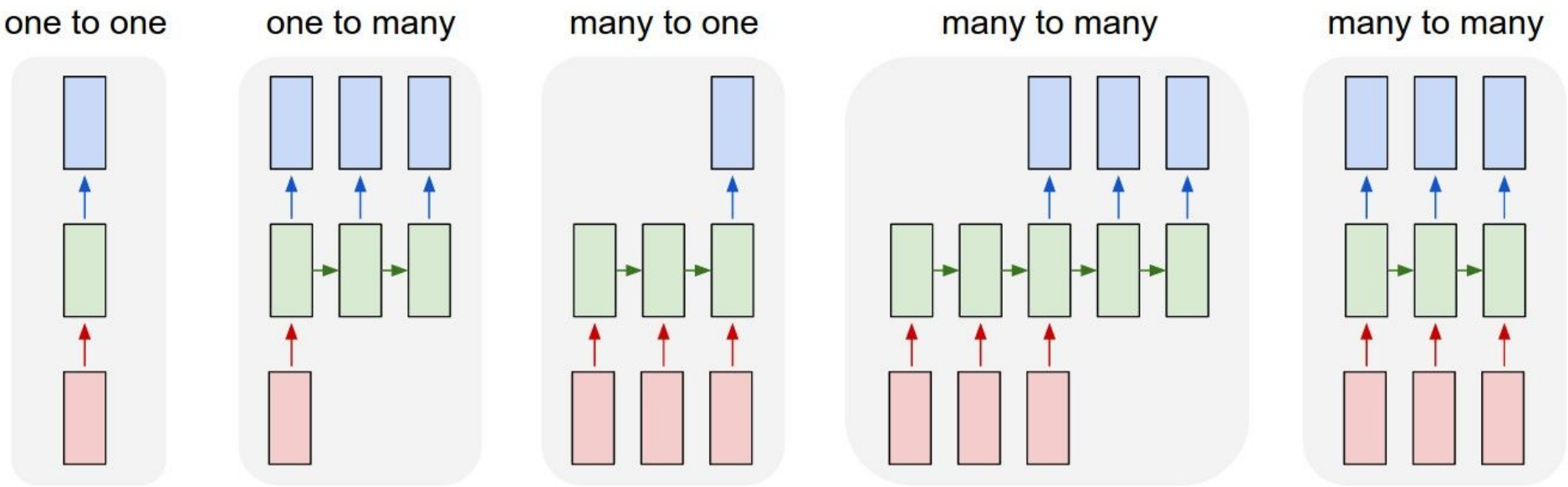
pink: input; green: hidden; blue: output



e.g. **Machine Translation**
seq of words -> seq of words

Recurrent Neural Networks: Process Sequences

pink: input; green: hidden; blue: output



e.g. **Video classification on frame level**

Sequential Processing of Non-Sequence Dat

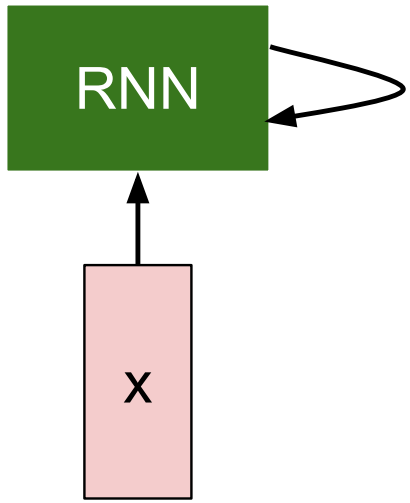
Making non-sequence data sequential

Classify images by taking a series of “glimpses”

Ba, Mnih, and Kavukcuoglu, “Multiple Object Recognition with Visual Attention”, ICLR 2015.
Gregor et al, “DRAW: A Recurrent Neural Network For Image Generation”, ICML 2015

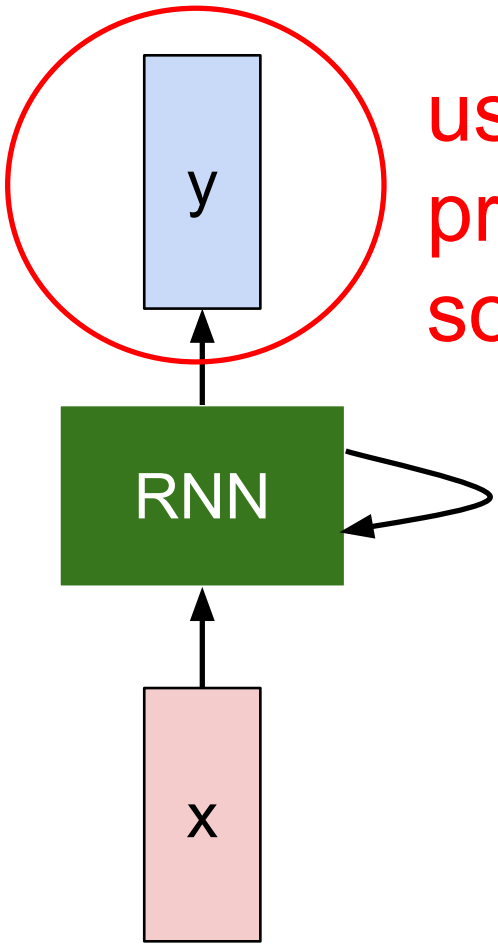
Recurrent Neural Networks

- Inputs and outputs are usually indexed by time.
- The recurrent unit generates states $\mathbf{h}_t$ for each time.



Recurrent Neural Networks

- Inputs and outputs are usually indexed by time.
- The recurrent unit generates states $\mathbf{h}_t$ for each time.



usually want to predict a vector at some time steps

Recurrent Neural Networks

Process a sequence of vectors x by applying a recurrence formula at every time step:

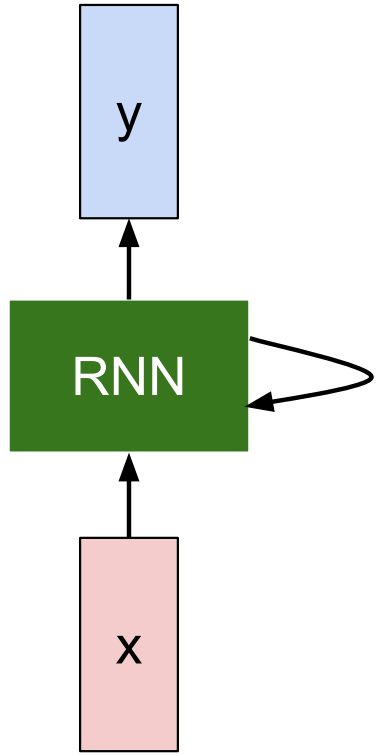
$$h_t = f_W(h_{t-1}, x_t)$$

new state

some function with parameters W

old state

input vector at some time step



Recurrent Neural Networks

Process a sequence of vectors x by applying a recurrence formula at every time step:

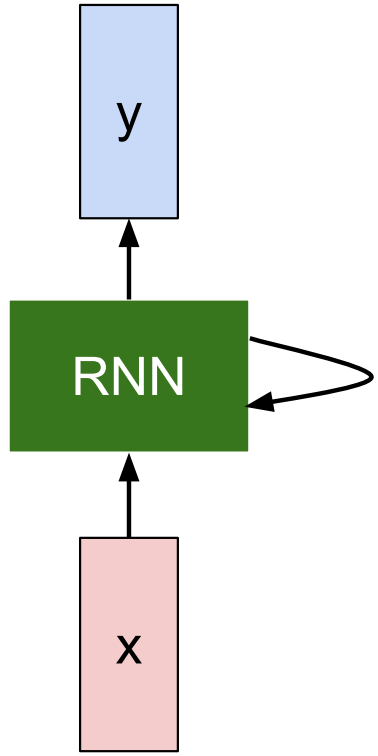
$$h_t = f_W(h_{t-1}, x_t)$$

new state

some function with parameters W

old state

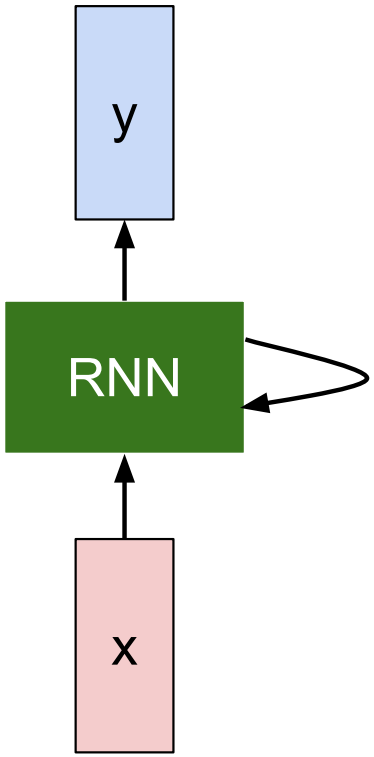
input vector at some time step



Notice: the same function and the same set of parameters are used at every time step. $\Rightarrow$ Shared!

Vanilla Recurrent Neural Network

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



$$\mathbf{h}_t = f_{\mathbf{W}}(\mathbf{h}_{t-1}, \mathbf{x}_t)$$

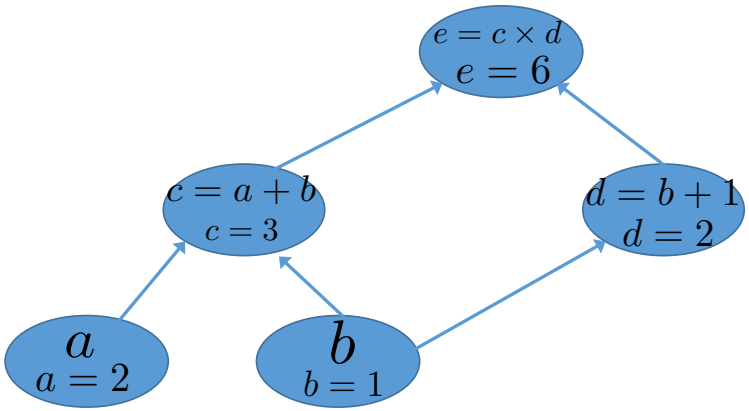
$$\Downarrow \textit{e.g.}$$

$$\mathbf{h}_t = \tanh\left(\mathbf{W}_{hh}\,\mathbf{h}_{t-1} + \mathbf{W}_{xh}\,\mathbf{x}_t\right)$$

$$\mathbf{y}_t = \mathbf{W}_{hy}\,\mathbf{h}_t$$

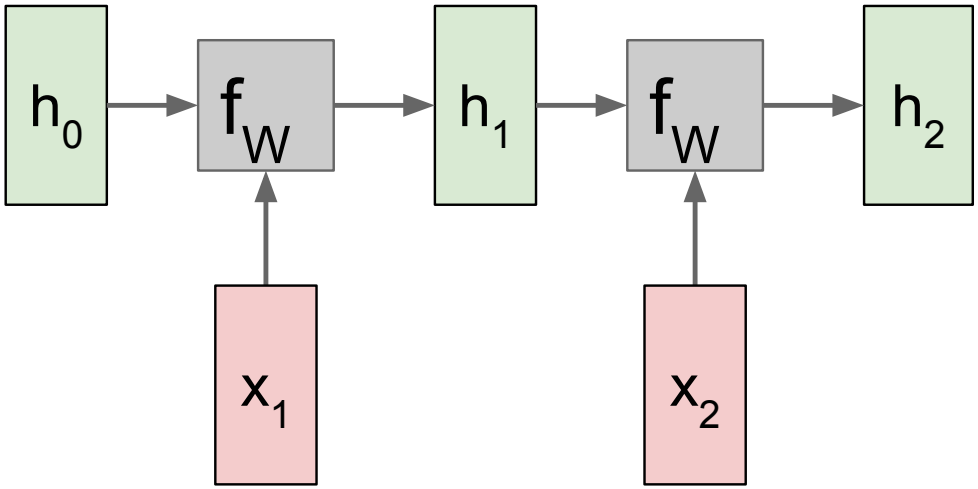
Recap: Computational Graph

- Each node is either
  - a variable: scalar, vector, matrix, tensor, or other type
  - or an operation
    - ★ simple function of one or more variables
    - ★ functions more complex than operations are obtained by composing operations
  - if variable y is computed by applying operation to variable x then draw directed edge from x to y.



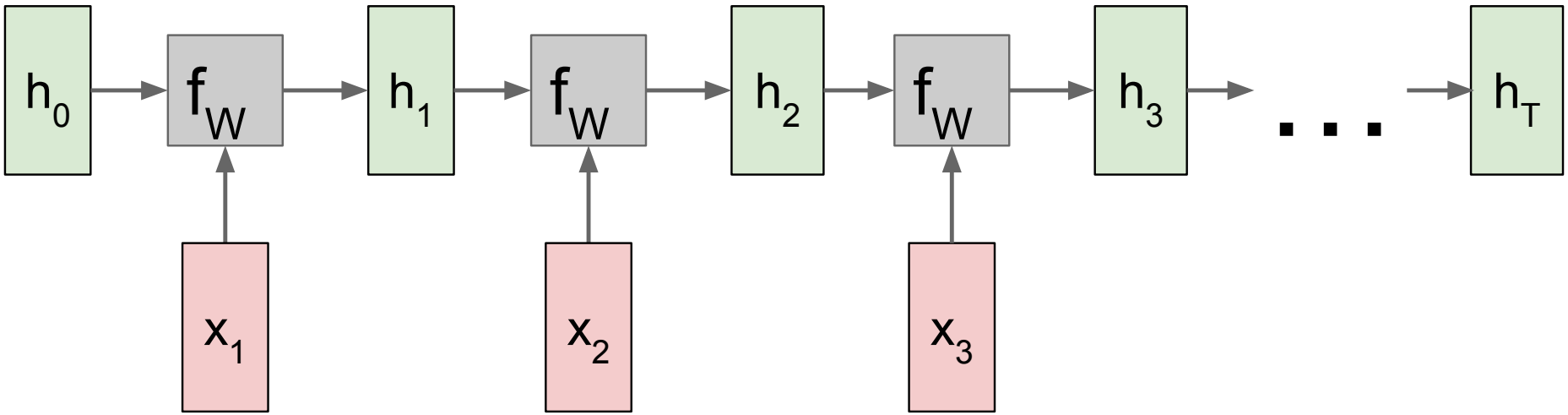
Unrolling RNN: Computational Graph

First two time steps:



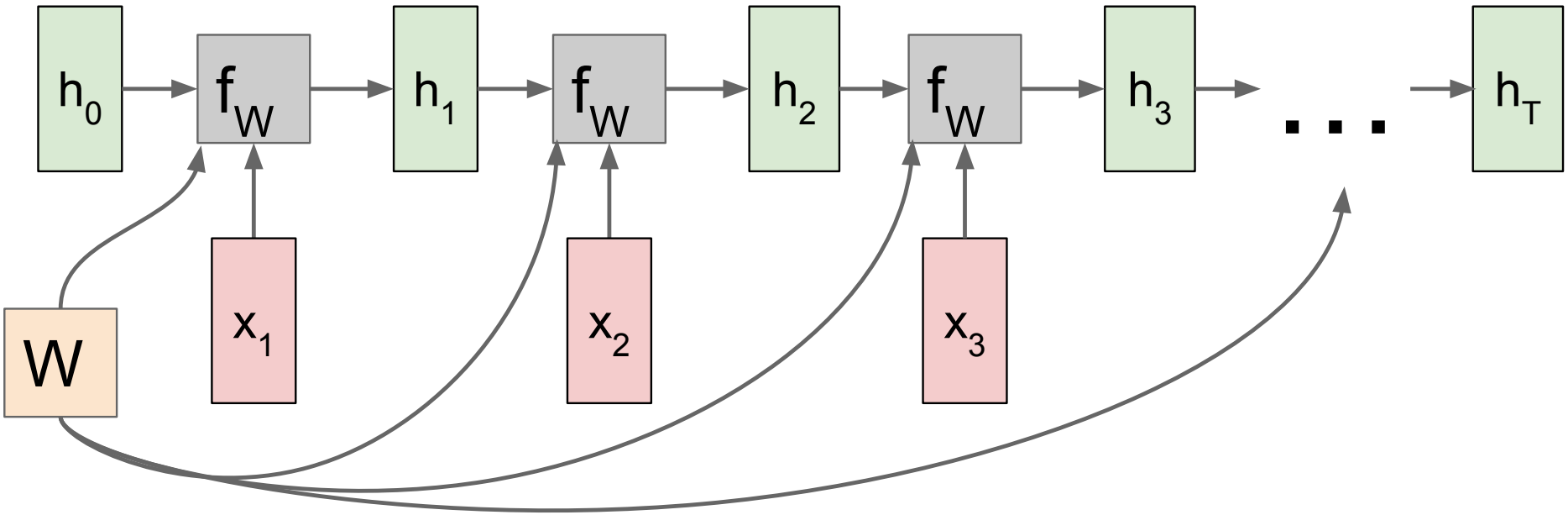
Unrolling RNN: Computational Graph

T time steps:



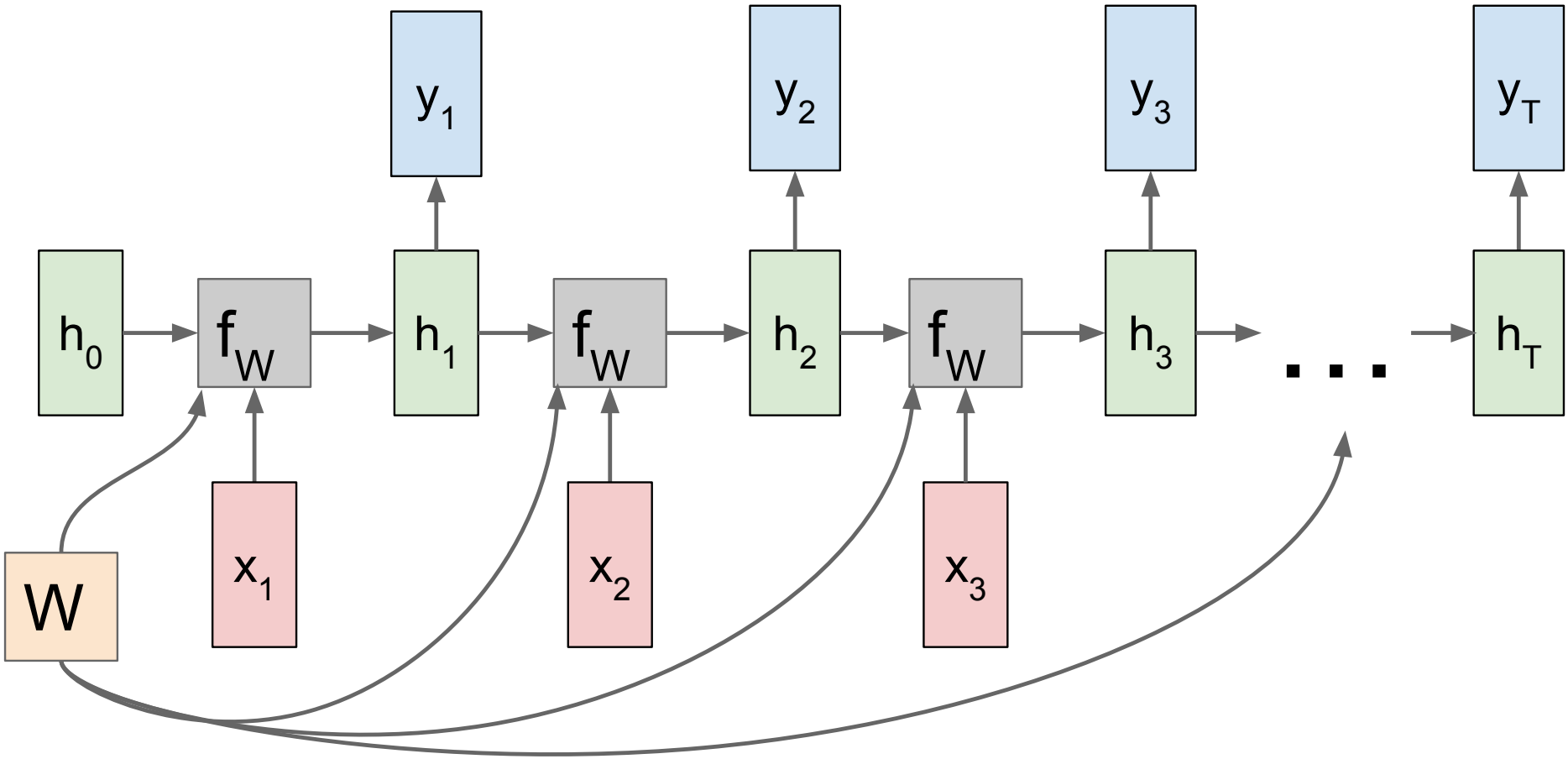
Unrolling RNN: Computational Graph

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



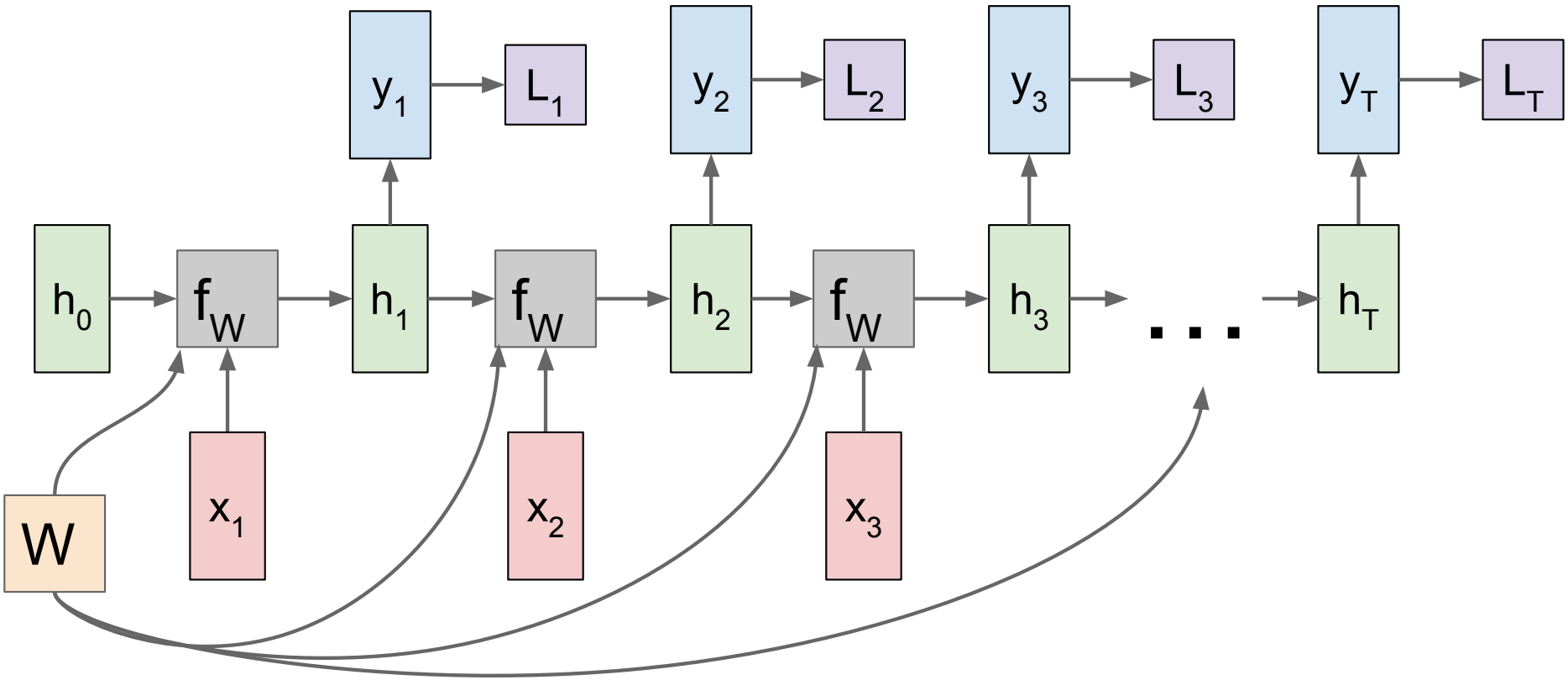
RNN: Computational Graph: Many to Many

Add outputs:



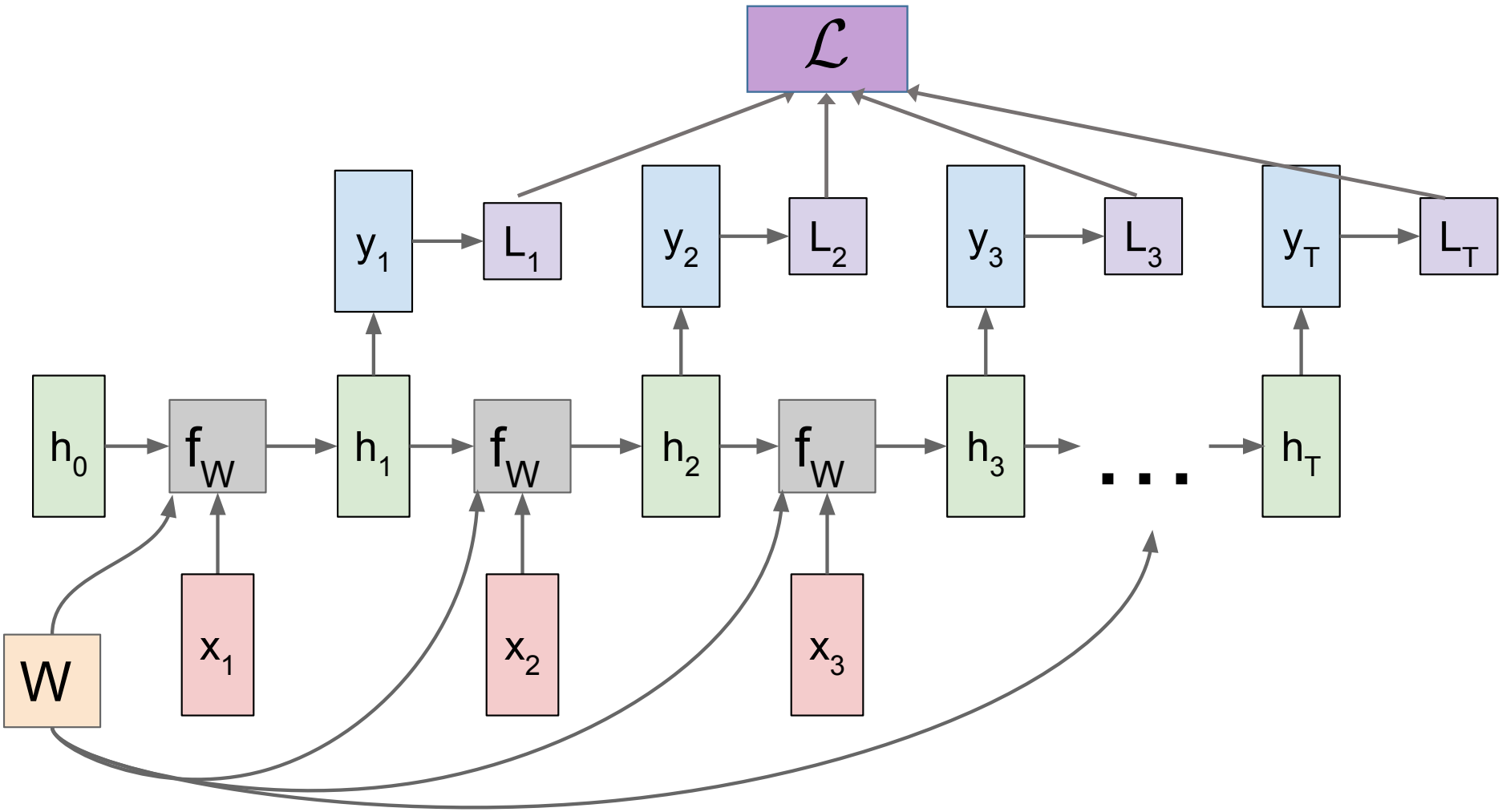
RNN: Computational Graph: Many to Many

Add loss for different time steps:



RNN: Computational Graph: Many to Many

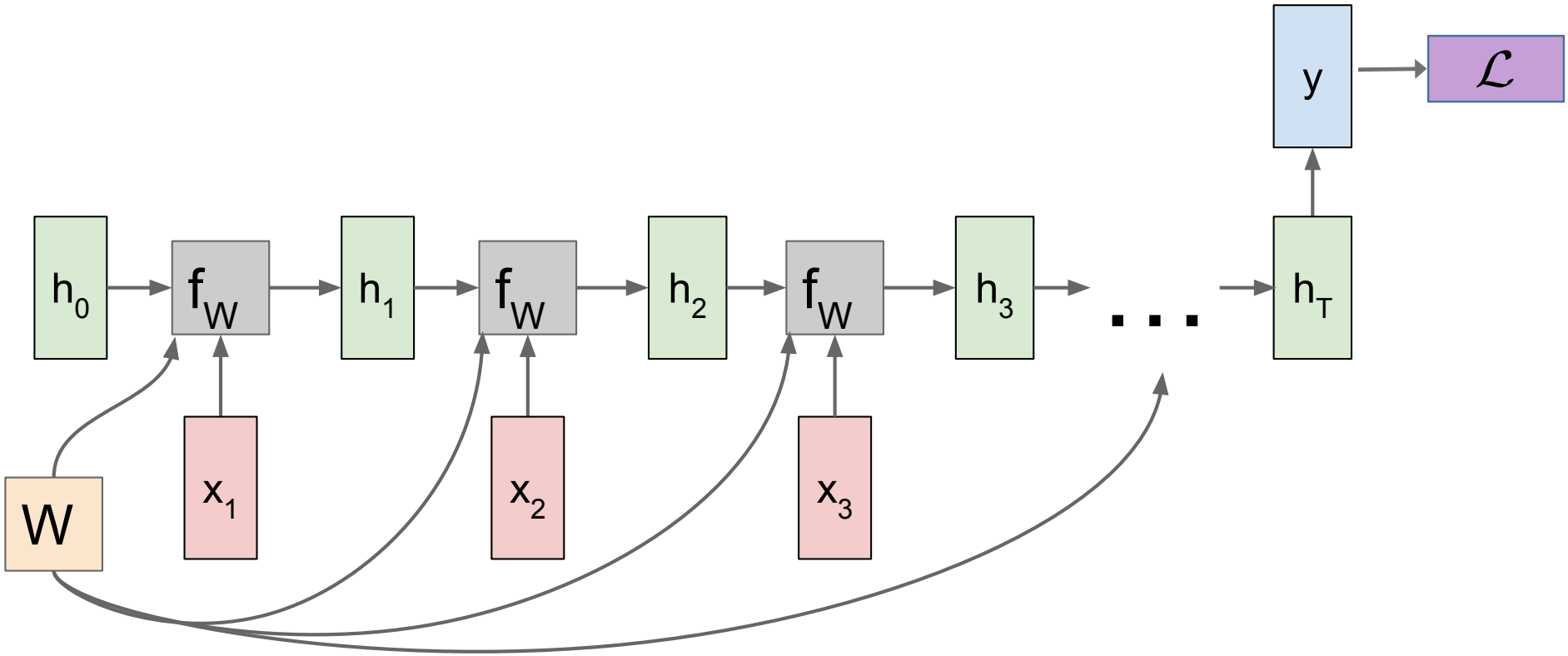
Total loss:



RNN: Computational Graph: Many to One

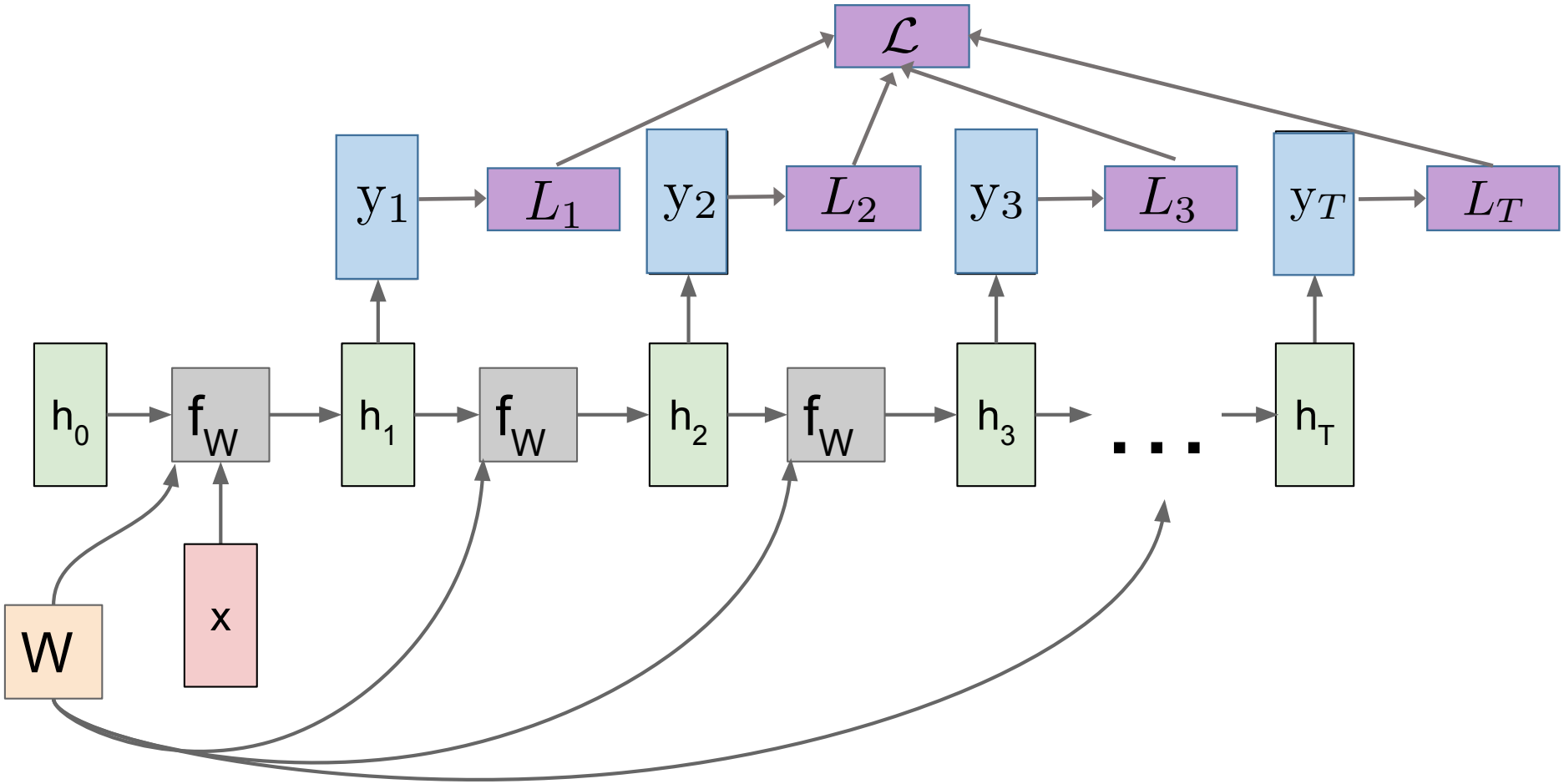
RNN: Computational Graph: Many to One

One output:



RNN: Computational Graph: One to Many

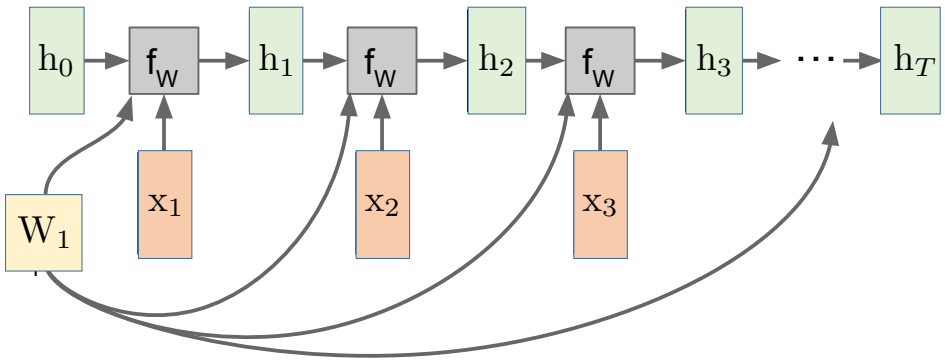
T outputs:



Sequence to Sequence: Many-to-one + one-to-many

- Encoder-decoder architecture, used for machine translation.

Many to one: Encode input sequence in a single vector

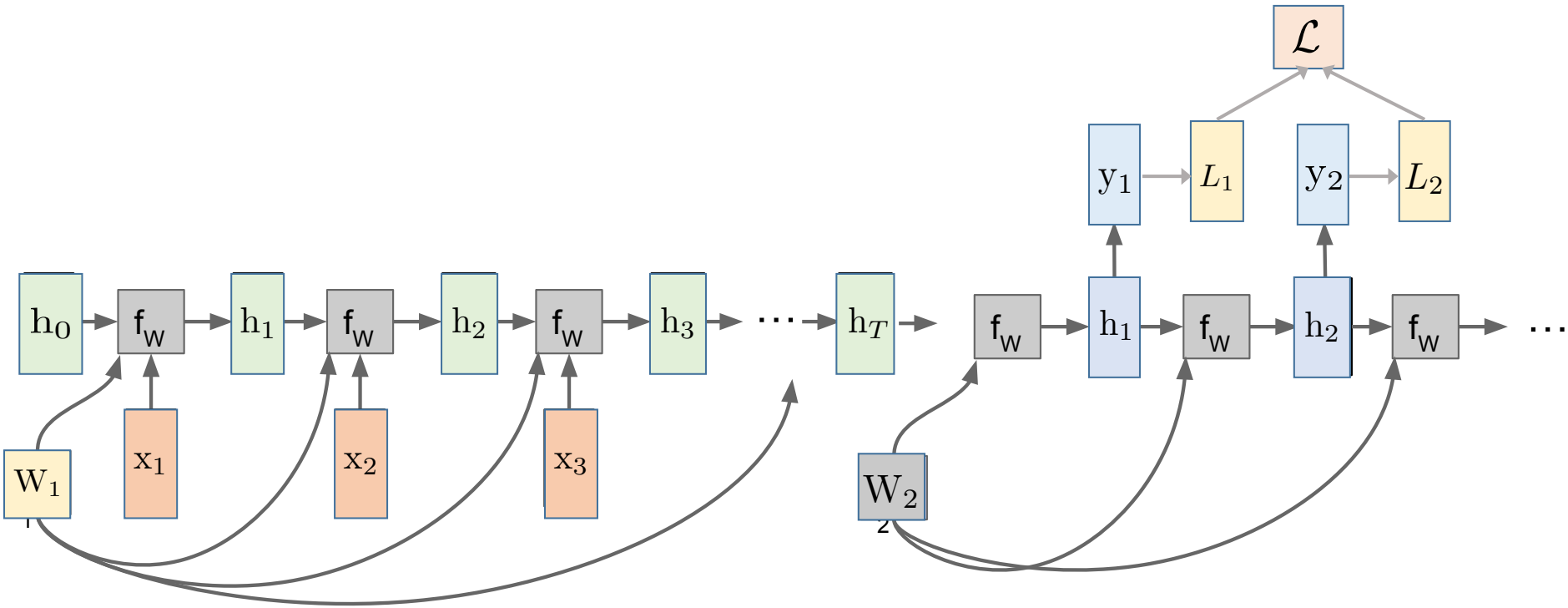


Sequence to Sequence: Many-to-one + one-to-many

- Encoder-decoder architecture, used for machine translation.

Many to one: Encode input sequence in a single vector

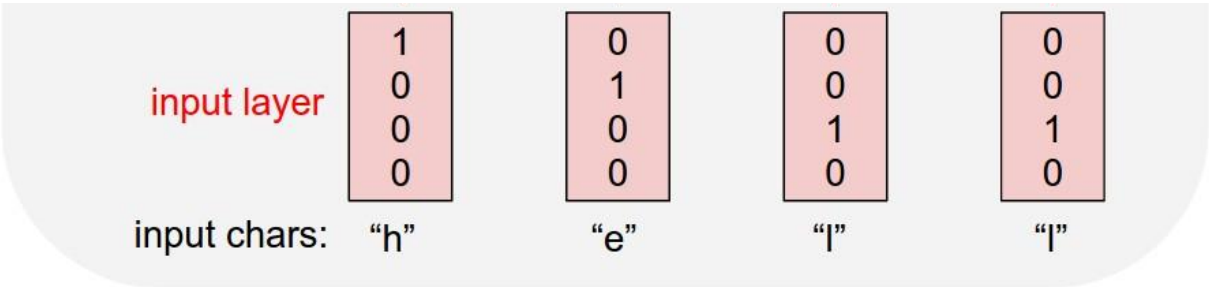
One to Many: Produce output sequence from single input vector



Example: Character-level Language Model

Many to many RNN model

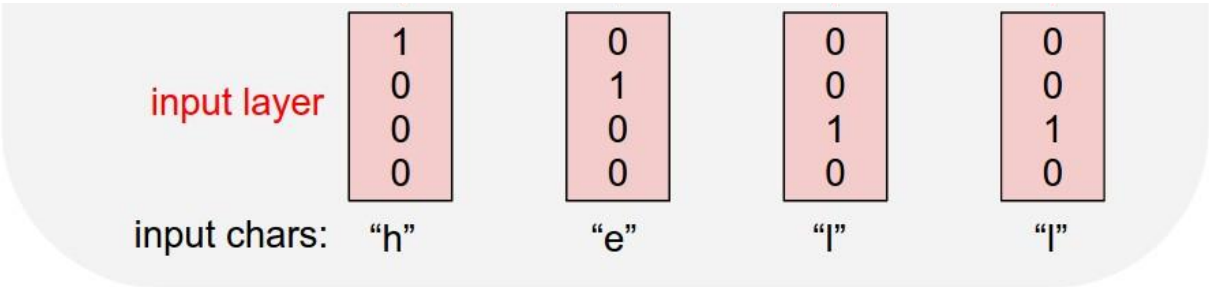
- Vocabulary:
  [h, e, l, o]
- Example training sequence:
  “hello”
  On training:



Example: Character-level Language Model

Many to many RNN model

- Vocabulary:
  [h, e, l, o]
- Example training sequence:
  “hello”
  On training:



Typically use word-embedding vector in practice.

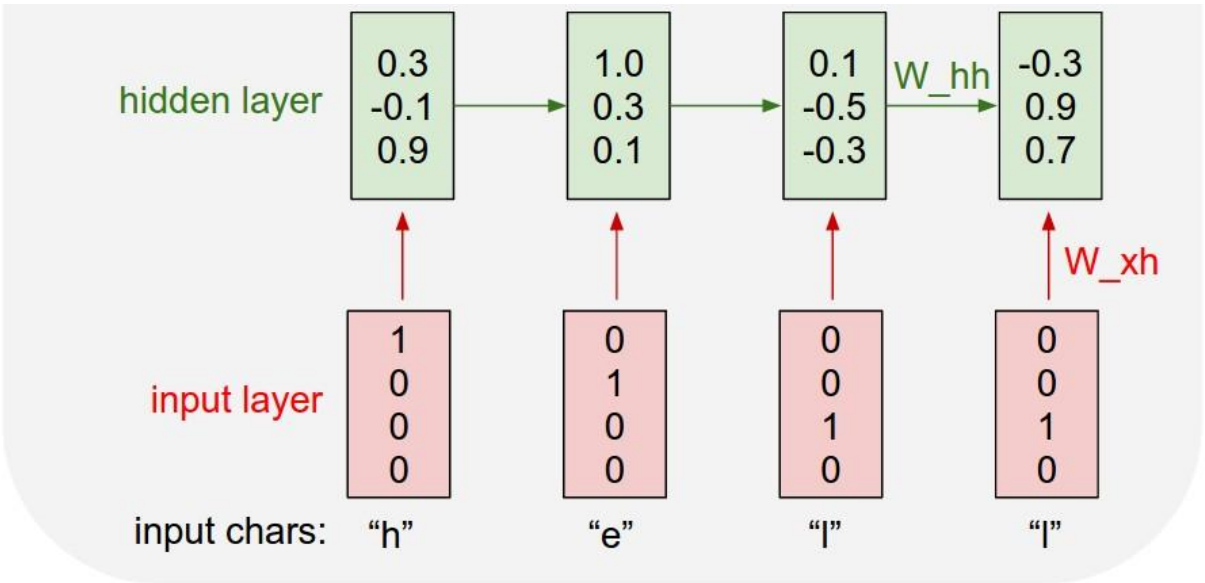
Example: Character-level Language Model

Many to many RNN model

$$\mathbf{h}_t = \tanh\left(\mathbf{W}_{hh}\,\mathbf{h}_{t-1} + \mathbf{W}_{xh}\,\mathbf{x}_t\right)$$

- Vocabulary: [h, e, l, o]
- Example training sequence: “hello”

On training:

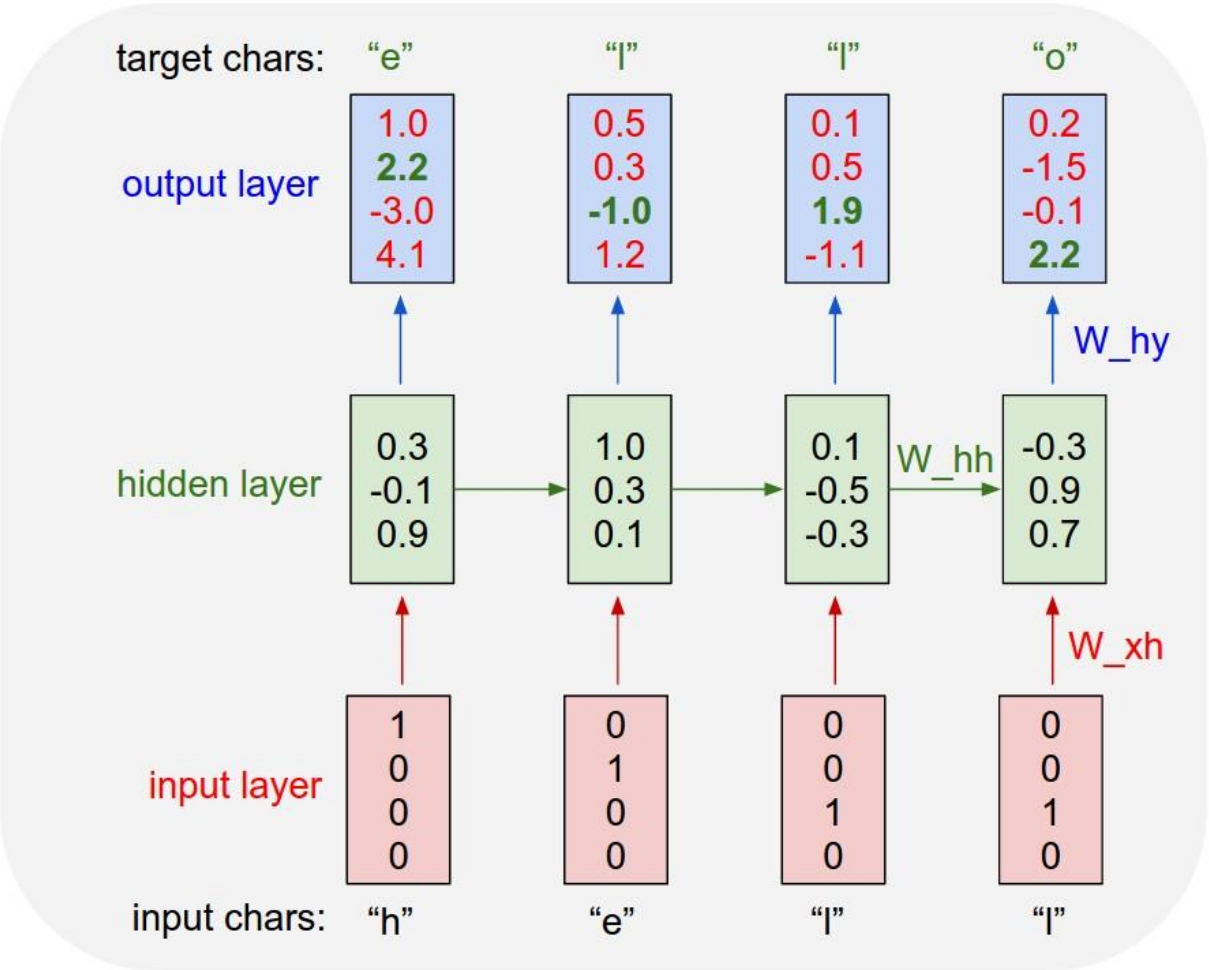


Example: Character-level Language Model

Many to many RNN model

- Vocabulary: [h, e, l, o]
- Example training sequence: “hello”

On training:

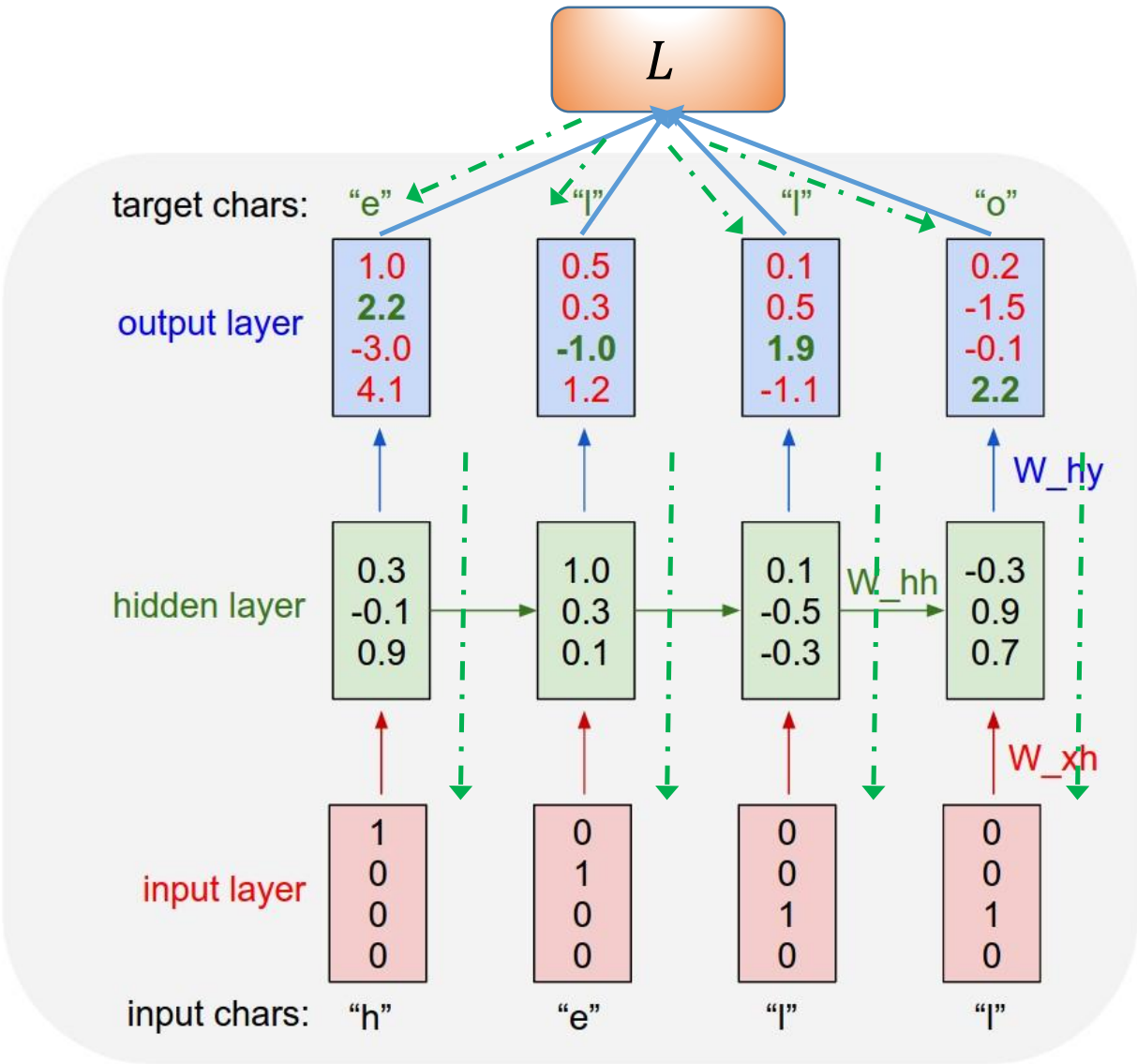


Example: Character-level Language Model

Many to many RNN model

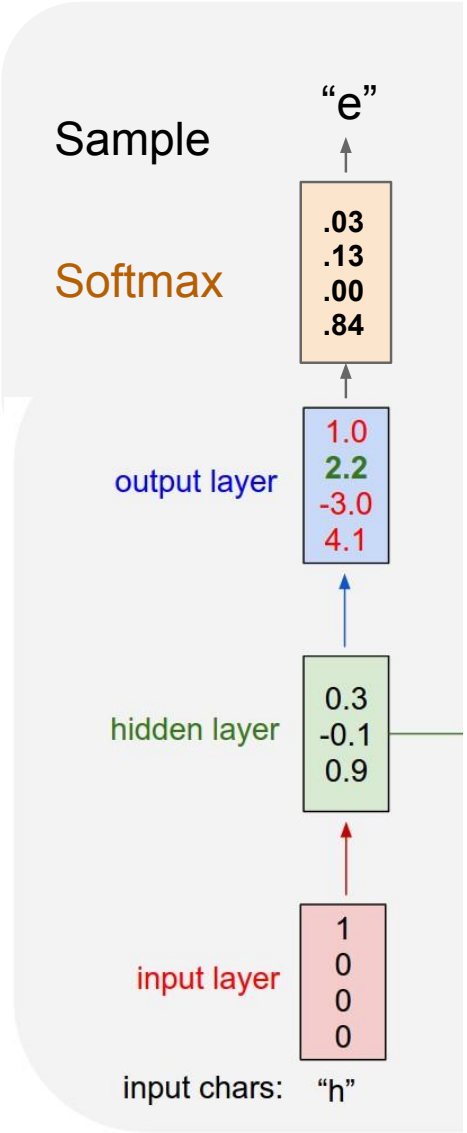
- Vocabulary: [h, e, l, o]
- Example training sequence: “hello”

  On training:



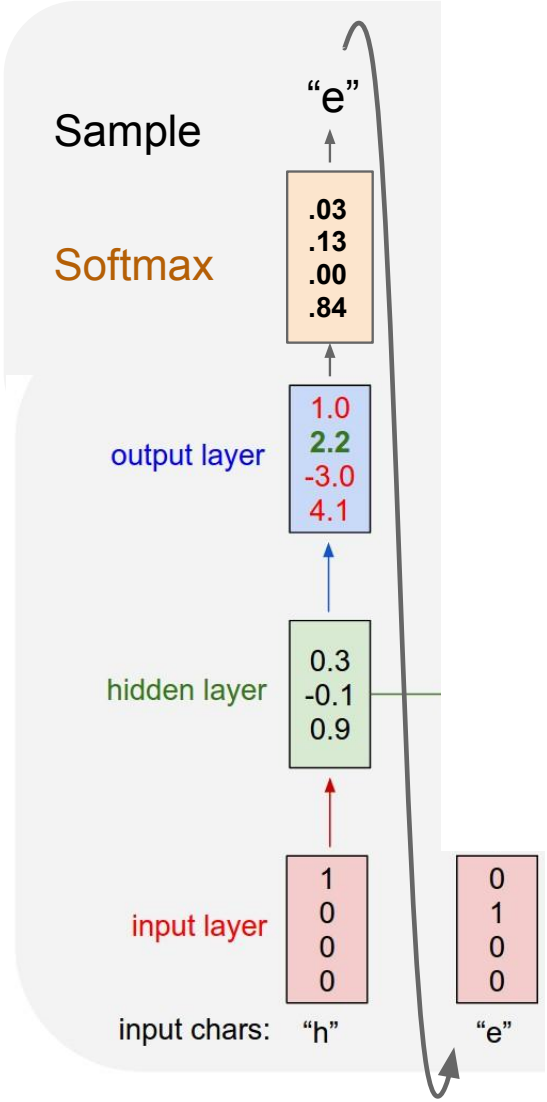
Example: Character-level Language Model

- Vocabulary: [h, e, l, o]
- At test-time sample characters one at a time, feed back to model



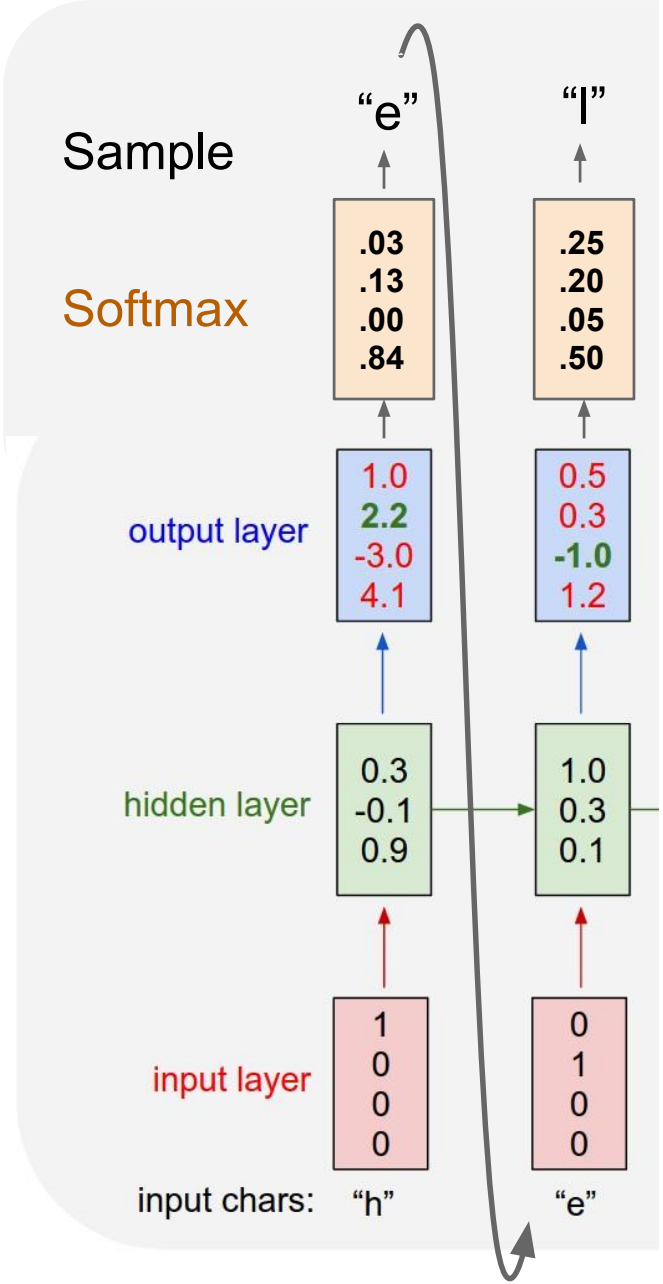
Example: Character-level Language Model

- Vocabulary: [h, e, l, o]
- At test-time sample characters one at a time, feed back to model



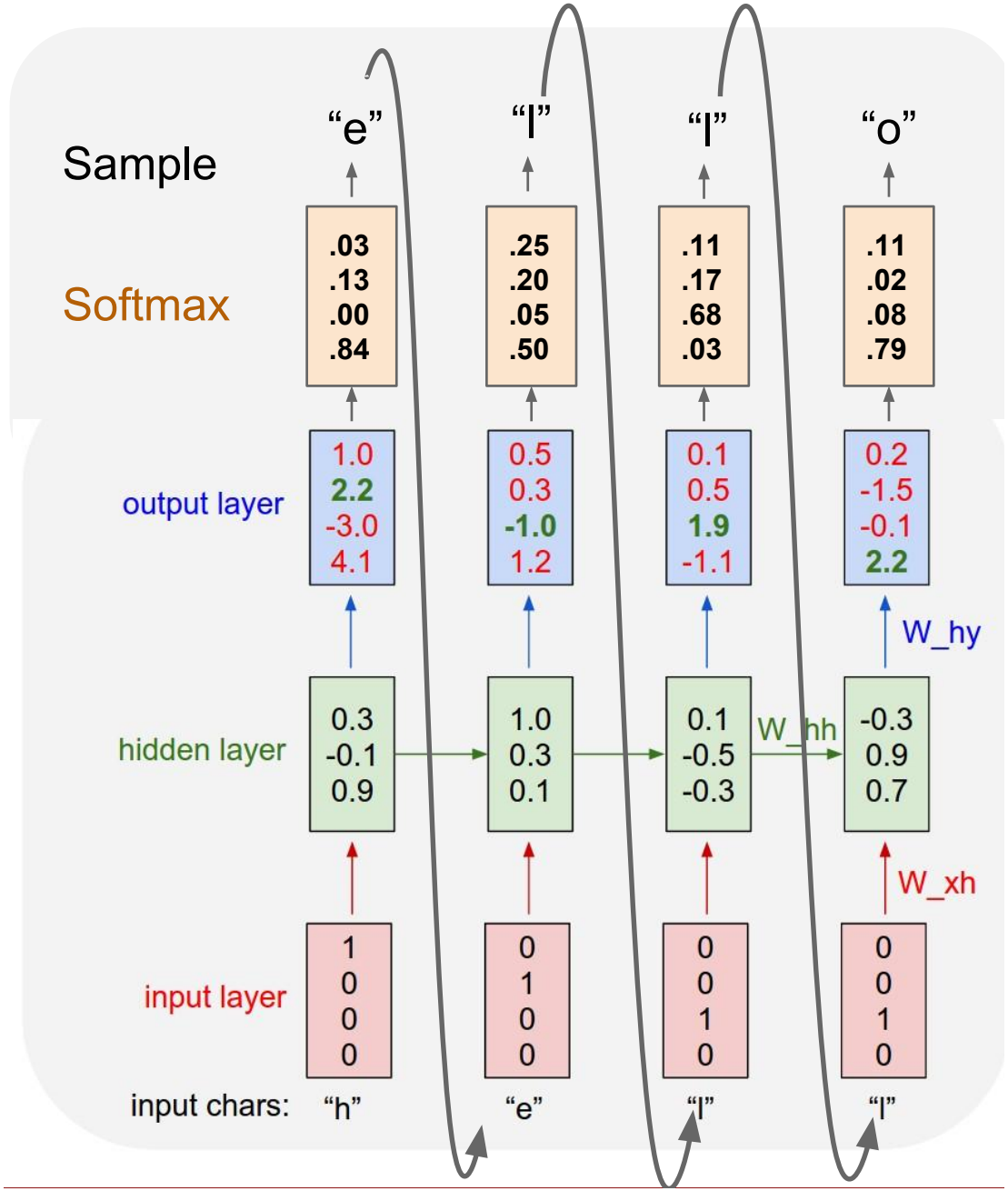
Example: Character-level Language Model

- Vocabulary: [h, e, l, o]
- At test-time sample characters one at a time, feed back to model



Example: Character-level Language Model

- Vocabulary: [h, e, l, o]
- At test-time sample characters one at a time, feed back to model



Character-level Language Model: Trained on Shakespeare's "The Sonnets"

random generate

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.
```

Character-level Language Model: Trained on Shakespeare's "The Sonnets"

random generate

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.
```

Q: how can we generate *space* and *comma*?

Character-level Language Model: Trained on Shakespeare's "The Sonnets"

random generate

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.
```

Q: How can we generate *space* and *comma*?

A: Consider them as auxiliary characters in the vocabulary.