



Sequence-to-sequence (seq2seq) Models + Structured Prediction as Search

Matt Gormley
Lecture 3
Sep. 7, 2022

Reminders

- **Homework 1: Neural Networks for Sequence Tagging**
 - **Out: Wed, Sep 7 (later today!)**
 - **Due: Fri, Sep 16 at 11:59pm**
 - Two parts:
 1. written part to Gradescope (Written slot)
 2. programming part to Gradescope (Programming slot)

Q&A

Q&A

Q: Can you show us a module-based AD example in PyTorch?

A: Sure thing! (next slide)

PyTorch

The same simple neural network we defined in pseudocode can also be defined in PyTorch.

```
1 # Define model
2 class NeuralNetwork(nn.Module):
3     def __init__(self):
4         super(NeuralNetwork, self).__init__()
5         self.flatten = nn.Flatten()
6         self.linear1 = nn.Linear(28*28, 512)
7         self.sigmoid = nn.Sigmoid()
8         self.linear2 = nn.Linear(512, 512)
9
10    def forward(self, x):
11        x = self.flatten(x)
12        a = self.linear1(x)
13        z = self.sigmoid(a)
14        b = self.linear2(z)
15        return b
16
17 # Take one step of SGD
18 def one_step_of_sgd(X, y):
19     loss_fn = nn.CrossEntropyLoss()
20     optimizer = torch.optim.SGD(model.parameters(), lr=1e-3)
21
22     # Compute prediction error
23     pred = model(X)
24     loss = loss_fn(pred, y)
25
26     # Backpropagation
27     optimizer.zero_grad()
28     loss.backward()
29     optimizer.step()
```

Q&A

Q: Can you show us a module-based AD example in PyTorch?

A: Sure thing! (next slide)

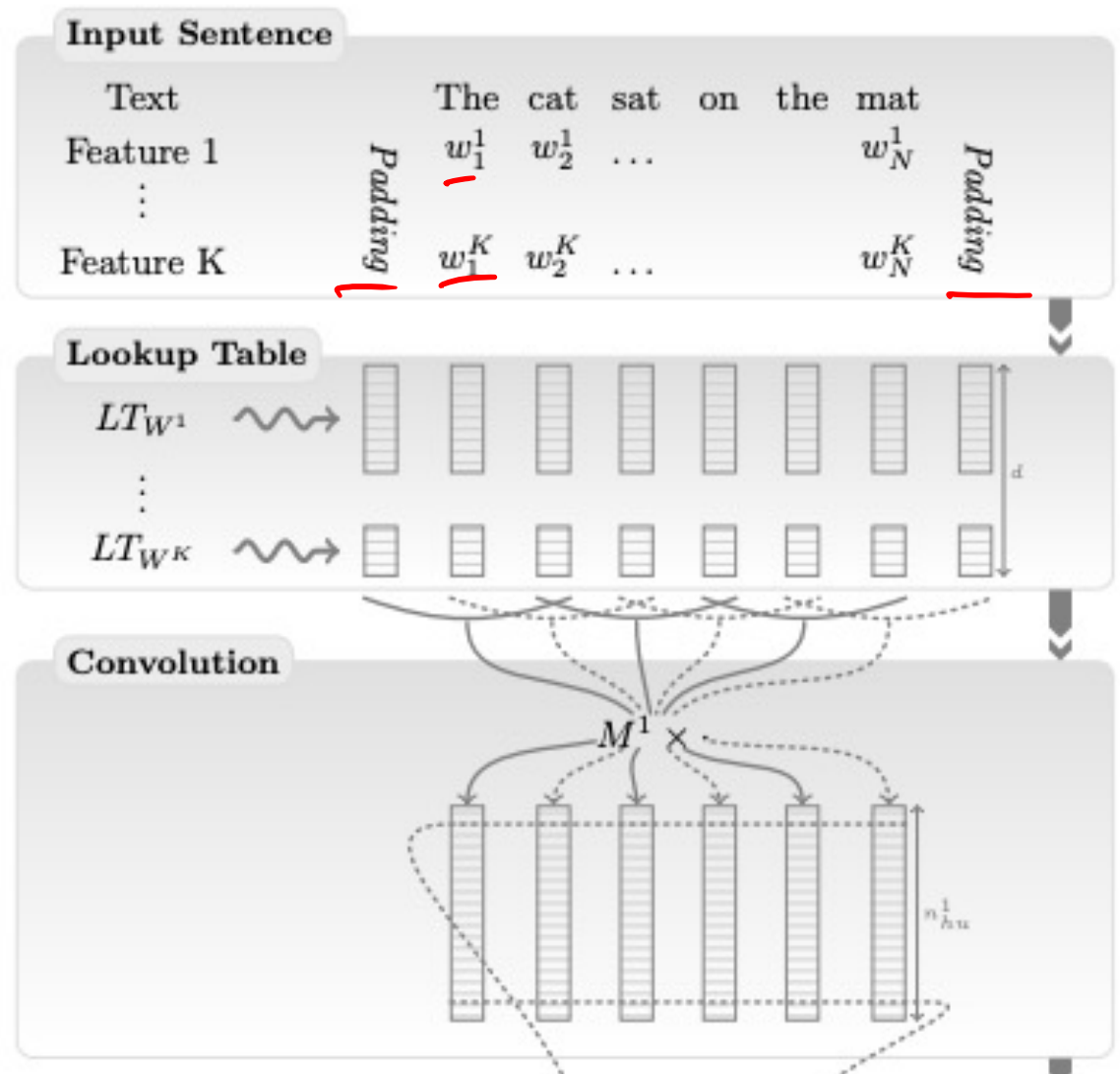
Q: That's pretty clean. Can you show us some more examples of PyTorch?

A: Sure thing! Come to recitation on Friday.

1D CONVOLUTION

1D Convolutional Neural Network

- Popularized for NLP by Collobert & Weston (2008)
- Two modules
 - **Embedding module:** each word type is mapped to a vector of parameters
 - **Convolution module:** each window of k embedding vectors is concatenated and then multiplied by a parameter matrix, to produce one vector per word token
+ non linearity



1D Convolutional Neural Network



Embedding Module

More formally, for each word $w \in \mathcal{D}$, an internal d_{word} -dimensional feature vector representation is given by the *lookup table* layer $LT_W(\cdot)$:

$$LT_W(w) = \langle W \rangle_w^1,$$

where $W \in \mathbb{R}^{d_{\text{word}} \times |\mathcal{D}|}$ is a matrix of parameters to be learned, $\langle W \rangle_w^1 \in \mathbb{R}^{d_{\text{word}}}$ is the w^{th} column of W and d_{word} is the word vector size (a hyper-parameter to be chosen by the user). Given a sentence or any sequence of T words $[w]_1^T$ in $\mathcal{D}$, the lookup table layer applies the same operation for each word in the sequence, producing the following output matrix:

$$LT_W([w]_1^T) = \begin{pmatrix} \langle W \rangle_{[w]_1}^1 & \langle W \rangle_{[w]_2}^1 & \cdots & \langle W \rangle_{[w]_T}^1 \end{pmatrix}. \quad (1)$$

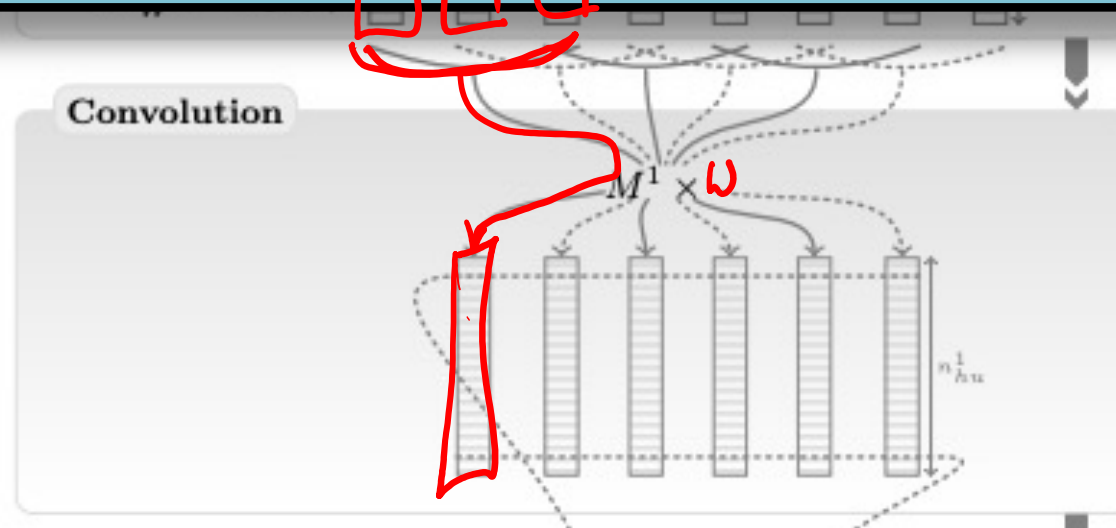
This matrix can then be fed to further neural network layers, as we will see below.

1D Convolutional Neural Network

1D Convolution Module

A window approach assumes the tag of a word depends mainly on its neighboring words. Given a word to tag, we consider a fixed size k_{sz} (a hyper-parameter) window of words around this word. Each word in the window is first passed through the lookup table layer (1) or (2), producing a matrix of word features of fixed size $d_{wrd} \times k_{sz}$. This matrix can be viewed as a $d_{wrd} k_{sz}$ -dimensional vector by concatenating each column vector, which can be fed to further neural network layers. More formally, the word feature window given by the first network layer can be written as:

$$f_{\theta}^1 = \langle LT_W([w]_1^T) \rangle_t^{d_{win}} = \begin{pmatrix} \langle W \rangle_{[w]_t}^1 \\ \vdots \\ \langle W \rangle_{[w]_t}^1 \\ \vdots \\ \langle W \rangle_{[w]_{t+d_{win}/2}}^1 \end{pmatrix} \cdot d_{win} \quad (3)$$



1D Convolutional Neural Network

1D Convolution Module

A window approach assumes the tag of a word depends mainly on its neighboring words. Given a word to tag, we consider a fixed size k_{sz} (a hyper-parameter) window of words around this word. Each word in the window is first passed through the lookup table layer (1) or (2), producing a matrix of word features of fixed size $d_{wrd} \times k_{sz}$. This matrix can be viewed as a $d_{wrd} k_{sz}$ -dimensional vector by concatenating each column vector, which can be fed to further neural network layers. More formally, the word feature window given by the first network layer can be written as:

$$f_{\theta}^1 = \langle LT_W([w]_1^T) \rangle_t^{d_{win}} = \begin{pmatrix} \langle W \rangle_{[w]_{t-d_{win}/2}}^1 \\ \vdots \\ \langle W \rangle_{[w]_t}^1 \\ \vdots \\ \langle W \rangle_{[w]_{t+d_{win}/2}}^1 \end{pmatrix}. \quad (3)$$

Linear Layer. The fixed size vector f_{θ}^1 can be fed to one or several standard neural network layers which perform affine transformations over their inputs:

$$f_{\theta}^l = W^l f_{\theta}^{l-1} + b^l, \quad (4)$$

where $W^l \in \mathbb{R}^{n_{hu}^l \times n_{hu}^{l-1}}$ and $b^l \in \mathbb{R}^{n_{hu}^l}$ are the parameters to be *trained*. The hyper-parameter n_{hu}^l is usually called the *number of hidden units* of the l^{th} layer.

→ non linear

1D Convolutional Neural Network

1D Convolution Module

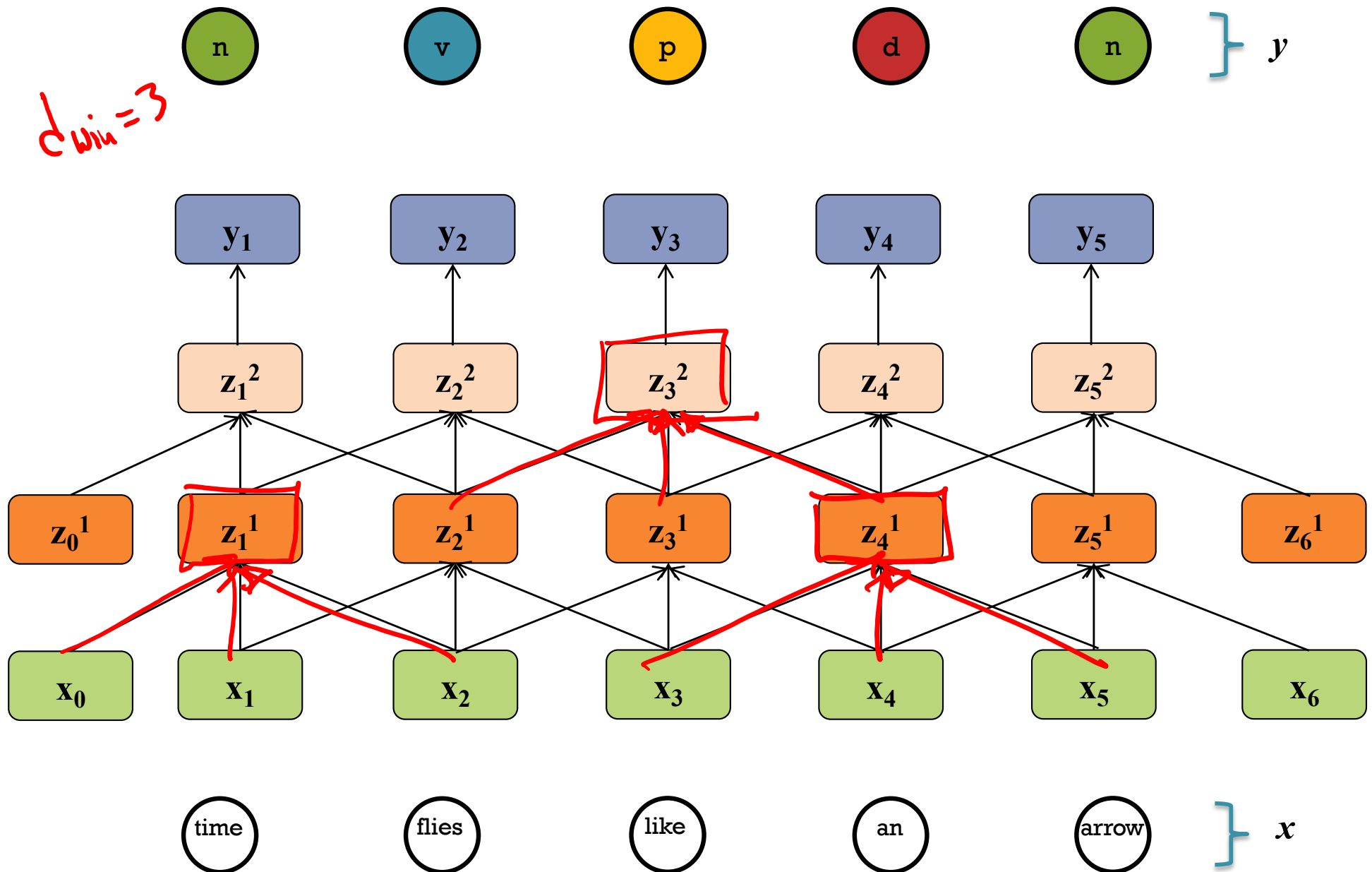
Convolutional Layer. A convolutional layer can be seen as a generalization of a window approach: given a sequence represented by columns in a matrix f_{θ}^{l-1} (in our lookup table matrix (1)), a matrix-vector operation as in (4) is applied to each window of successive windows in the sequence.

Using previous notations, the t^{th} output column of the l^{th} layer can be computed as:

$$\langle f_{\theta}^l \rangle_t^1 = W^l \langle f_{\theta}^{l-1} \rangle_t^{d_{win}} + b^l \quad \forall t, \quad (6)$$

where the weight matrix W^l is the same across all windows t in the sequence. Convolutional layers extract local features around each window of the given sequence. As for standard affine layers (4), convolutional layers are often stacked to extract higher level features. In this case, each layer must be followed by a non-linearity (5) or the network would be equivalent to one convolutional layer.

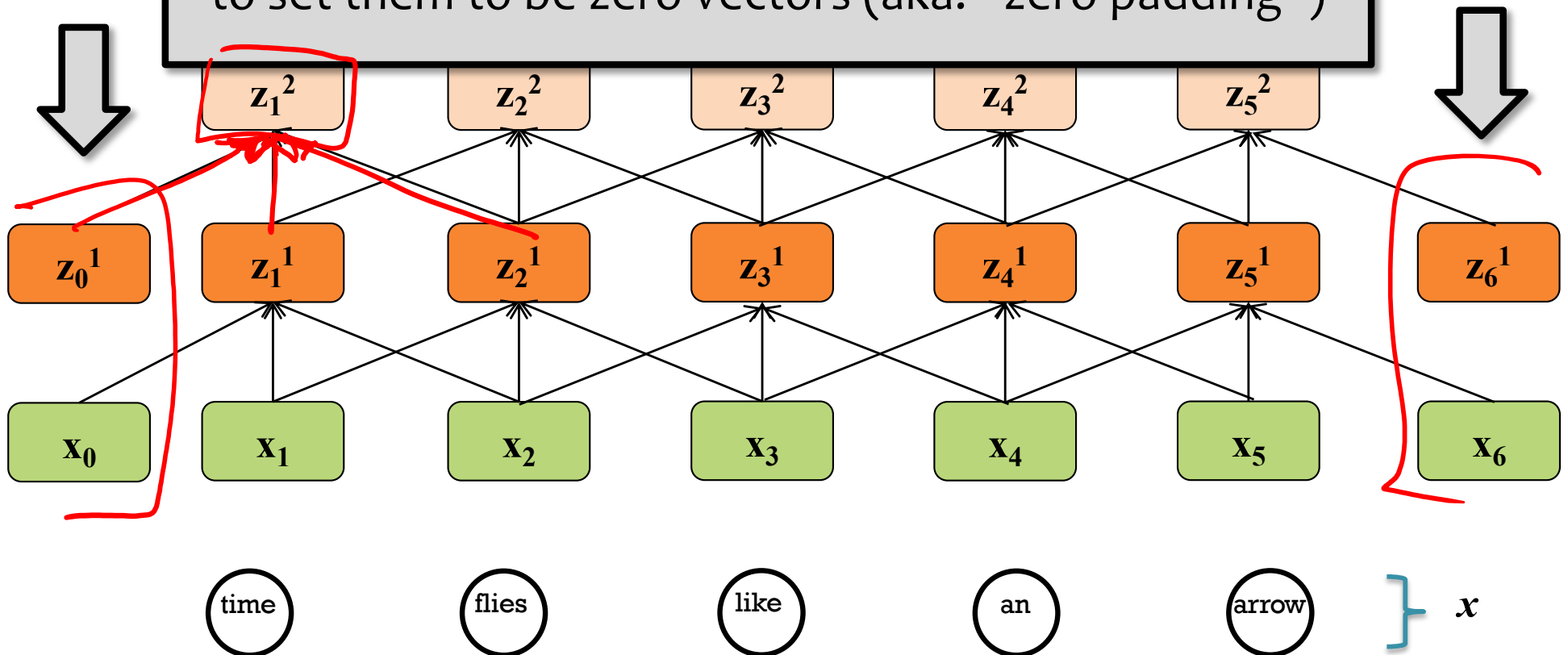
1D Convolutional Neural Network



1D Convolutional Neural Network

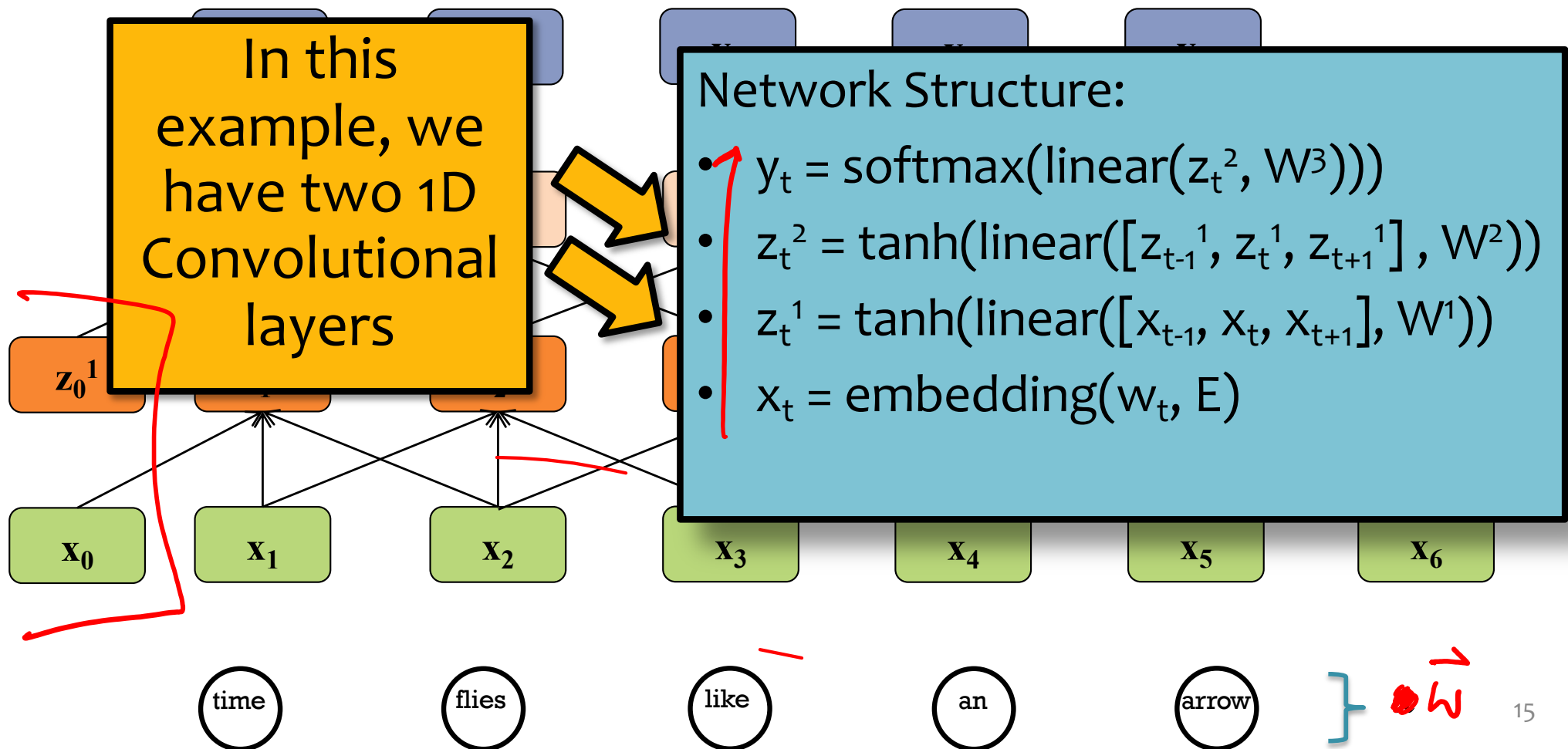
n v p d n } y

These extra vectors are called “padding”. It’s common to set them to be zero vectors (aka. “zero padding”)



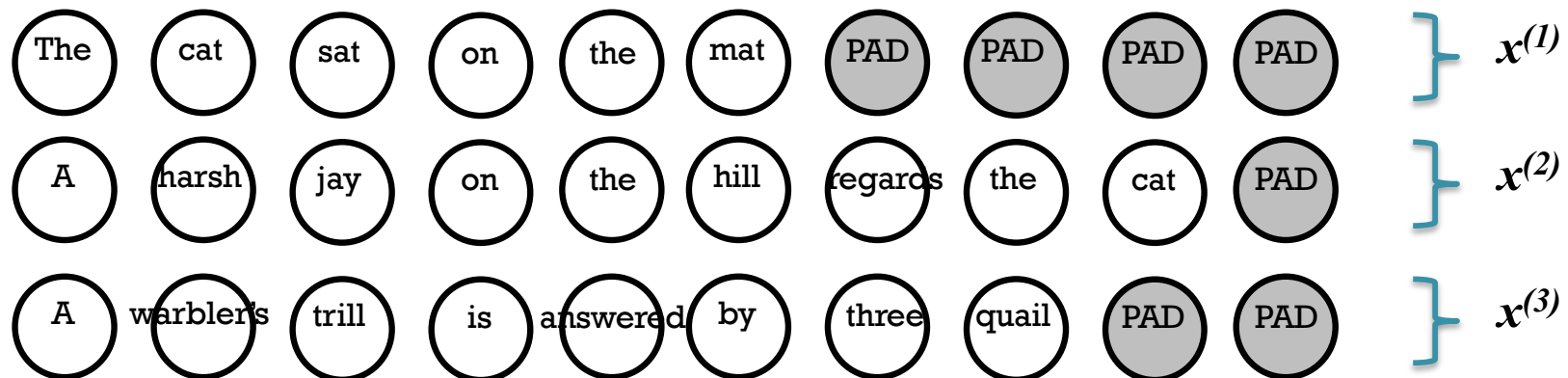
1D Convolutional Neural Network

n v p d n } y



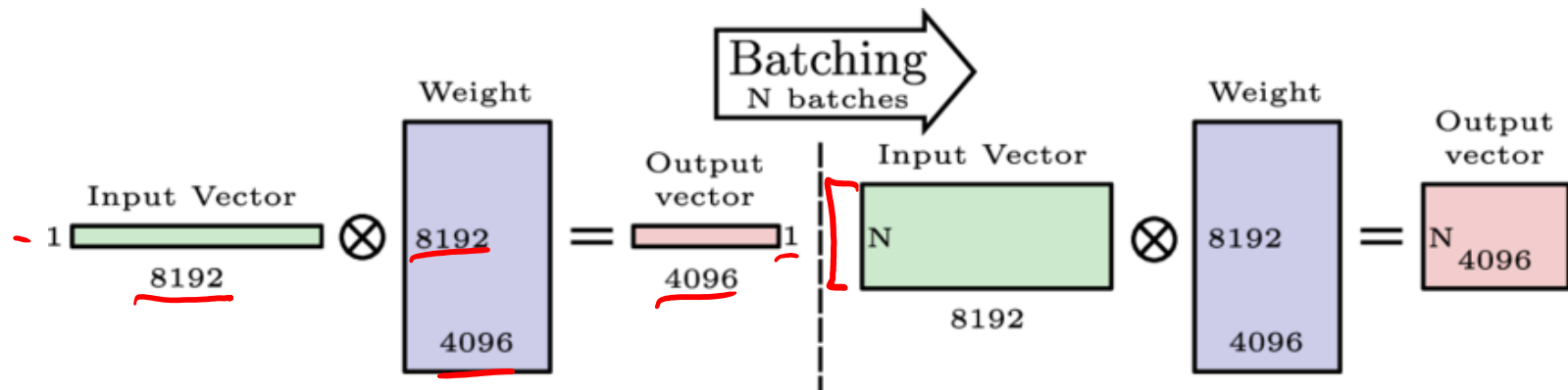
Efficiency Tricks

- Padding:
 - When working with sentences of different lengths, it's common to work with a fixed maximum length L
 - For a sentence of length $T < L$, we append $(L - T)$ zero vectors after the sentence (i.e. pad the sentence)



Efficiency Tricks

- Batching:
 - The most computationally intensive modules involve matrix-vector arithmetic
 - These computations can be made more efficient by squashing a collection of input vectors together into an input matrix

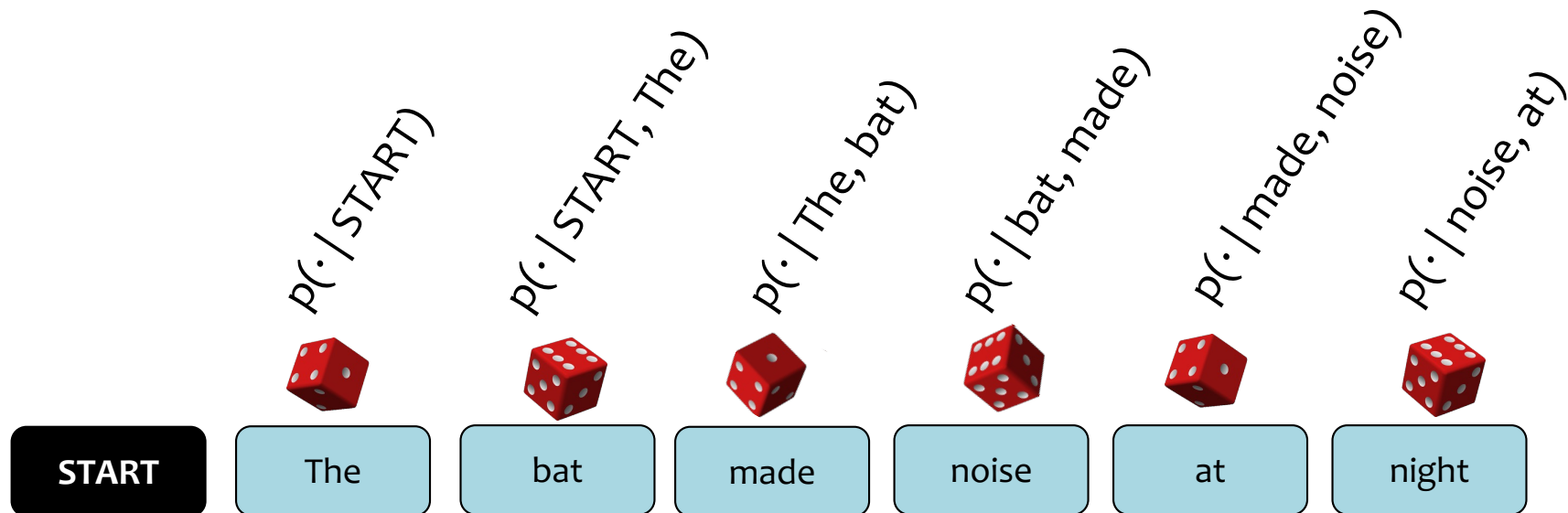


- Batching Sentences:
 - If our input are sentences, then we can group together a collection of sentences (all of length L, thanks to padding)
 - That group is called a “batch” and passed through each layer as a unit
 - Each layer in module-based AD can then be implemented to support this batched input efficiently

BACKGROUND: N-GRAM LANGUAGE MODELS

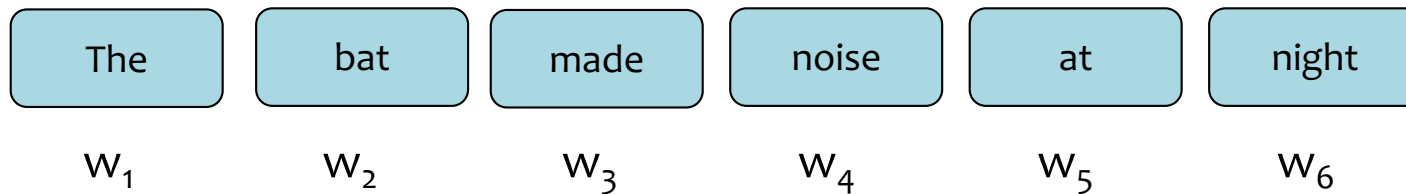
n-Gram Language Model

- Goal: Generate realistic looking sentences in a human language
- Key Idea: condition on the last $n-1$ words to sample the n^{th} word



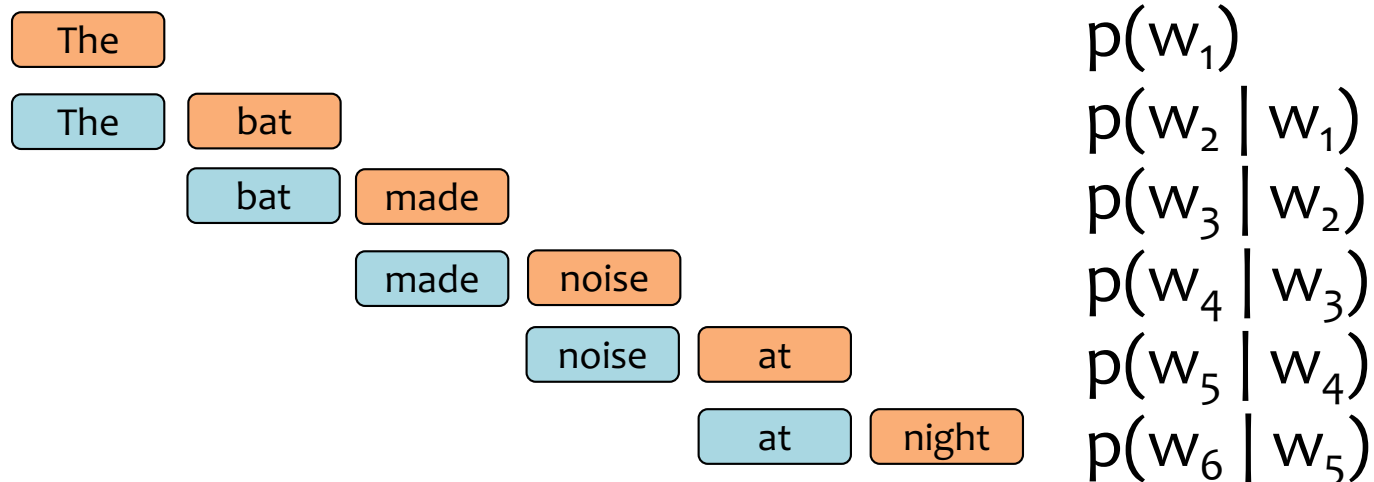
n-Gram Language Model

Question: How can we **define** a probability distribution over a sequence of length T?



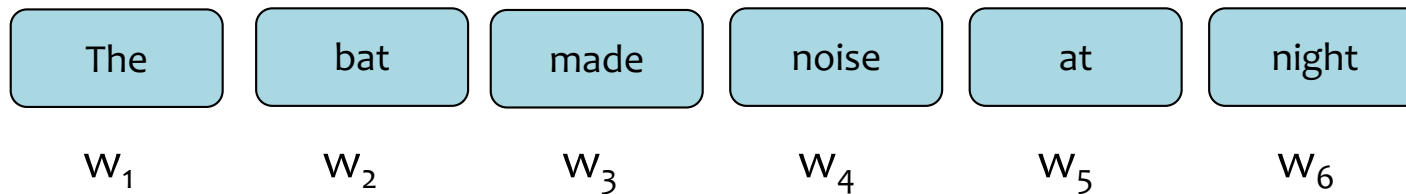
$$\text{n-Gram Model (n=2)} \quad p(w_1, w_2, \dots, w_T) = \prod_{t=1}^T p(w_t \mid w_{t-1})$$

$$p(w_1, w_2, w_3, \dots, w_6) =$$



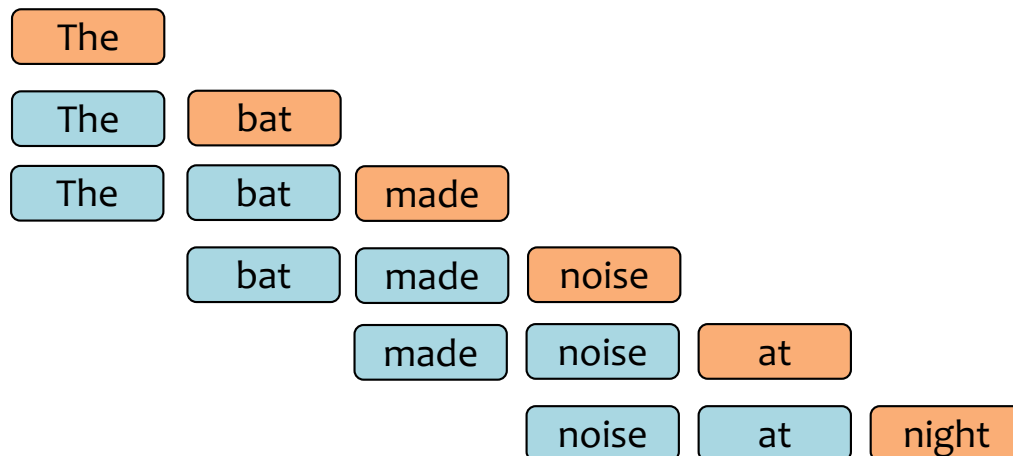
n-Gram Language Model

Question: How can we **define** a probability distribution over a sequence of length T?



n-Gram Model (n=3)
$$p(w_1, w_2, \dots, w_T) = \prod_{t=1}^T p(w_t \mid w_{t-1}, w_{t-2})$$

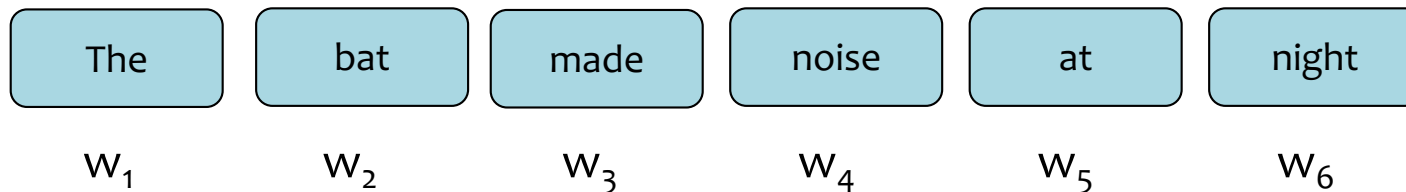
$$p(w_1, w_2, w_3, \dots, w_6) =$$



$$\begin{aligned} & p(w_1 \mid \text{START}, \text{START}) \\ & p(w_2 \mid w_1, \text{START}) \\ & p(w_3 \mid w_2, w_1) \\ & p(w_4 \mid w_3, w_2) \\ & p(w_5 \mid w_4, w_3) \\ & p(w_6 \mid w_5, w_4) \end{aligned}$$

n-Gram Language Model

Question: How can we **define** a probability distribution over a sequence of length T?



n-Gram Model (n=3)

$$p(w_1, w_2, \dots, w_T) = \prod_{t=1}^T p(w_t \mid w_{t-1}, w_{t-2})$$

$p(w_1, w_2, w_3, \dots, w_6) =$

$p(w_1)$

$p(w_2 \mid w_1)$

Note: This is called a **model** because we made some **assumptions** about how many previous words to condition on (i.e. only n-1 words)

Learning an n-Gram Model

Question: How do we **learn** the probabilities for the n-Gram Model?

$p(w_t \mid w_{t-2} = \text{The}, w_{t-1} = \text{bat})$



w_t	$p(\cdot \mid \cdot, \cdot)$
ate	0.015

...

flies	0.046
-------	-------

...

zebra	0.000
-------	-------

$p(w_t \mid w_{t-2} = \text{made}, w_{t-1} = \text{noise})$



w_t	$p(\cdot \mid \cdot, \cdot)$
at	0.020

...

pollution	0.030
-----------	-------

...

zebra	0.000
-------	-------

$p(w_t \mid w_{t-2} = \text{cows}, w_{t-1} = \text{eat})$



w_t	$p(\cdot \mid \cdot, \cdot)$
corn	0.420

...

grass	0.510
-------	-------

...

zebra	0.000
-------	-------

Learning an n-Gram Model

Question: How do we **learn** the probabilities for the n-Gram Model?

Answer: From data! Just **count** n-gram frequencies

... the **cows** eat **grass**...
... our **cows** eat hay daily...
... factory-farm **cows** eat **corn**...
... on an organic farm, **cows** eat hay and...
... do your **cows** eat **grass** or corn?...
... what do **cows** eat if they have...
... **cows** eat **corn** when there is no...
... which **cows** eat which foods depends...
... if **cows** eat **grass**...
... when **cows** eat **corn** their stomachs...
... should we let **cows** eat **corn**?...

$$p(w_t \mid w_{t-2} = \text{cows}, w_{t-1} = \text{eat})$$

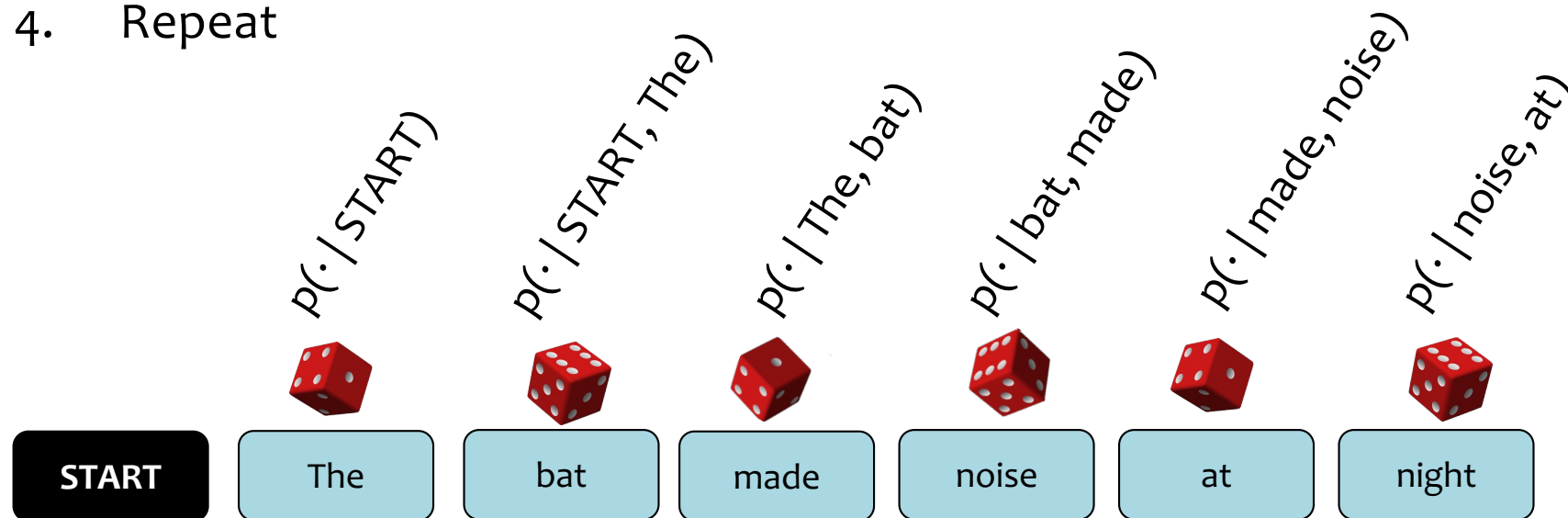

w_t	$p(\cdot \mid \cdot, \cdot)$
corn	4/11
grass	3/11
hay	2/11
if	1/11
which	1/11

Sampling from a Language Model

Question: How do we sample from a Language Model?

Answer:

1. Treat each probability distribution like a (50k-sided) weighted die
2. Pick the die corresponding to $p(w_t \mid w_{t-2}, w_{t-1})$
3. Roll that die and generate whichever word w_t lands face up
4. Repeat



Sampling from a Language Model

Question: How do we sample from a Language Model?

Answer:

1. Treat each probability distribution like a (50k-sided) weighted die
2. Pick the die corresponding to $p(w_t \mid w_{t-2}, w_{t-1})$
3. Roll that die and generate whichever word w_t lands face up
4. Repeat

Training Data (Shakespeare)

I tell you, friends, most charitable care
ave the patricians of you. For your
wants, Your suffering in this dearth,
you may as well Strike at the heaven
with your staves as lift them Against
the Roman state, whose course will on
The way it takes, cracking ten thousand
curbs Of more strong link asunder than
can ever Appear in your impediment.
For the dearth, The gods, not the
patricians, make it, and Your knees to
them, not arms, must help.

5-Gram Model

Approacheth, denay. duncy
Thither! Julius think: grant,--0
Yead linens, sheep's Ancient,
Agreed: Petrarch plaguy Resolved
pear! observingly honourest
adulteries wherever scabbard
guess; affirmation--his monsieur;
died. jealousy, chequins me.
Daphne building. weakness: sun-
rise, cannot stays carry't,
unpurposed. prophet-like drink;
back-return 'gainst surmise
Bridget ships? wane; interim?
She's striving wet;

RECURRENT NEURAL NETWORK (RNN) LANGUAGE MODELS

Recurrent Neural Networks (RNNs)

inputs: $\mathbf{x} = (x_1, x_2, \dots, x_T), x_i \in \mathcal{R}^I$

hidden units: $\mathbf{h} = (h_1, h_2, \dots, h_T), h_i \in \mathcal{R}^J$

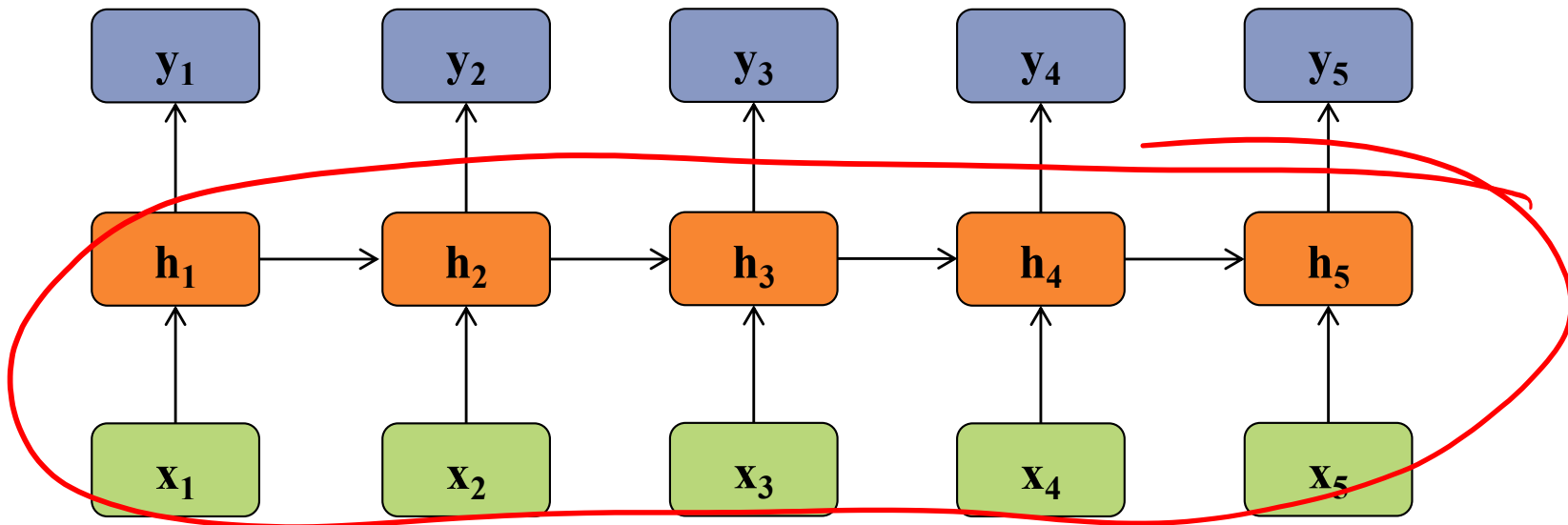
outputs: $\mathbf{y} = (y_1, y_2, \dots, y_T), y_i \in \mathcal{R}^K$

nonlinearity: $\mathcal{H}$

Definition of the RNN:

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

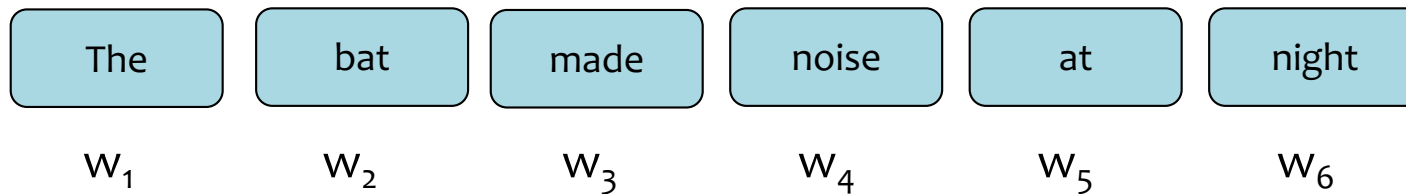
$$y_t = W_{hy}h_t + b_y$$



Recall...

The Chain Rule of Probability

Question: How can we **define** a probability distribution over a sequence of length T?



Chain rule of probability:
$$p(w_1, w_2, \dots, w_T) = \prod_{t=1}^T p(w_t \mid w_{t-1}, \dots, w_1)$$

$p(w_1, w_2, w_3, \dots, w_6) =$

$p(w_1)$

$p(w_2 \mid w_1)$

Note: This is called the chain **rule** because it is **always** true for every probability distribution

The

The

The

The

The

The

bat

made

noise

at

night

$p(w_6 \mid w_5, w_4, w_3, w_2, w_1)$

RNN Language Model

$$\text{RNN Language Model: } \underline{p(w_1, w_2, \dots, w_T)} = \prod_{t=1}^T \underline{p(w_t \mid f_{\theta}(w_{t-1}, \dots, w_1))}$$

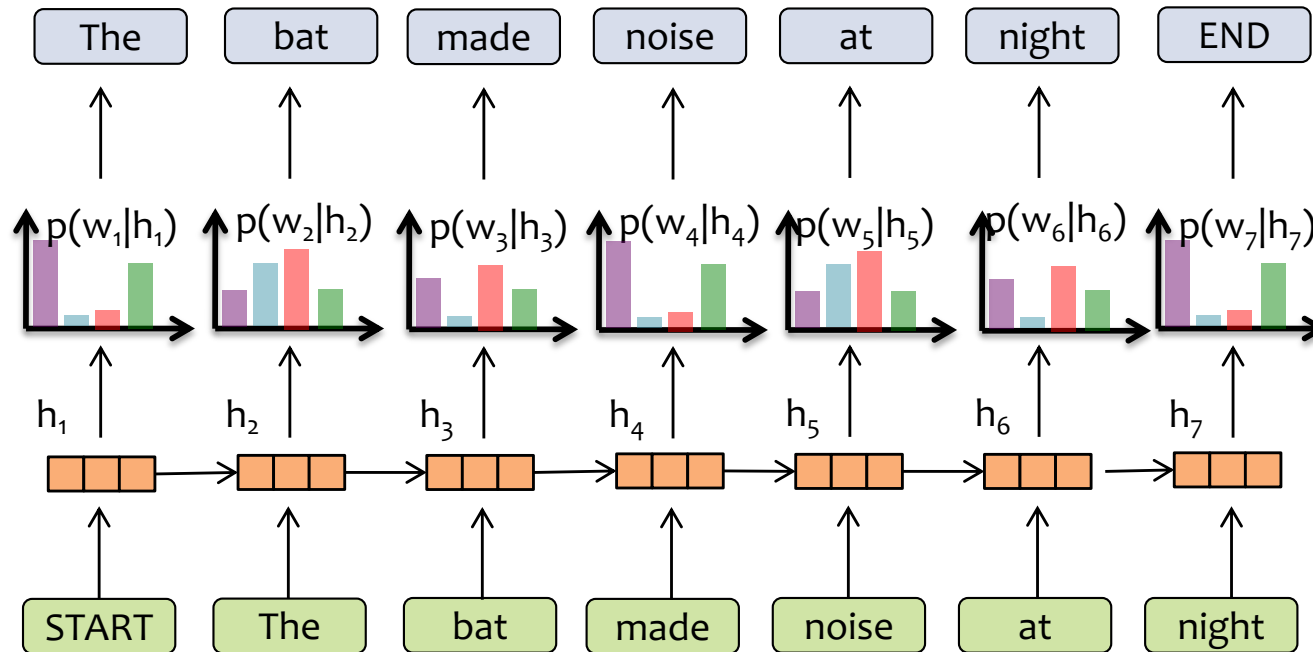
$$p(w_1, w_2, w_3, \dots, w_6) =$$

The						$p(w_1)$
The	bat					$p(w_2 \mid f_{\theta}(w_1))$
The	bat	made				$p(w_3 \mid f_{\theta}(w_2, w_1))$
The	bat	made	noise			$p(w_4 \mid f_{\theta}(w_3, w_2, w_1))$
The	bat	made	noise	at		$p(w_5 \mid f_{\theta}(w_4, w_3, w_2, w_1))$
The	bat	made	noise	at	night	$p(w_6 \mid f_{\theta}(w_5, w_4, w_3, w_2, w_1))$

Key Idea:

- (1) convert all previous words to a **fixed length vector**
- (2) define distribution $p(w_t \mid f_{\theta}(w_{t-1}, \dots, w_1))$ that conditions on the vector

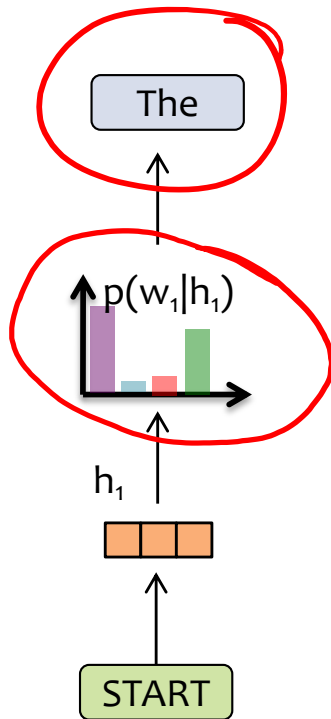
RNN Language Model



Key Idea:

- (1) convert all previous words to a **fixed length vector**
- (2) define distribution $p(w_t | f_{\theta}(w_{t-1}, \dots, w_1))$ that conditions on the vector $\mathbf{h}_t = f_{\theta}(w_{t-1}, \dots, w_1)$

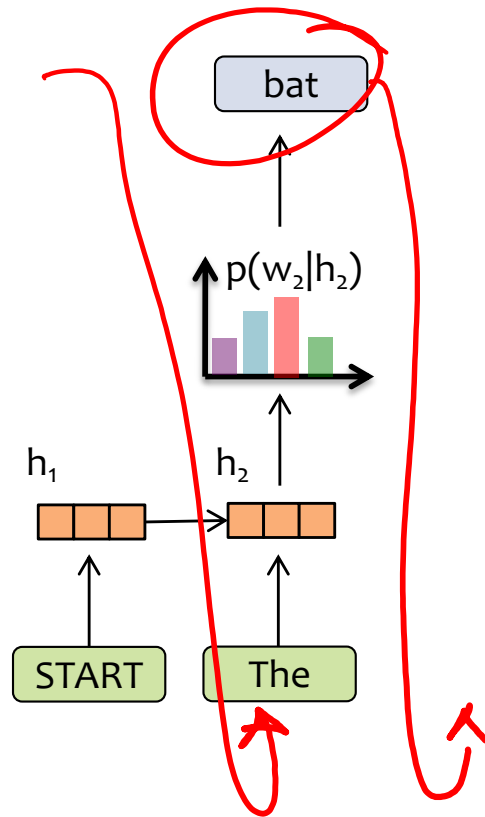
RNN Language Model



Key Idea:

- (1) convert all previous words to a **fixed length vector**
- (2) define distribution $p(w_t | f_{\theta}(w_{t-1}, \dots, w_1))$ that conditions on the vector $\mathbf{h}_t = f_{\theta}(w_{t-1}, \dots, w_1)$

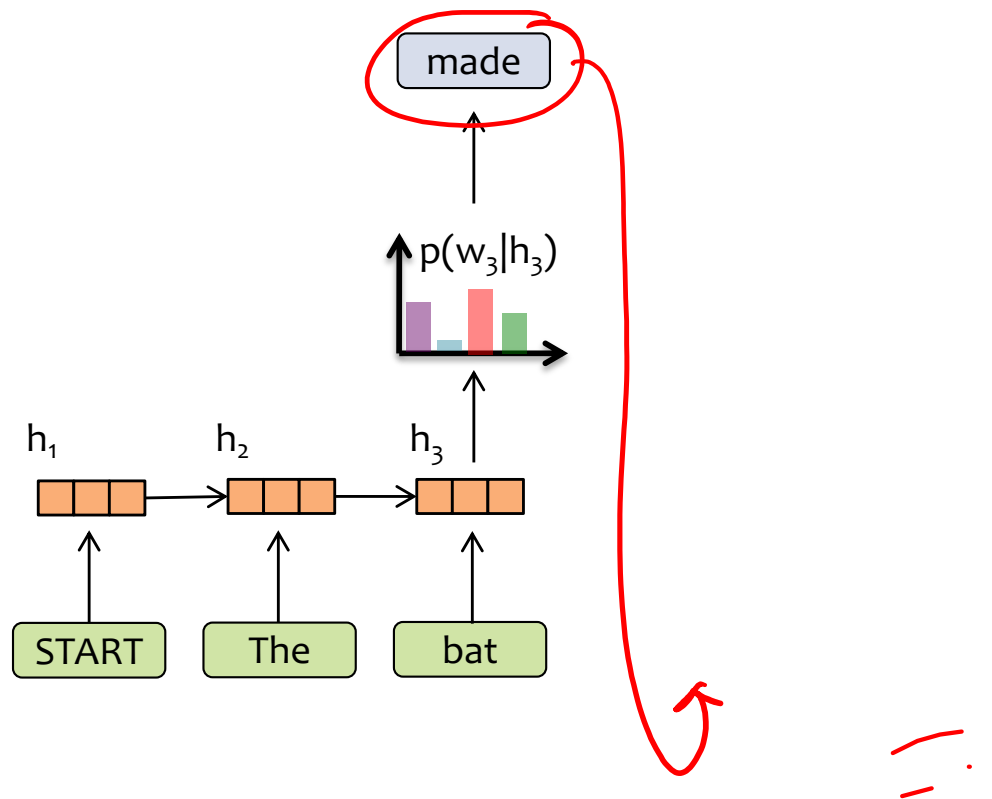
RNN Language Model



Key Idea:

- (1) convert all previous words to a **fixed length vector**
- (2) define distribution $p(w_t | f_{\theta}(w_{t-1}, \dots, w_1))$ that conditions on the vector $\mathbf{h}_t = f_{\theta}(w_{t-1}, \dots, w_1)$

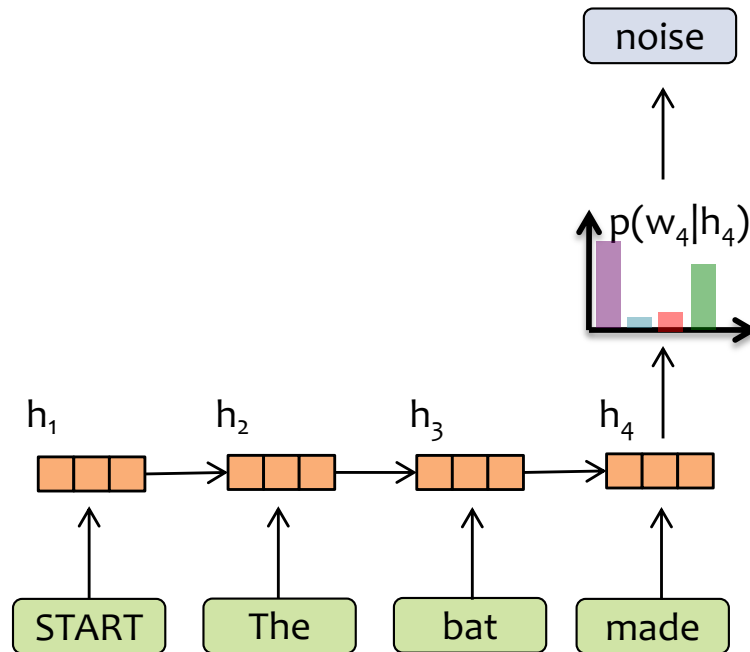
RNN Language Model



Key Idea:

- (1) convert all previous words to a **fixed length vector**
- (2) define distribution $p(w_t | f_{\theta}(w_{t-1}, \dots, w_1))$ that conditions on the vector $\mathbf{h}_t = f_{\theta}(w_{t-1}, \dots, w_1)$

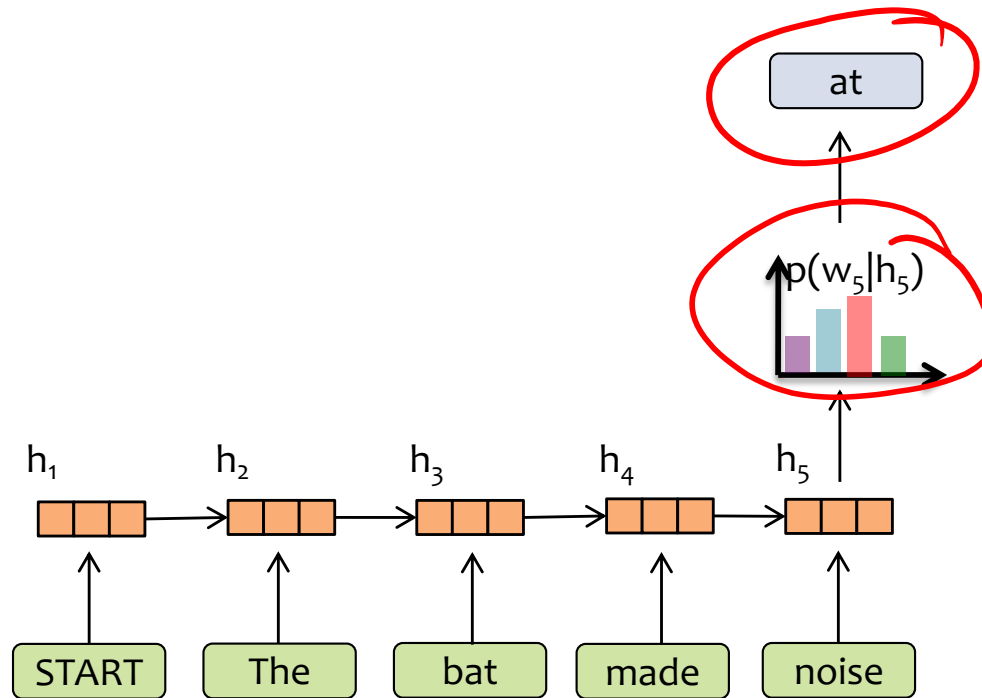
RNN Language Model



Key Idea:

- (1) convert all previous words to a **fixed length vector**
- (2) define distribution $p(w_t | f_{\theta}(w_{t-1}, \dots, w_1))$ that conditions on the vector $\mathbf{h}_t = f_{\theta}(w_{t-1}, \dots, w_1)$

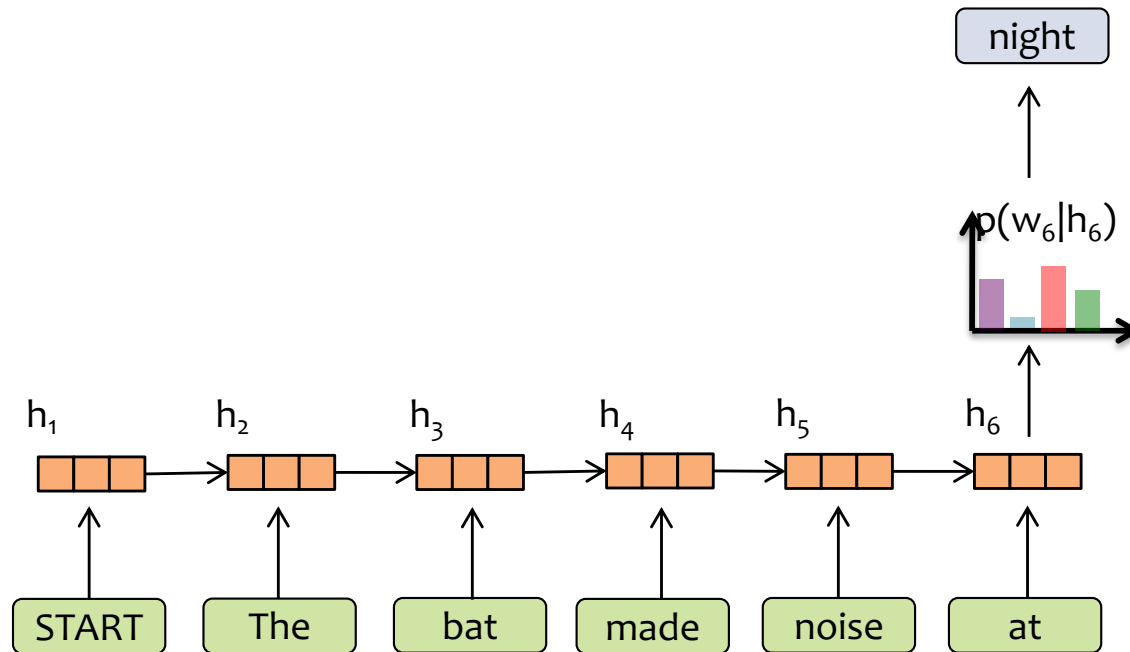
RNN Language Model



Key Idea:

- (1) convert all previous words to a **fixed length vector**
- (2) define distribution $p(w_t | f_{\theta}(w_{t-1}, \dots, w_1))$ that conditions on the vector $\mathbf{h}_t = f_{\theta}(w_{t-1}, \dots, w_1)$

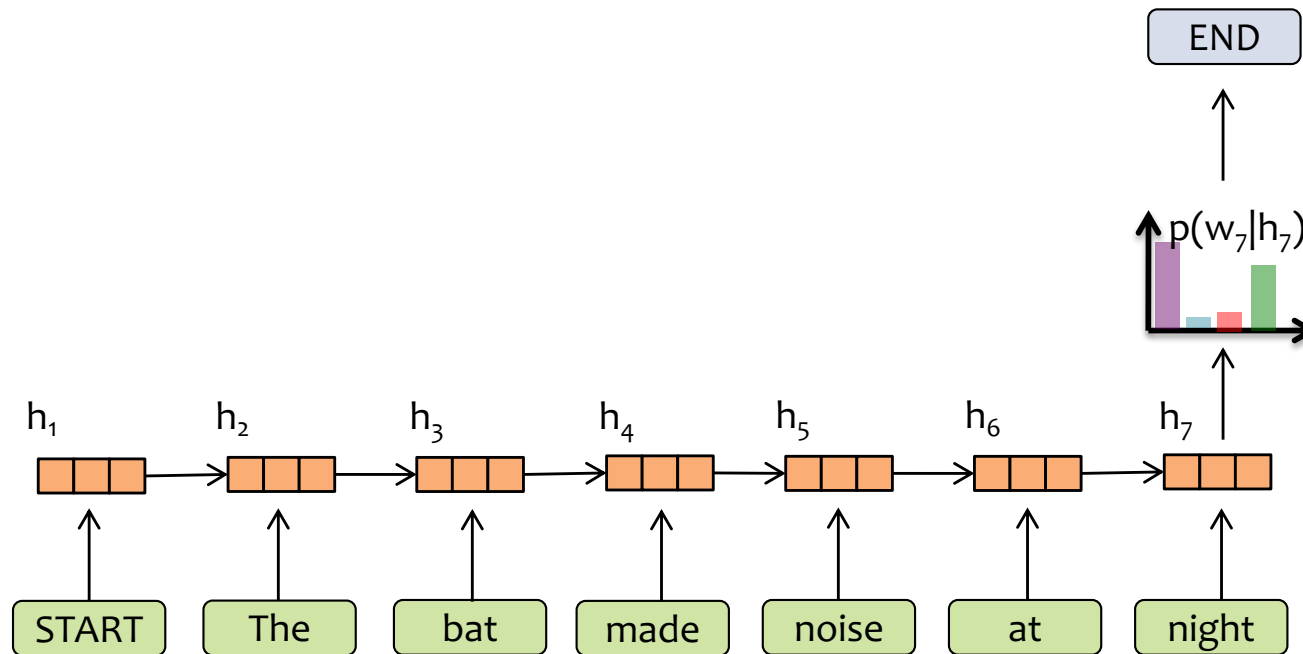
RNN Language Model



Key Idea:

- (1) convert all previous words to a **fixed length vector**
- (2) define distribution $p(w_t | f_{\theta}(w_{t-1}, \dots, w_1))$ that conditions on the vector $\mathbf{h}_t = f_{\theta}(w_{t-1}, \dots, w_1)$

RNN Language Model

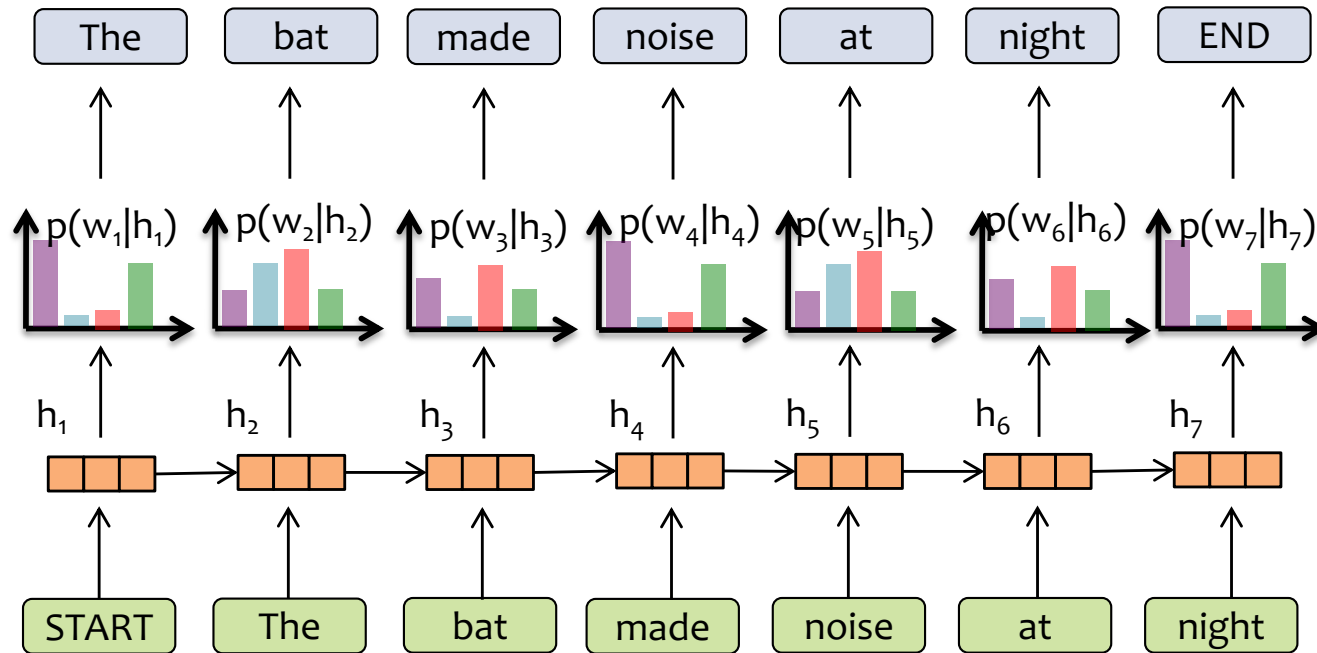


Key Idea:

- (1) convert all previous words to a **fixed length vector**
- (2) define distribution $p(w_t | f_{\theta}(w_{t-1}, \dots, w_1))$ that conditions on the vector $\mathbf{h}_t = f_{\theta}(w_{t-1}, \dots, w_1)$

RNN Language Model

$p(\text{The}/h_1)$ $p(\text{bat}/h_2)$ $p(\text{made}/h_3)$ $\dots$ $p(\text{END}/h_7)$



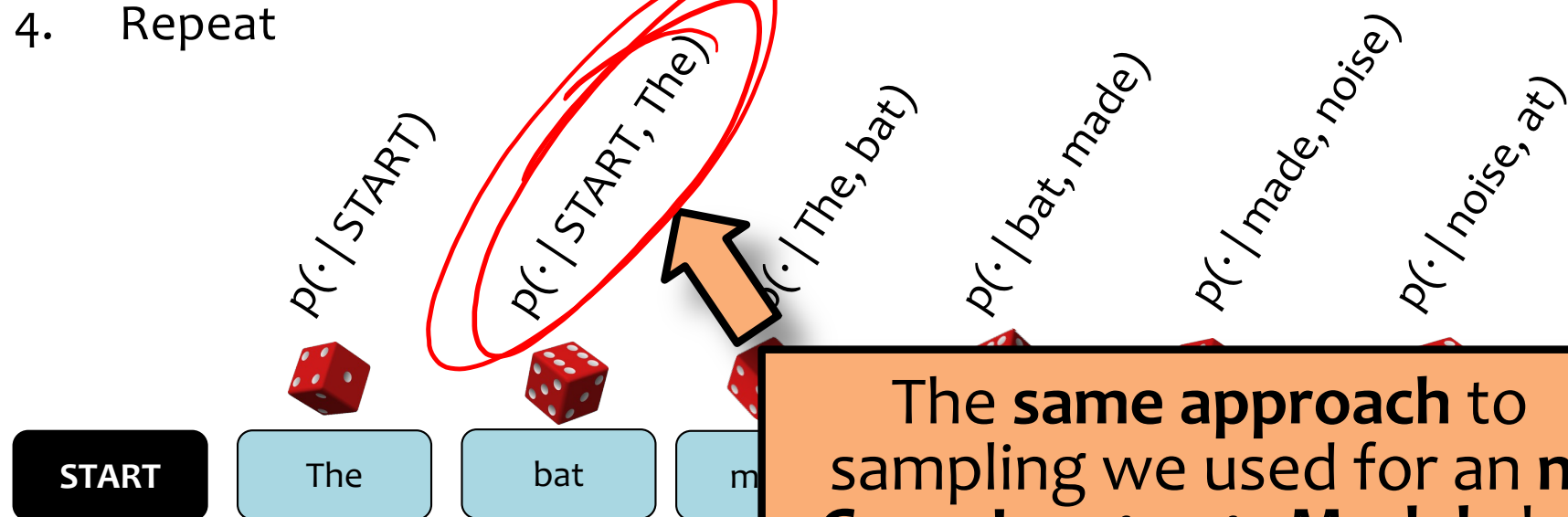
$$p(w_1, w_2, w_3, \dots, w_T) = p(w_1 | h_1) p(w_2 | h_2) \dots p(w_T | h_T)$$

Sampling from a Language Model

Question: How do we sample from a Language Model?

Answer:

1. Treat each probability distribution like a (50k-sided) weighted die
2. Pick the die corresponding to $p(w_t | w_{t-2}, w_{t-1})$
3. Roll that die and generate whichever word w_t lands face up
4. Repeat



The **same approach** to sampling we used for an **n-Gram Language Model** also works here for an **RNN Language Model**

Sampling from an RNN-LM

??

VIOLA: Why, Salisbury must find his flesh and thought That which I am not apt, 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 a master's ready there My power so much as hell: Some service i bondman here, Would show hi

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.

??

CHARLES: Marry, do I, sir; and I came to acquaint you with a matter. I am given, sir, secretly to understand that your younger brother Orlando hath a disposition to come in disguised against me to try a fall. To-morrow, sir, I wrestle for my credit; and he that escapes me without some broken limb shall acquit him. He is but young and tender; and I should be loath to foil him, as I should lose my honour, if he come in: My love to you, I came hither to acquaint you withal, that either you might stay him from his intention or brook such disgrace well as he shall run into, in that it is a thing of his own search and altogether against my will.

TOUCHSTONE: For my part, I had rather bear with you than bear you; yet I should bear no cross if I did bear you, for I think you have no money in your purse.

Which is the real Shakespeare?!

Sampling from an RNN-LM

Shakespeare's As You Like 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.

RNN-LM Sample

CHARLES: Marry, do I, sir; and I came to acquaint you with a matter. I am given, sir, secretly to understand that your younger brother Orlando hath a disposition to come in disguised against me to try a fall. To-morrow, sir, I wrestle for my credit; and he that escapes me without some broken limb shall acquit him well. Your brother is but young and tender; and, for your love, I would be loath to foil him, as I must, for my own honour, if he come in: therefore, out of my love to you, I came hither to acquaint you withal, that either you might stay him from his intendment or brook such disgrace well as he shall run into, in that it is a thing of his own search and altogether against my will.

TOUCHSTONE: For my part, I had rather bear with you than bear you; yet I should bear no cross if I did bear you, for I think you have no money in your purse.

Sampling from an RNN-LM

RNN-LM Sample

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.

Shakespeare's As You Like It

CHARLES: Marry, do I, sir; and I came to acquaint you with a matter. I am given, sir, secretly to understand that your younger brother Orlando hath a disposition to come in disguised against me to try a fall. To-morrow, sir, I wrestle for my credit; and he that escapes me without some broken limb shall acquit him well. Your brother is but young and tender; and, for your love, I would be loath to foil him, as I must, for my own honour, if he come in: therefore, out of my love to you, I came hither to acquaint you withal, that either you might stay him from his intendment or brook such disgrace well as he shall run into, in that it is a thing of his own search and altogether against my will.

TOUCHSTONE: For my part, I had rather bear with you than bear you; yet I should bear no cross if I did bear you, for I think you have no money in your purse.

Sampling from an RNN-LM

??

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 a master's ready there My power so much as hell: Some service i bondman here, Would show hi

KING LEAR: O, if you we 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.

??

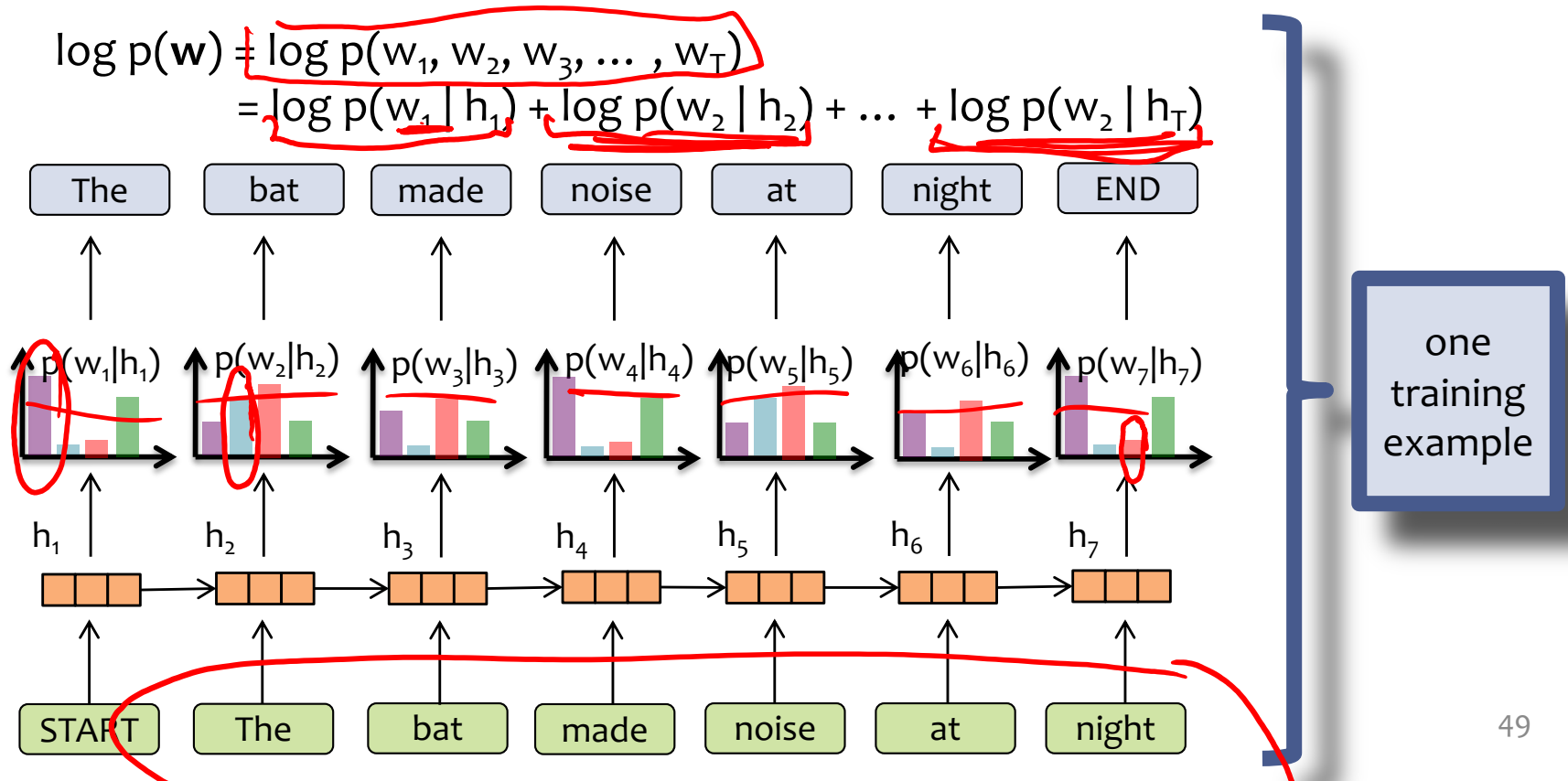
CHARLES: Marry, do I, sir; and I came to acquaint you with a matter. I am given, sir, secretly to understand that your younger brother Orlando hath a disposition to come in disguised against me to try a fall. To-morrow, sir, I wrestle for my credit; and he that escapes me without some broken limb shall acquit him is but young and tender; and, I should be loath to foil him, as I honour, if he come in: my love to you, I came hither to acquaint you with, that either you might stay him from his intent or brook such disgrace well as he shall run into, in that it is a thing of his own search and altogether against my will.

TOUCHSTONE: For my part, I had rather bear with you than bear you; yet I should bear no cross if I did bear you, for I think you have no money in your purse.

Which is the real Shakespeare?!

Learning an RNN-LM

- Each training example is a sequence (e.g. sentence), so we have training data $D = \{\mathbf{w}^{(1)}, \mathbf{w}^{(2)}, \dots, \mathbf{w}^{(N)}\}$
- The objective function for an RNN-LM is (typically) the log-likelihood of the training examples: $J(\boldsymbol{\theta}) = \sum_i \log p_{\boldsymbol{\theta}}(\mathbf{w}^{(i)})$
- We train by mini-batch SGD (or your favorite flavor of mini-batch SGD)

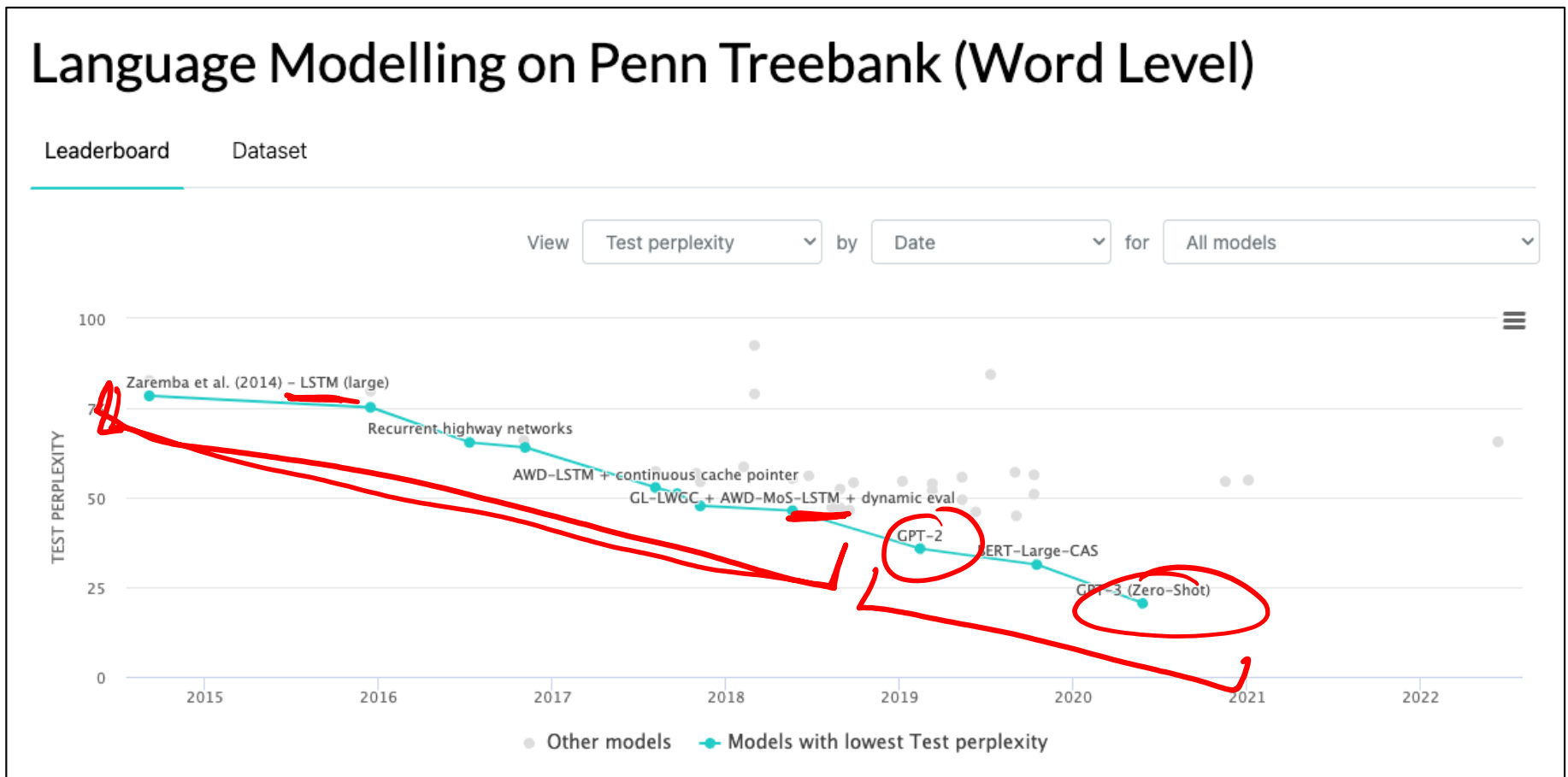


Language Modeling

LM NC joint
 $p(x)p(y|x) = p(x,y) \approx p(x|y)$

An aside:

- State-of-the-art language models currently tend to rely on **transformer networks** (e.g. GPT-2)
- RNN-LMs comprised most of the early neural LMs that **led to** current SOTA architectures



Why does efficiency matter?

Case Study: GPT-3

- # of training tokens = 500 billion
- # of parameters = 175 billion
- # of cycles = 50 petaflop/s-days (each of which are 8.64×10^{19} flops)

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

Table 2.2: Datasets used to train GPT-3. “Weight in training mix” refers to the fraction of examples during training that are drawn from a given dataset, which we intentionally do not make proportional to the size of the dataset. As a result, when we train for 300 billion tokens, some datasets are seen up to 3.4 times during training while other datasets are seen less than once.

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or “GPT-3”	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

Table 2.1: Sizes, architectures, and learning hyper-parameters (batch size in tokens and learning rate) of the models which we trained. All models were trained for a total of 300 billion tokens.

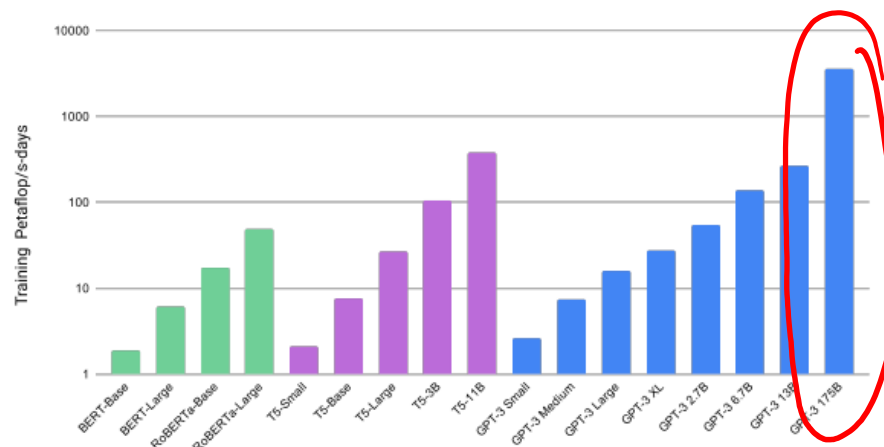
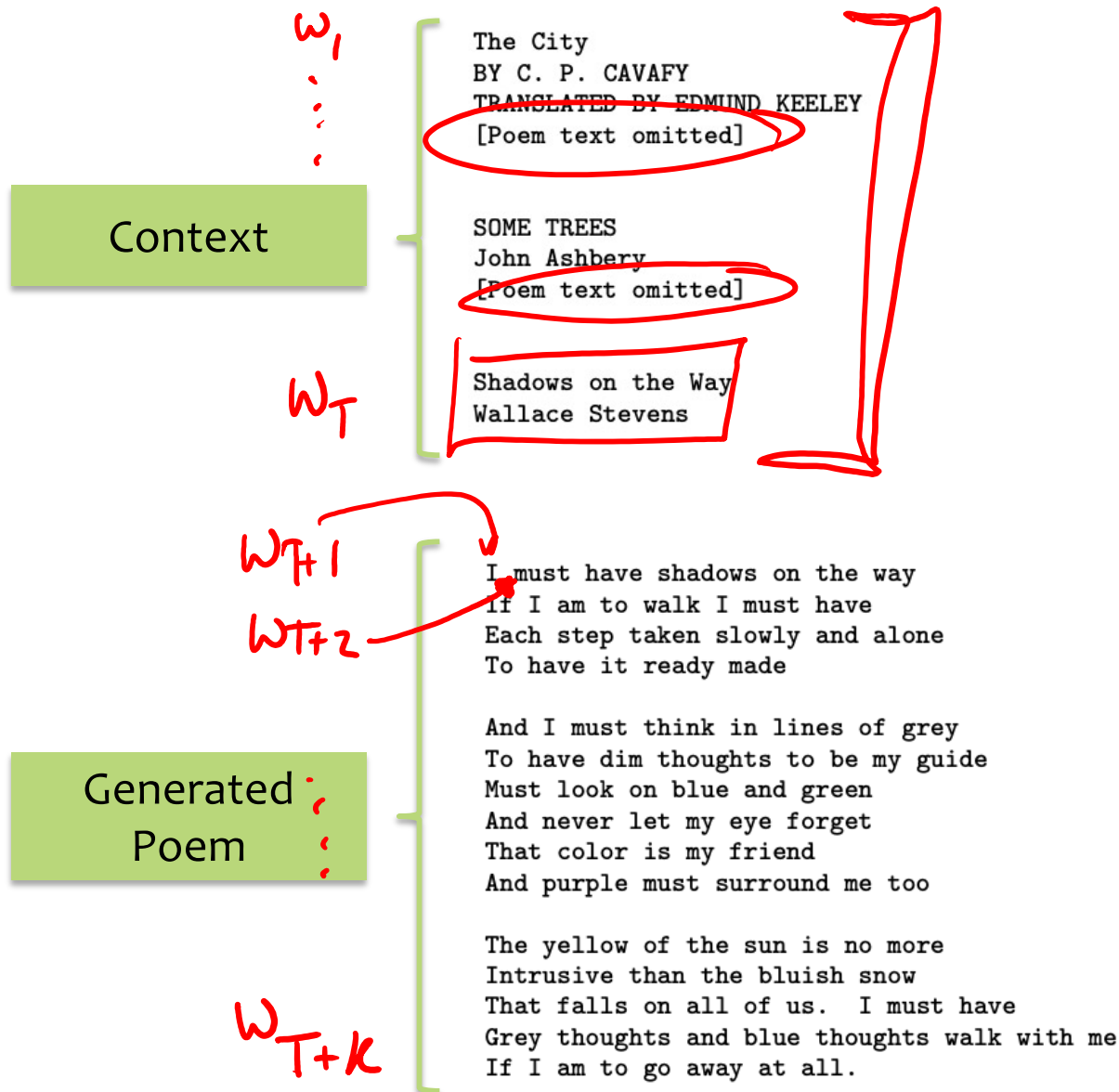


Figure 2.2: Total compute used during training. Based on the analysis in Scaling Laws For Neural Language Models [KMH⁺20] we train much larger models on many fewer tokens than is typical. As a consequence, although GPT-3 3B is almost 10x larger than RoBERTa-Large (355M params), both models took roughly 50 petaflop/s-days of compute during pre-training. Methodology for these calculations can be found in Appendix D.

GPT-3 Example Output



Conditional Sampling:

To draw a sample, feed in a prompt (i.e. context) and then sample from the model conditioned on that prompt

What else can a large LM (attempt to) do?

Using the idea of **prompts**, we can apply LMs to a **variety of different problems** in natural language processing.

In the **zero-shot setting**, we simply feed the context to the model and observe how it completes the sequence. (i.e. there is no additional training)

Answer fact-based questions:

Context →	<u>Organisms require energy in order to do what?</u>
Correct Answer →	mature and develop.
Incorrect Answer →	rest soundly.
Incorrect Answer →	absorb light.
Incorrect Answer →	take in nutrients.

Complete sentences logically:

Context →	<u>My body cast a shadow over the grass because</u>
Correct Answer →	the sun was rising. ←
Incorrect Answer →	the grass was cut. ←

Complete sentences logically:

Context →	lull is to trust as
Correct Answer →	cajole is to compliance
Incorrect Answer →	balk is to fortitude
Incorrect Answer →	betray is to loyalty
Incorrect Answer →	hinder is to destination
Incorrect Answer →	soothe is to passion

Reading comprehension:

Context →	anli 1: anli 1: Fulton James MacGregor MSP is a Scottish politician who is a Scottish National Party (SNP) Member of Scottish Parliament for the constituency of Coatbridge and Chryston. MacGregor is currently Parliamentary Liaison Officer to Shona Robison, Cabinet Secretary for Health & Sport. He also serves on the Justice and Education & Skills committees in the Scottish Parliament. Question: Fulton James MacGregor is a Scottish politician who is a Liaison officer to Shona Robison who he swears is his best friend. True, False, or Neither?
Correct Answer →	Neither
Incorrect Answer →	True
Incorrect Answer →	False

RNN Language Models

Whiteboard:

- RNNLM for scoring of a path in a search space
- What's missing? Dependence on the input.

SEQUENCE TO SEQUENCE MODELS

Why seq2seq?

Motivating Question:

How can we model input/output pairs when the length of the input might be **different** from the length of the output?

Why seq2seq?

- **10+ years ago:** state-of-the-art machine translation or speech recognition systems were complex pipelines
 - MT
 - unsupervised word-level alignment of sentence-parallel corpora (e.g. via GIZA++)
 - build phrase tables based on (noisily) aligned data (use prefix trees and on demand loading to reduce memory demands)
 - use factored representation of each token (word, POS tag, lemma, morphology)
 - learn a separate language model (e.g. SRILM) for target
 - combine language model with phrase-based decoder
 - tuning via minimum error rate training (MERT)
 - ASR
 - MFCC and PLP feature extraction
 - acoustic model based on Gaussian Mixture Models (GMMs)
 - model phones via Hidden Markov Models (HMMs)
 - learn a separate n-gram language model
 - learn a phonetic model (i.e. mapping words to phones)
 - combine language model, acoustic model, and phonetic model in a weighted finite-state transducer (WFST) framework (e.g. OpenFST)
 - decode from a confusion network (lattice)
- **Today:** just use a seq2seq model
 - *encoder*: reads the input one token at a time to build up its vector representation
 - *decoder*: starts with encoder vector as context, then decodes one token at a time – feeding its own outputs back in to maintain a vector representation of what was produced so far

Sequence to Sequence Model

Speech Recognition



Machine Translation

기계 번역은 특히 영어와 한국어와 같은 언어 쌍의 경우 매우 어렵습니다.

Summarization

[illegible]

Sequence to Sequence Model

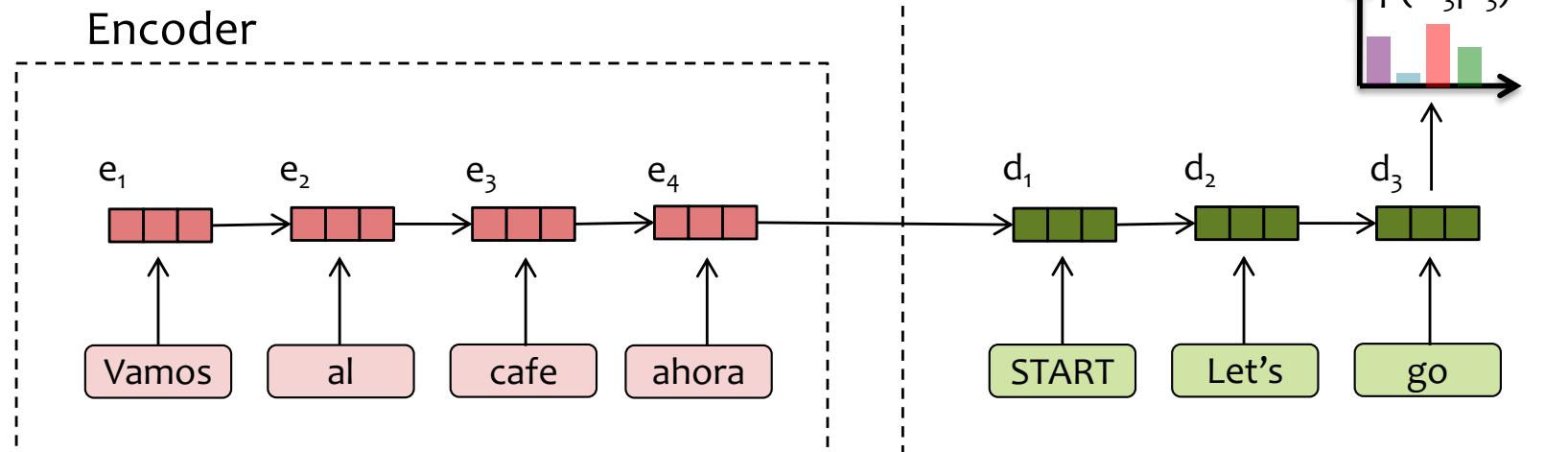
Now suppose you want generate a sequence conditioned on another input

Key Idea:

1. Use an **encoder** model to generate a vector representation of the **input**
2. Feed the output of the encoder to a **decoder** which will generate the **output**

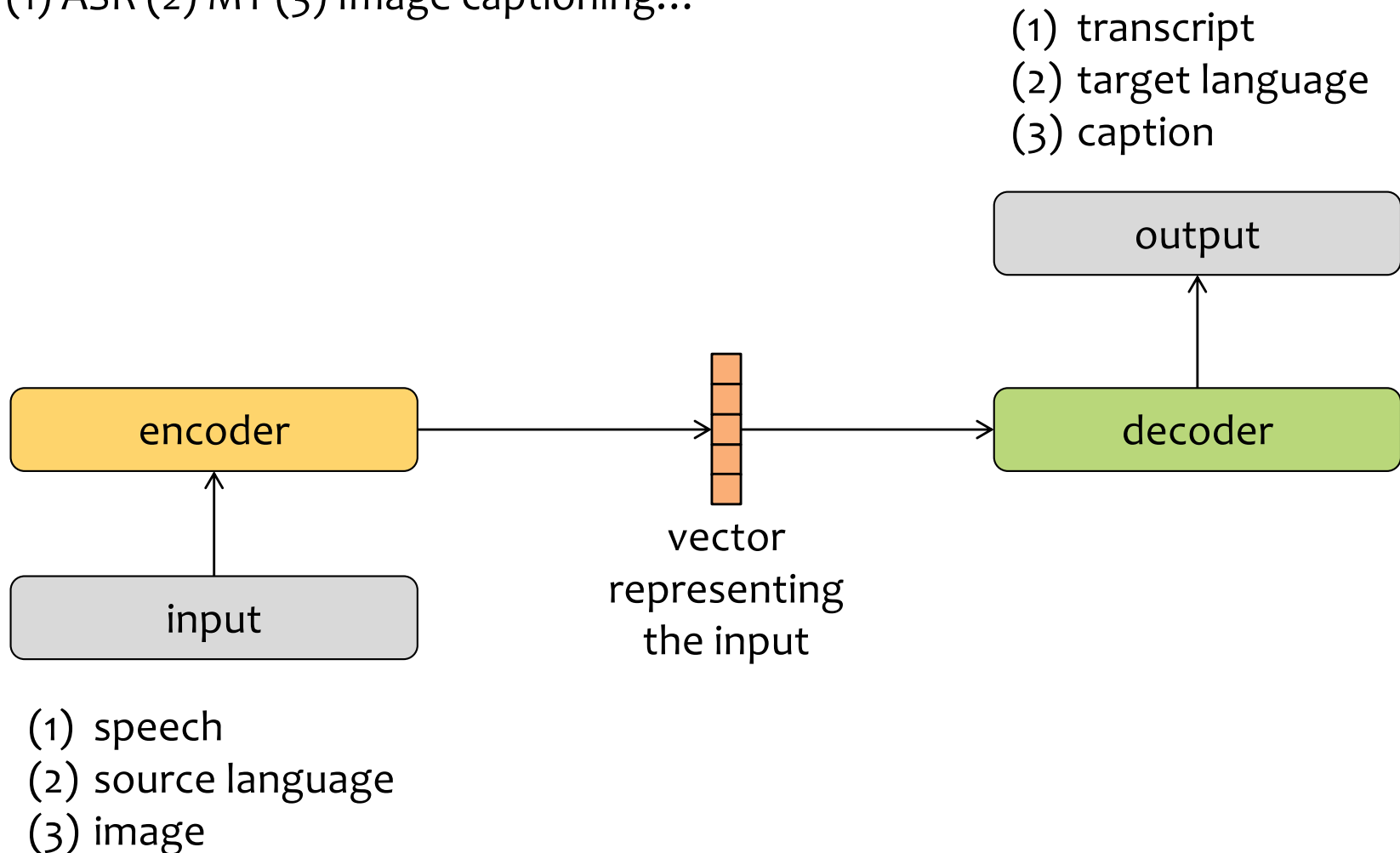
Applications:

- translation: Spanish → English
- summarization: article → summary
- speech recognition: speech signal → transcription



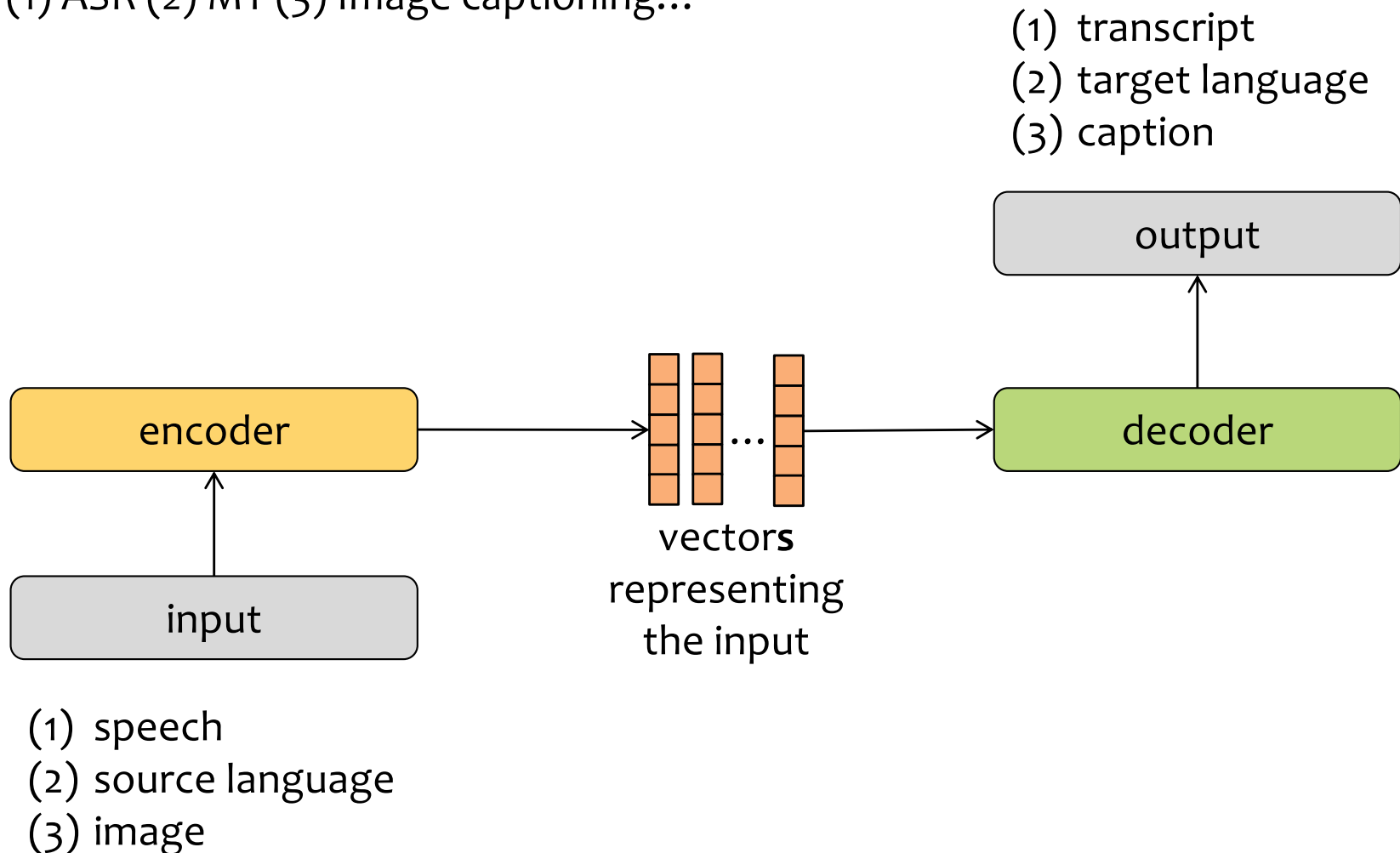
Encoder-Decoder Architectures

For (1) ASR (2) MT (3) Image captioning...



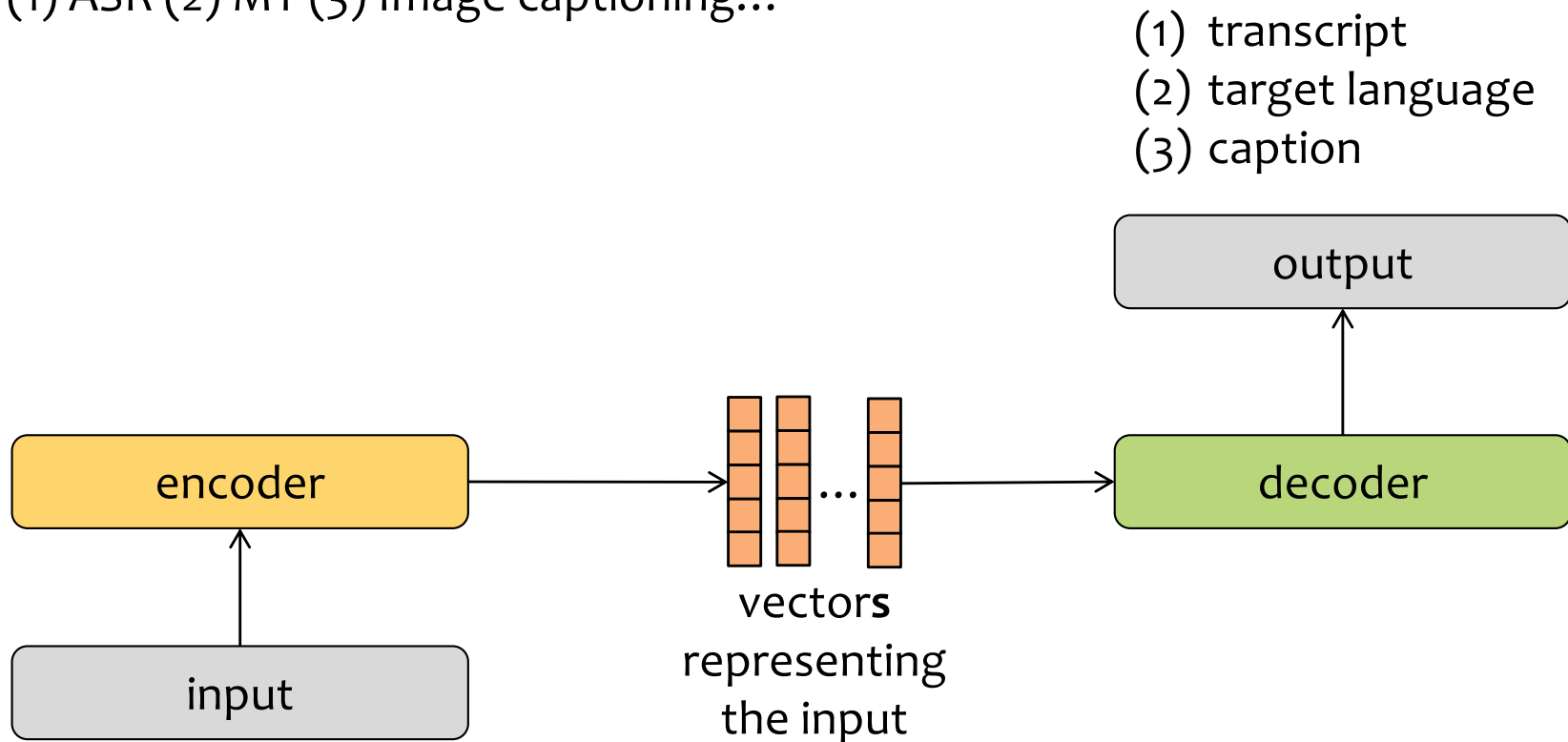
Encoder-Decoder Architectures

For (1) ASR (2) MT (3) Image captioning...



Encoder-Decoder Architectures

For (1) ASR (2) MT (3) Image captioning...



(1) transcript
(2) target language
(3) caption

(1) speech
(2) source language
(3) image



- A seq2seq model is one flavor of the (more general) encoder-decoder architecture.
- Image captioning (or video captioning) provides an example in which the input is not a sequence, and the encoder must handle its 2D (or 3D) input.

Comparing RNN, RNN-LM, seq2seq

Question: Q1

Fill in the blank: A recurrent neural network (RNN) is a ____.

- * ☒ A. discriminative model
- B. generative model

Answer:

$$p(y|x) \quad 50\% \quad 50\%$$

Question: Q2

Fill in the blank: An RNN-LM is a ____.

- A. discriminative model
- ☒ B. generative model

Answer:

$$p(x,y) \\ p(x)$$

Question: Q3

Fill in the blank: A seq1seq model is a ____.

- ☒ A. discriminative model
- B. generative model

Answer:

$$\underline{p(y|x)}$$