Complex Neural Networks

RNNs and LSTMs

Recap: Language Model

Language model defines "how probable a sentence is"

John is driving a car vs. John is driving a cat

In other words, what is the probability to predict **cat** or **car** given the context "John is driving a"

Shortcomings of NNLM

Q: What are some shortcomings of the feed forward neural network language model that we have seen so far?

Shortcomings of NNLM

Q: What are some shortcomings of the feed forward neural network language model that we have seen so far?

- Independence assumption: We have a “hard” limit on the amount of context we see - bigram, trigram or some ngram.

Shortcomings of NNLM

Q: What are some shortcomings of the feed forward neural network language model that we have seen so far?

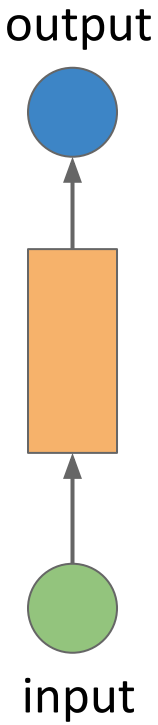
- Independence assumption: We have a "hard" limit on the amount of context we see - bigram, trigram or some ngram.
- Limit can *never* large enough

Shortcomings of NNLM

Q: What are some shortcomings of the feed forward neural network language model that we have seen so far?

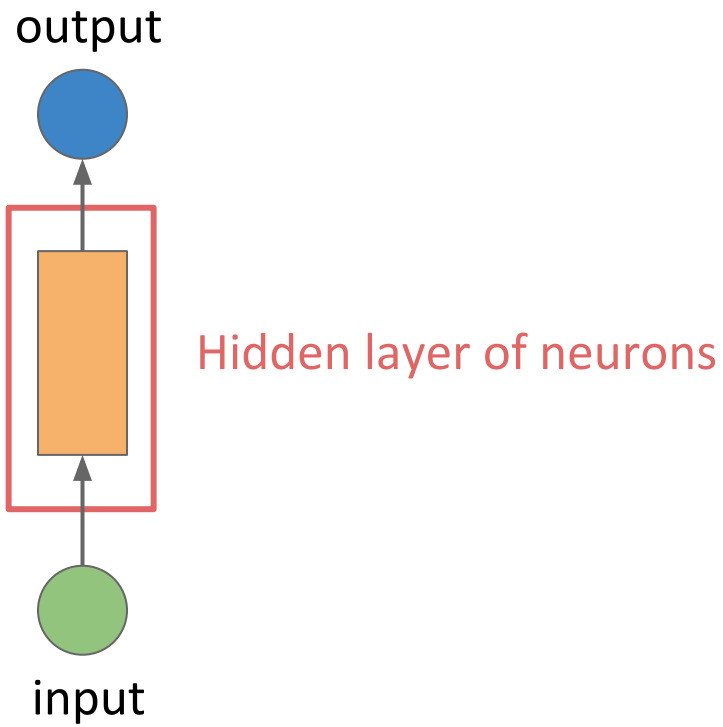
- Independence assumption: We have a “hard” limit on the amount of context we see - bigram, trigram or some ngram.
- Limit can *never* large enough
- It is not uncommon to have longer range dependencies in language!

Recurrent Neural Network



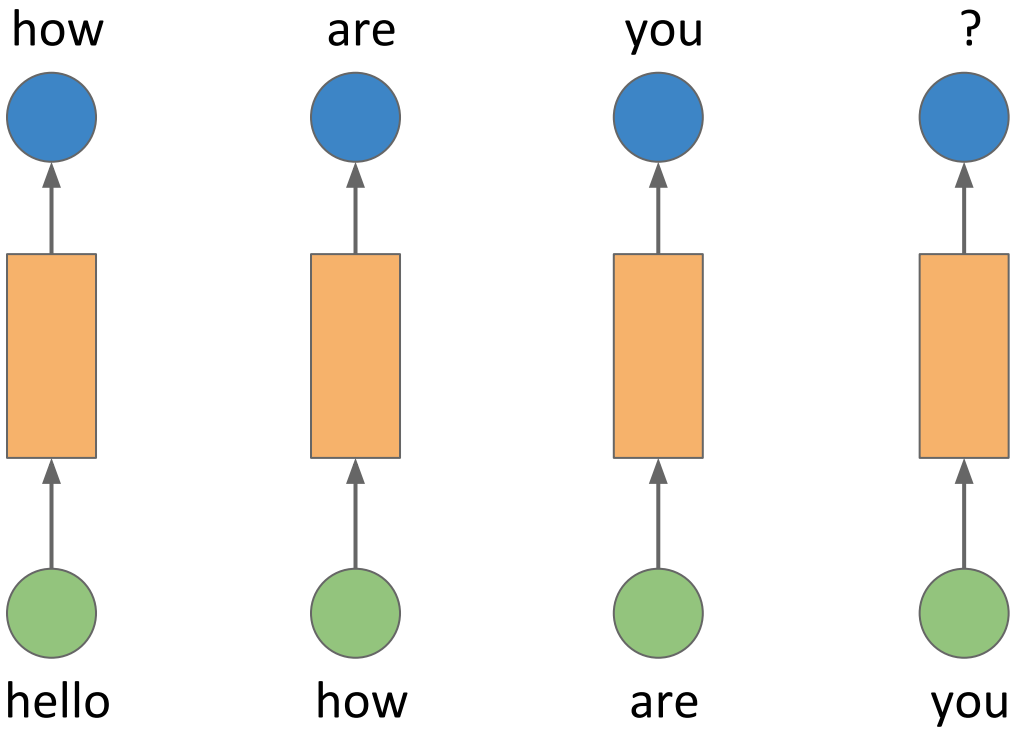
1-layer Feedforward Neural network

Recurrent Neural Network

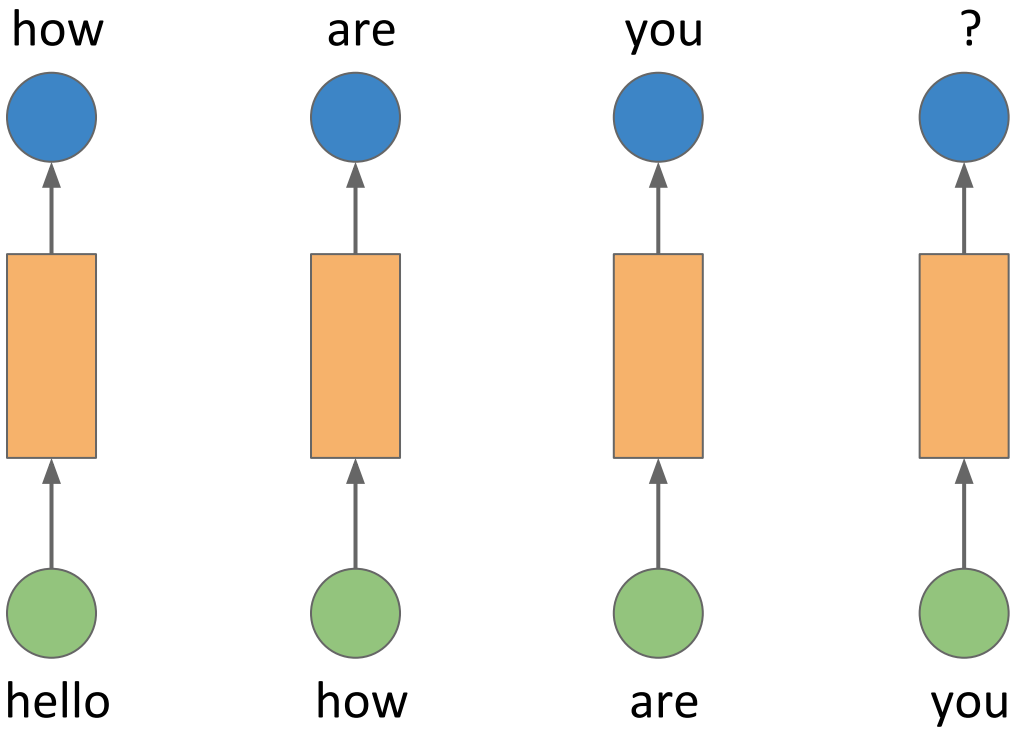


1-layer Feedforward Neural network

Recurrent Neural Network

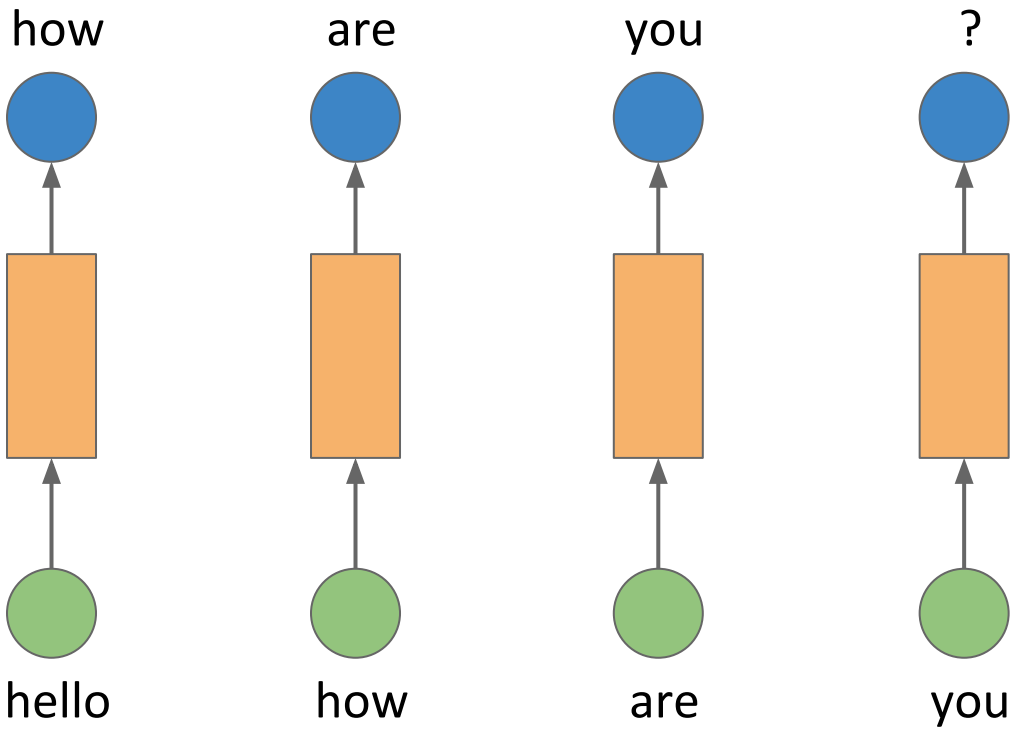


Recurrent Neural Network



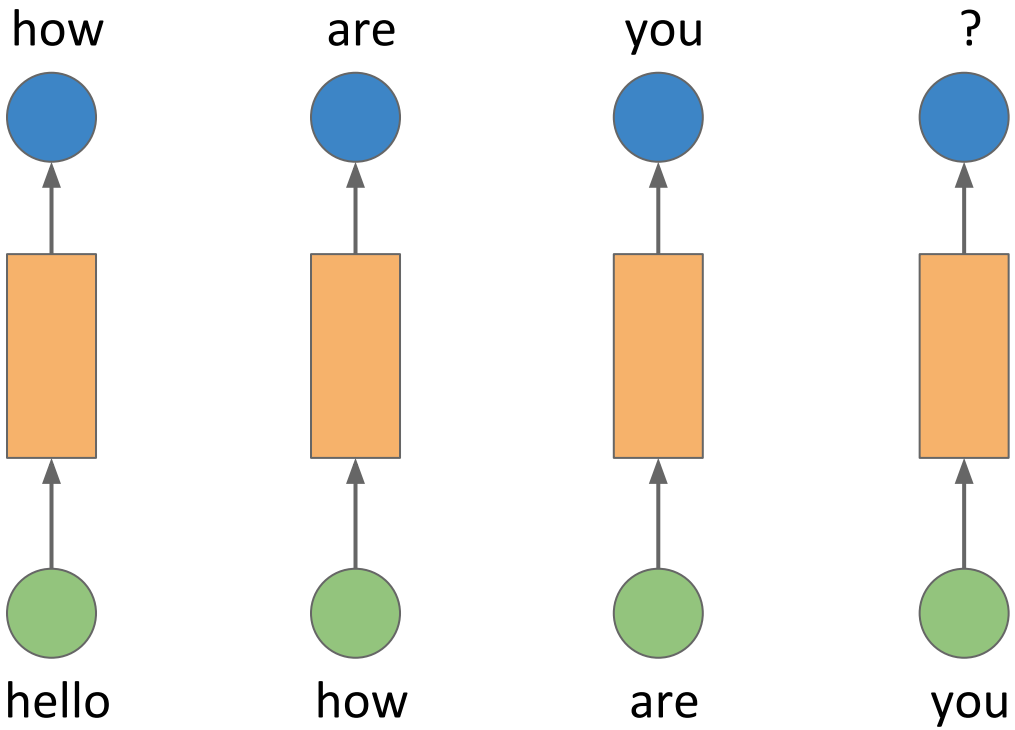
In the real world, we remember some history of previous words

Recurrent Neural Network



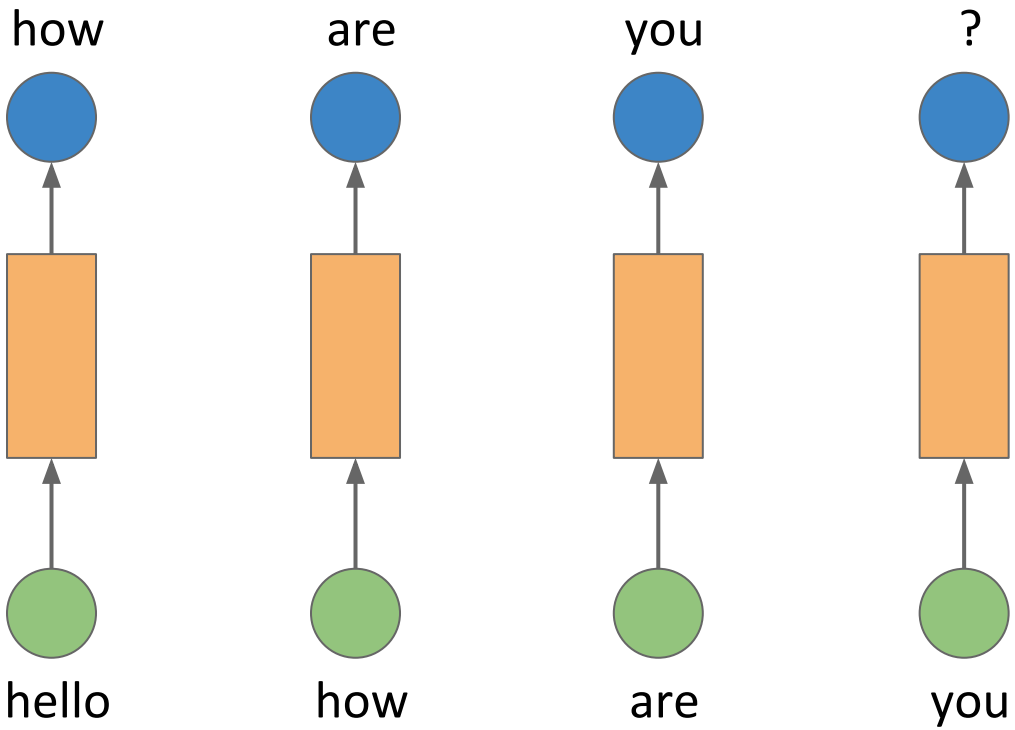
In our network here, each step is independent of the previous steps

Recurrent Neural Network



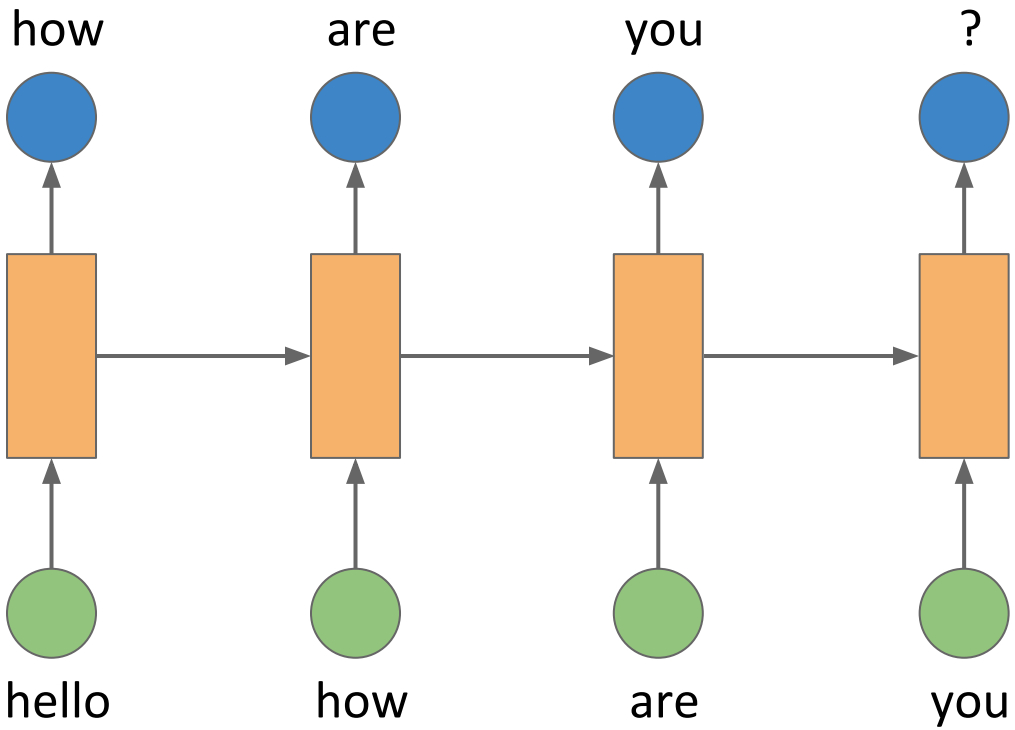
The only context available at every step is the input we provide to the network (bigram, trigram etc)

Recurrent Neural Network



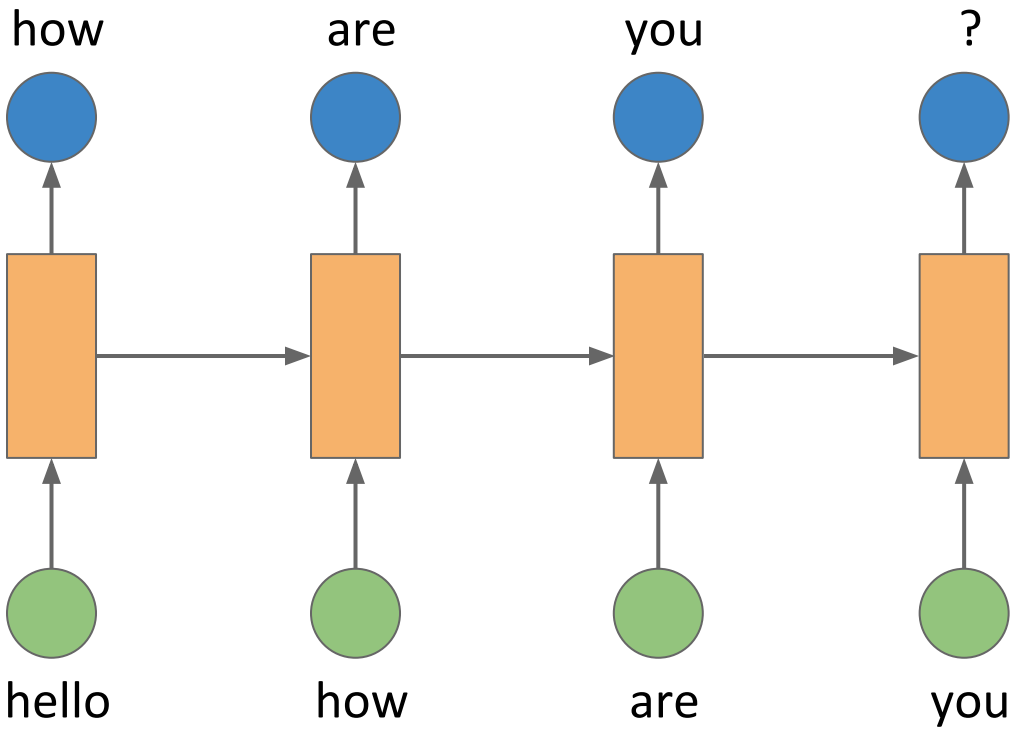
Why not connect these networks?

Recurrent Neural Network



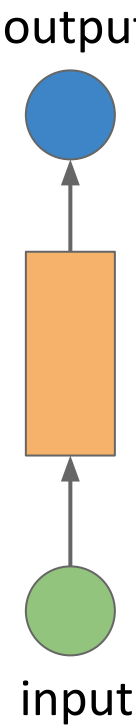
Why not connect these networks?

Recurrent Neural Network



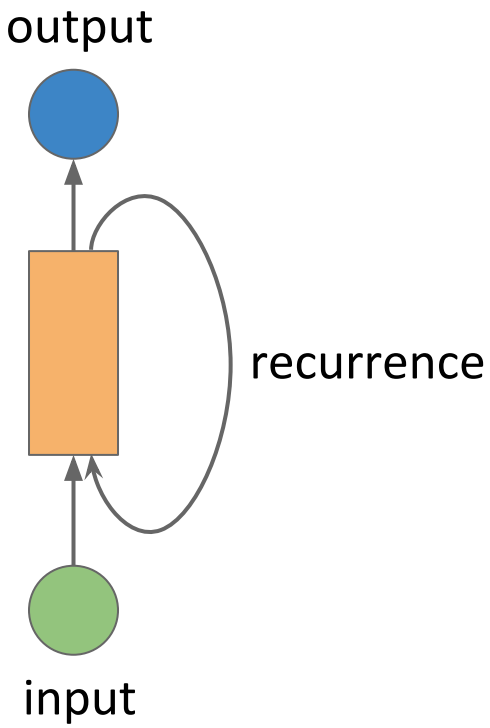
This is what recurrent neural networks do

Recurrent Neural Network



1-layer Feedforward

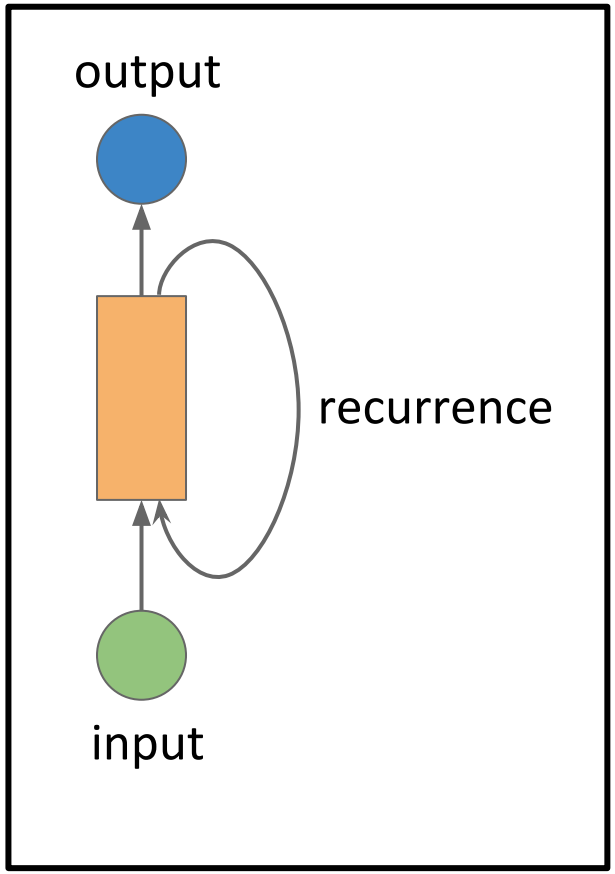
Neural network



1-layer Recurrent

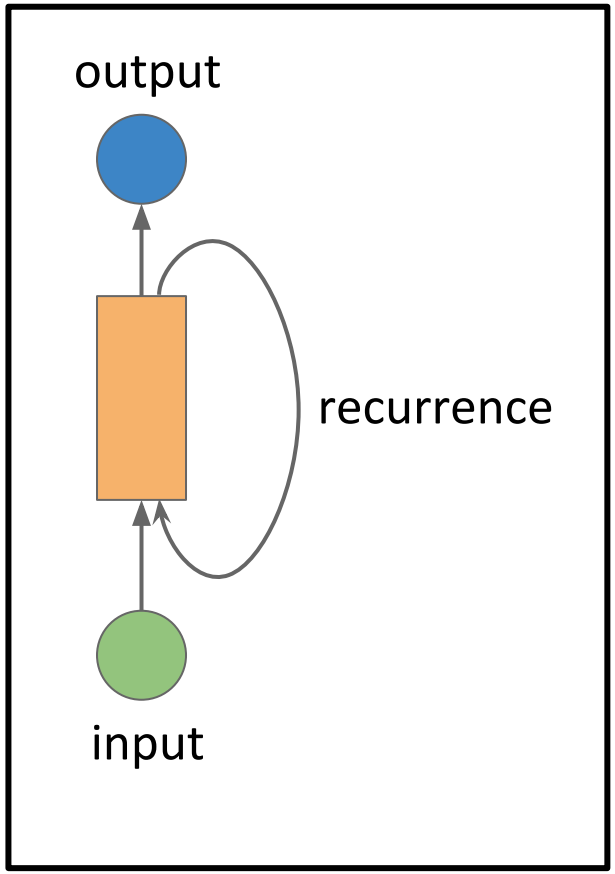
Neural network

Recurrent Neural Network

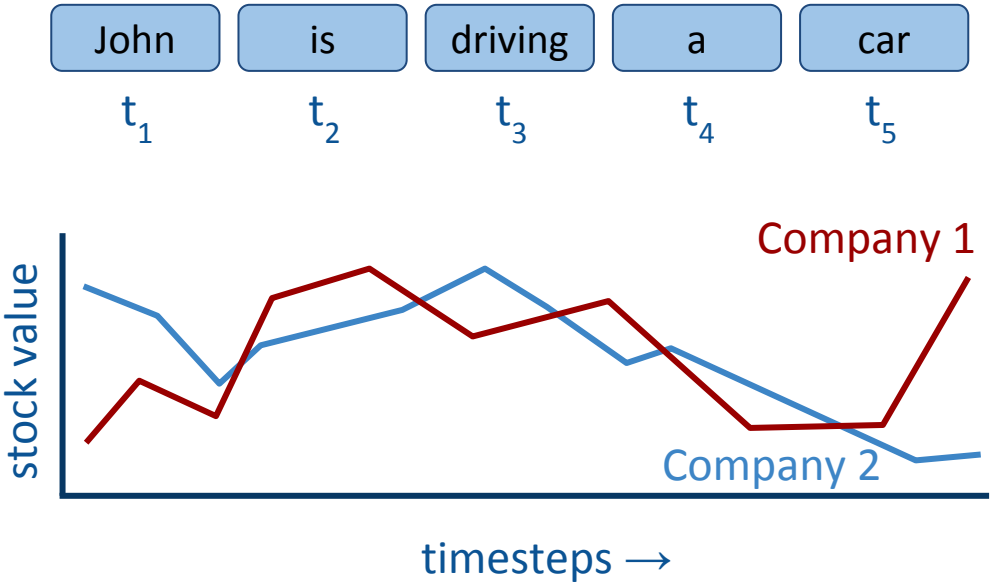


Recurrent units work very well for sequential information like a series of words, or knowledge across *timesteps*

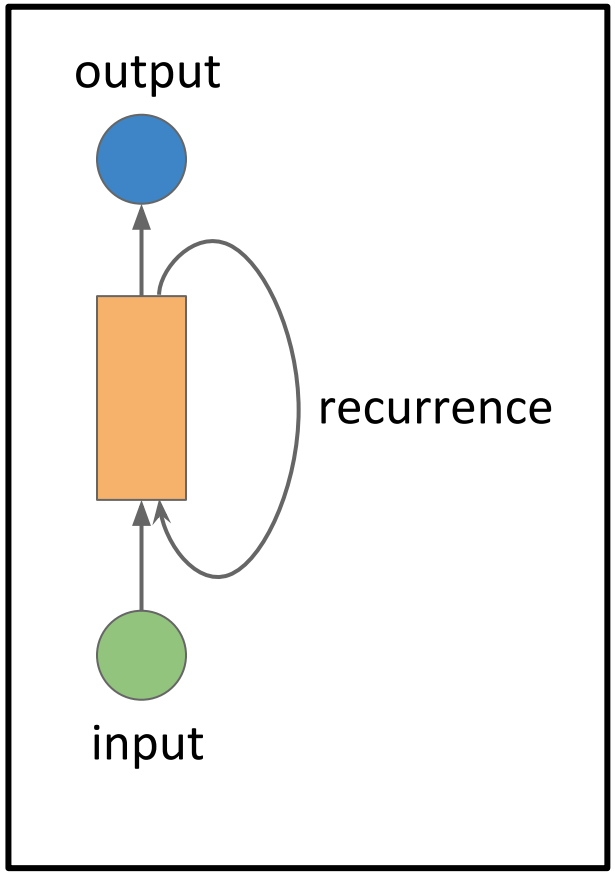
Recurrent Neural Network



Recurrent units work very well for sequential information like a series of words, or knowledge across *timesteps*



Recurrent Neural Network



Recurrent units work very well for sequential information like a series of words, or knowledge across *timesteps*

The recurrence unit has two inputs:

1) x_i (input at time i)
2) h_{i-1} (input from previous state)

Recurrent Neural Network

Mathematically,

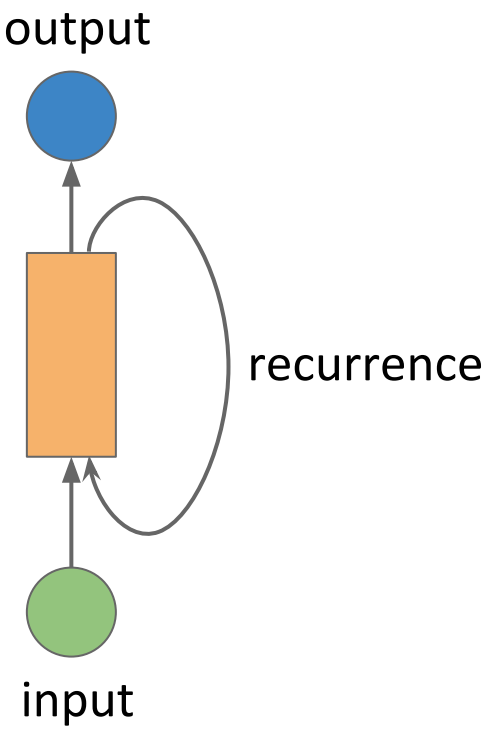
$$h = Wx + b \longrightarrow h_t = Wx + W_h h_{t-1} + b$$

Linear — Recurrent

We have one additional set of parameters: W_h which deals with the information transferred from the previous step

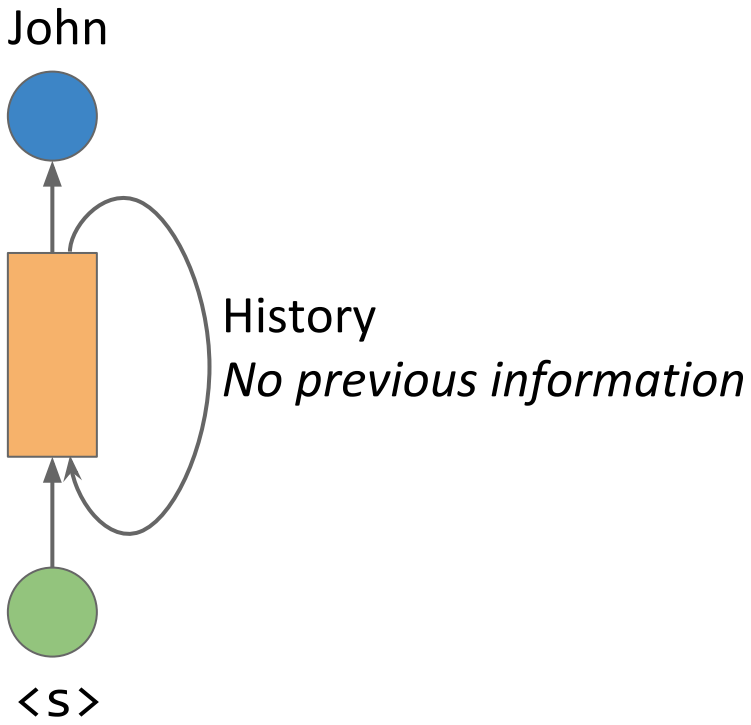
Recurrent Neural Network

Consider an example: <s> John is driving a car </s>



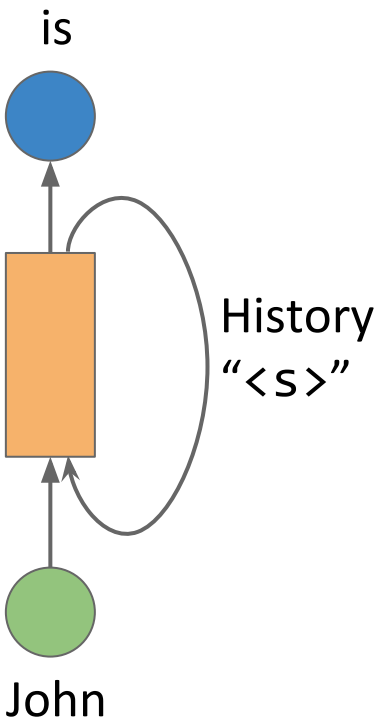
Recurrent Neural Network

Consider an example: <s> John is driving a car </s>



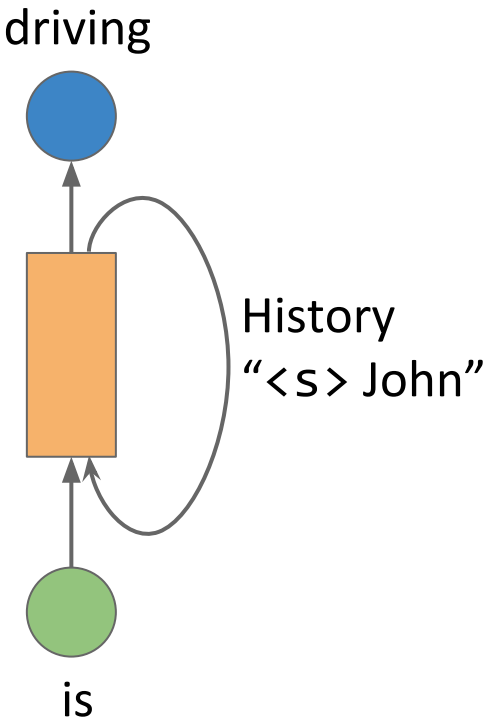
Recurrent Neural Network

Consider an example: <s> John is driving a car </s>



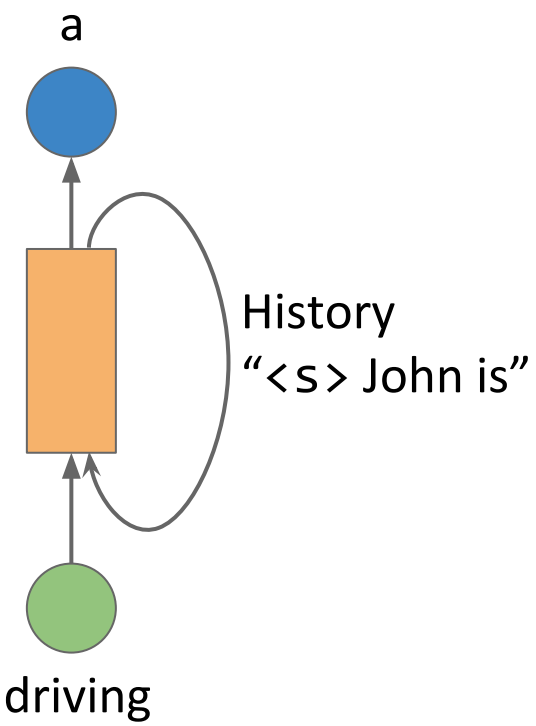
Recurrent Neural Network

Consider an example: <s> John is driving a car </s>



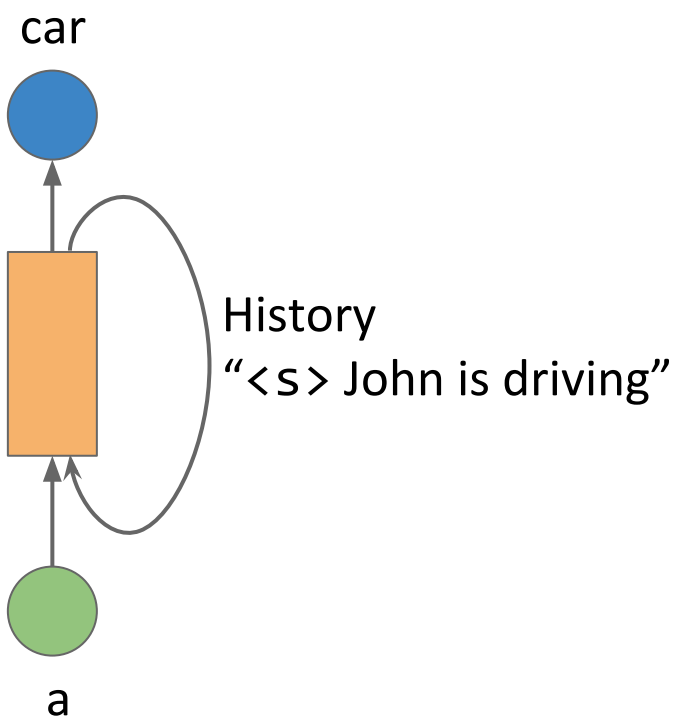
Recurrent Neural Network

Consider an example: <s> John is driving a car </s>



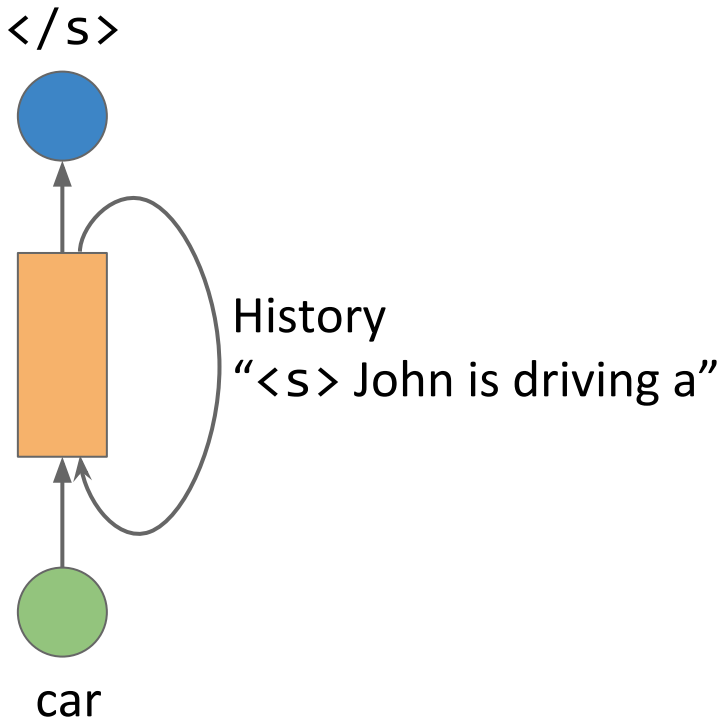
Recurrent Neural Network

Consider an example: <s> John is driving a car </s>



Recurrent Neural Network

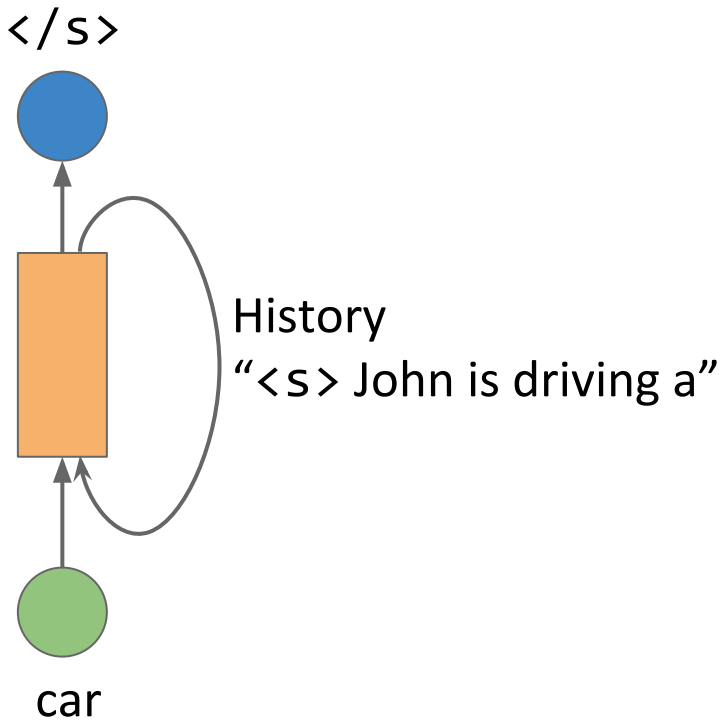
Consider an example: <s> John is driving a car </s>



Recurrent Neural Network

Consider an example: <s> John is driving a car </s>

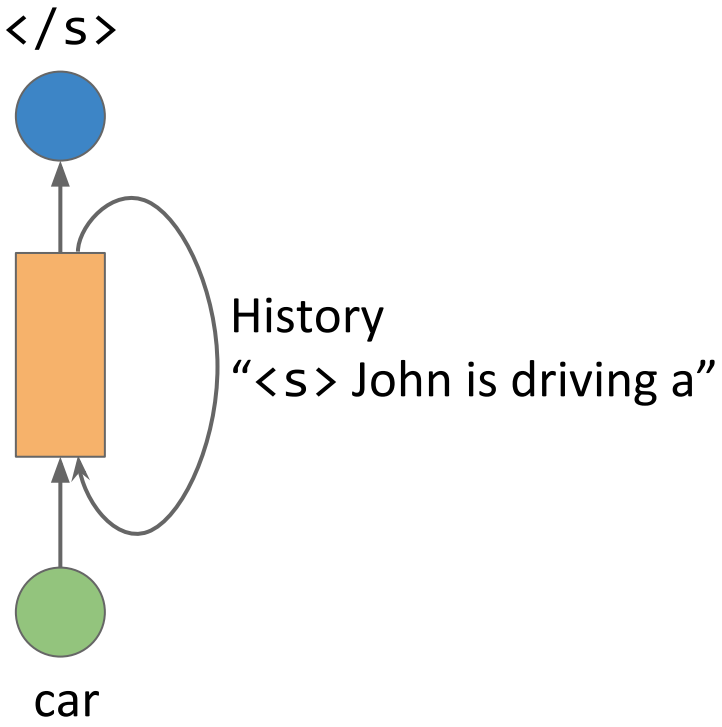
At the last timestep, the hidden state will have information about the entire sentence: **“John is driving a”** from history and **“car”** from the input



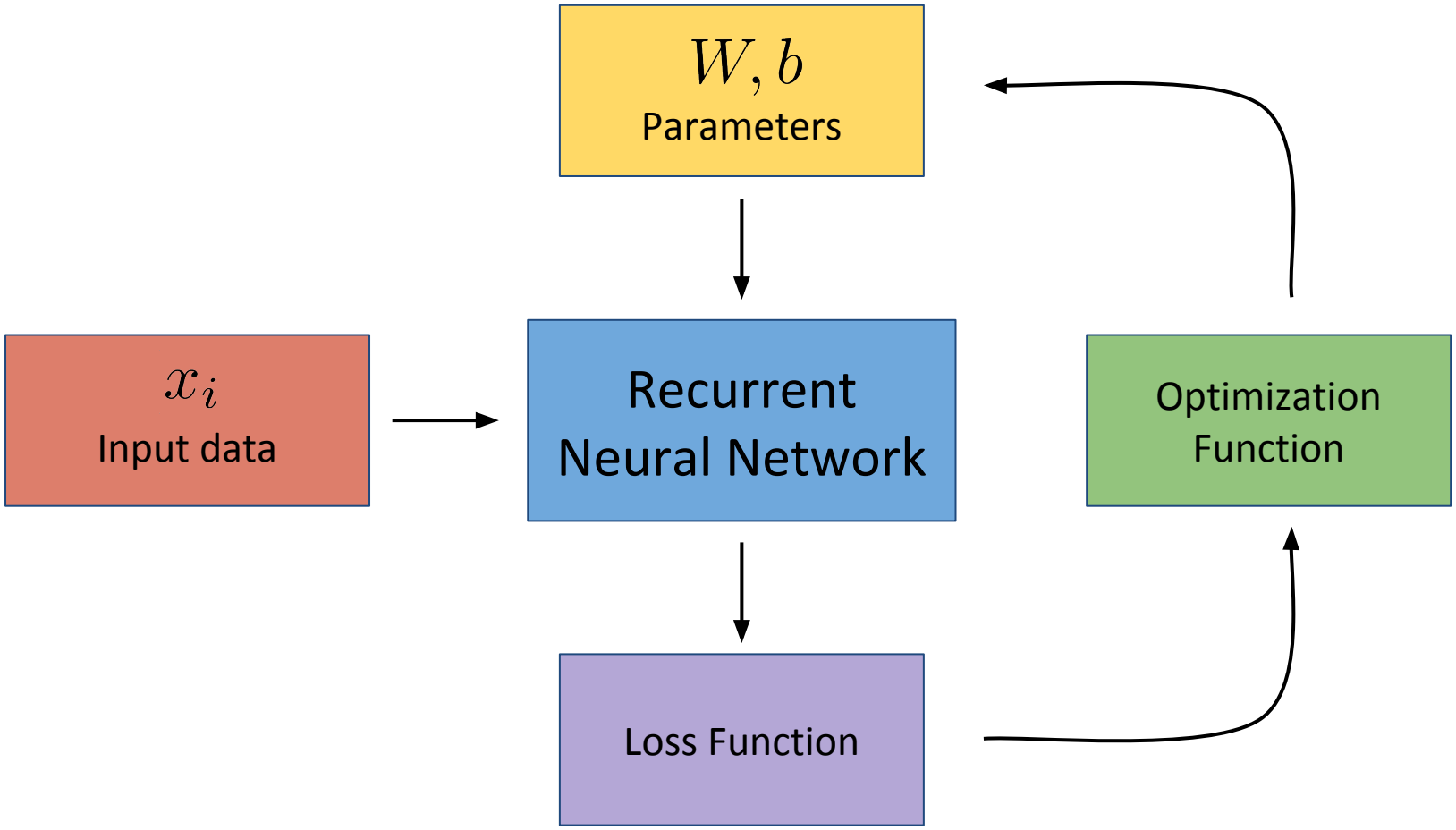
Recurrent Neural Network

Consider an example: <s> John is driving a car </s>

This hidden state can be considered as a "summary" of the entire sentence represented as a vector



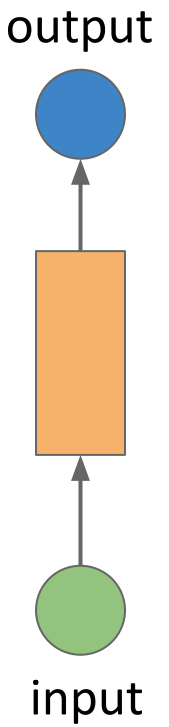
Recap


W, b
Parameters
x_i
Input data
Recurrent
Neural Network
Optimization
Function
Loss Function

Loss computation

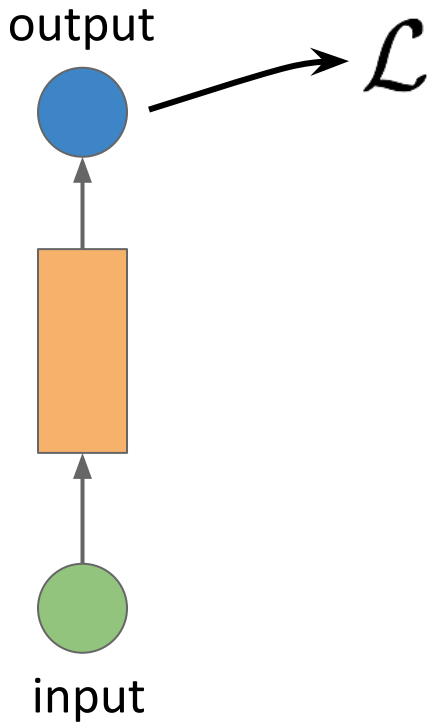
in recurrent neural networks

Loss Computation



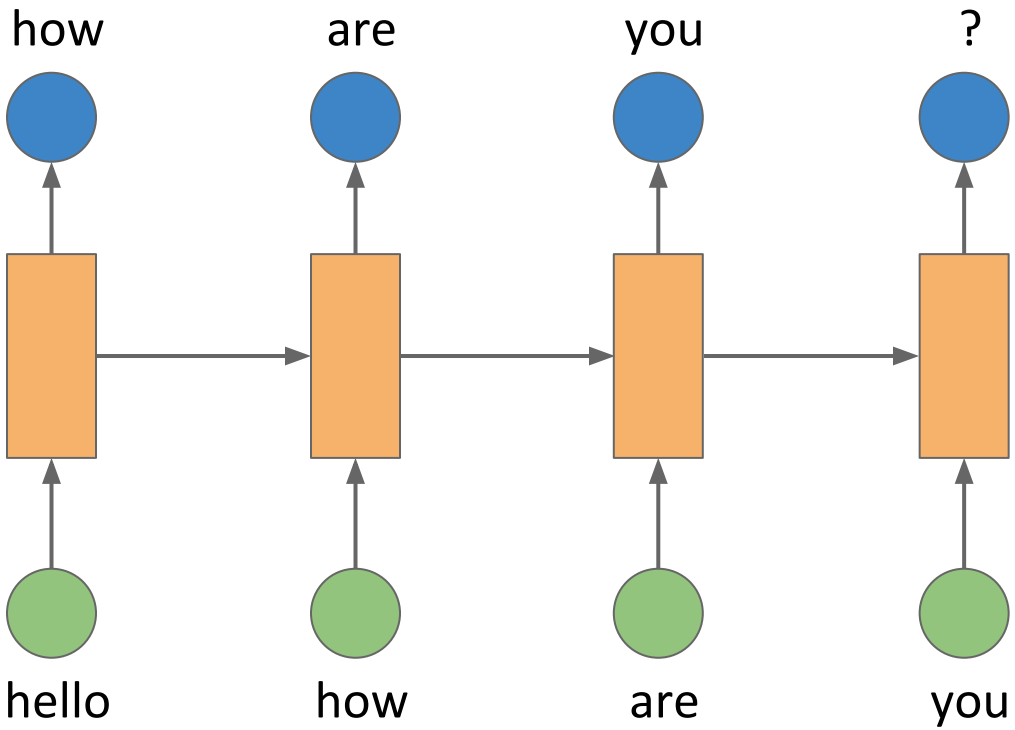
Recall that in a feed forward network, we have a single output

Loss Computation



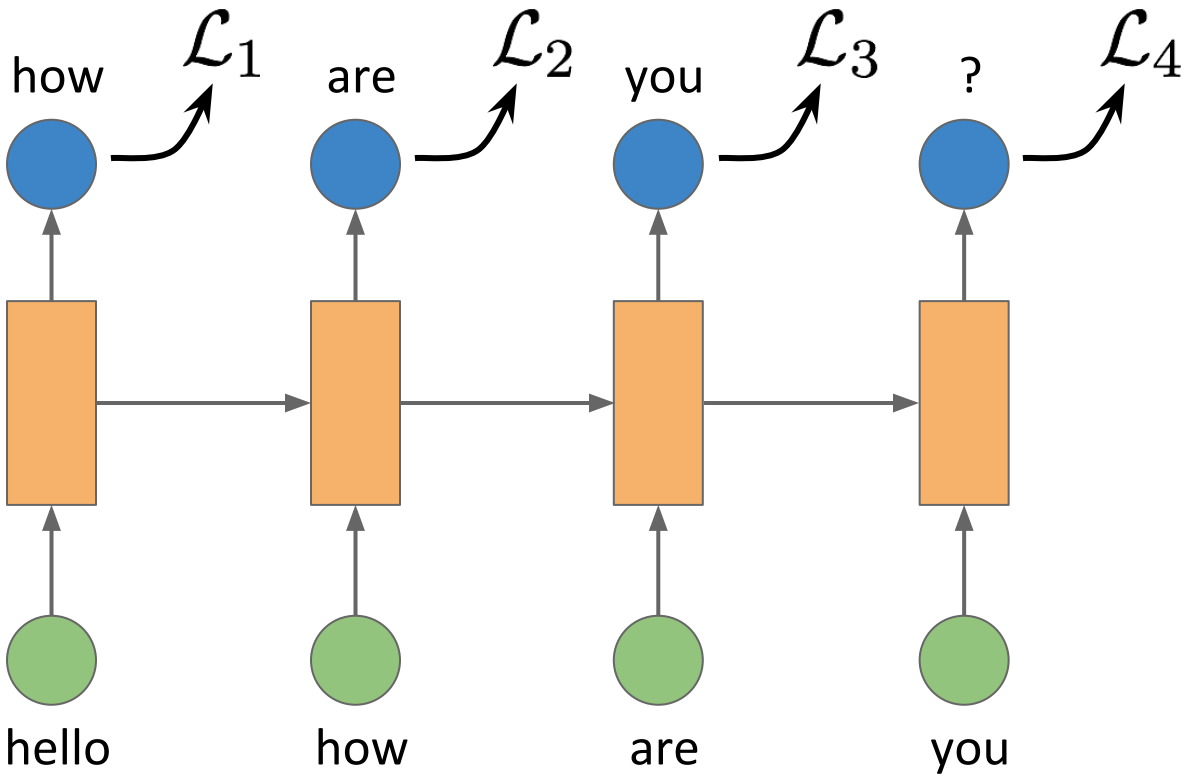
We compare this single output with the true label to get a loss value

Loss Computation



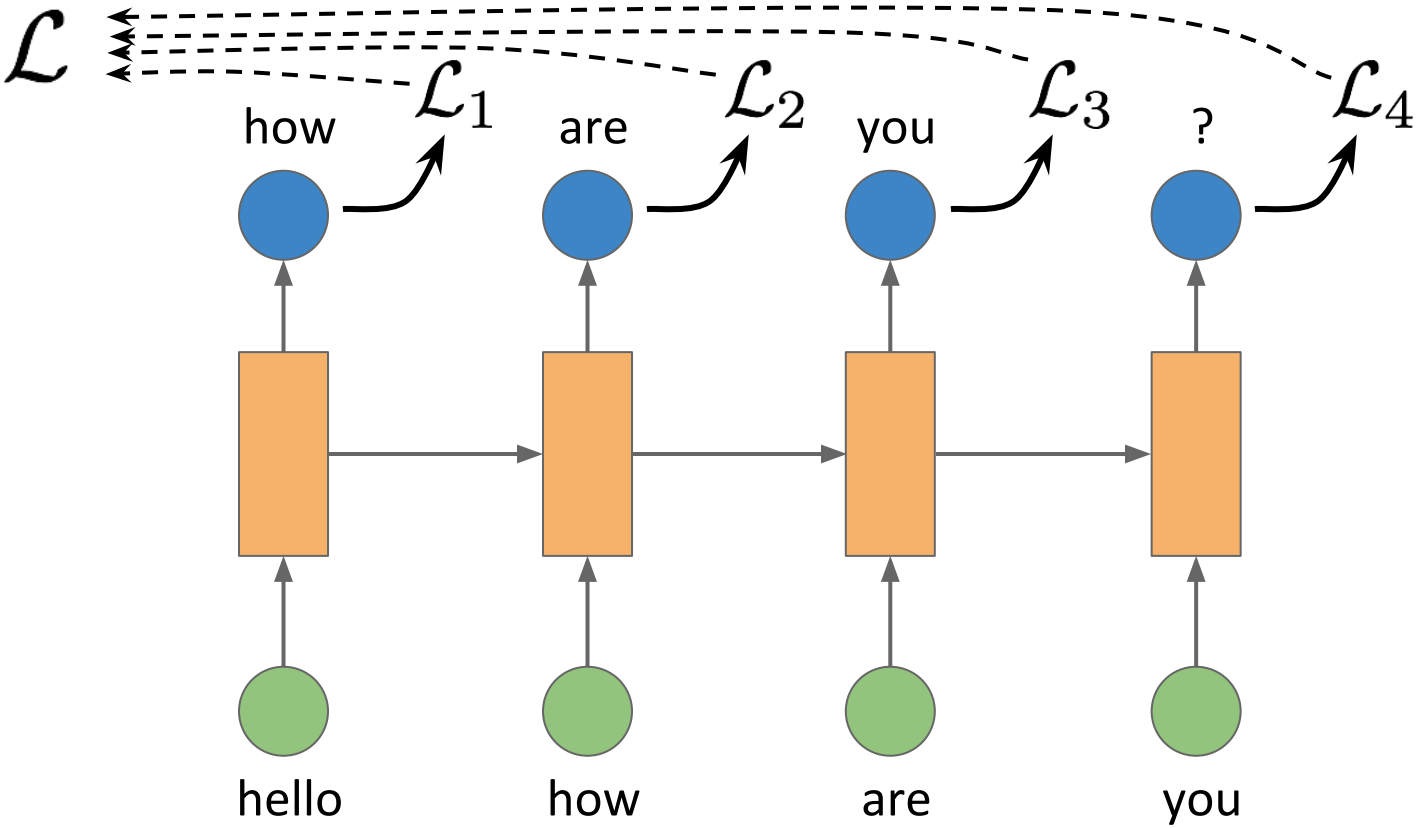
In the case of recurrent neural networks, we have an output per **timestep**

Loss Computation



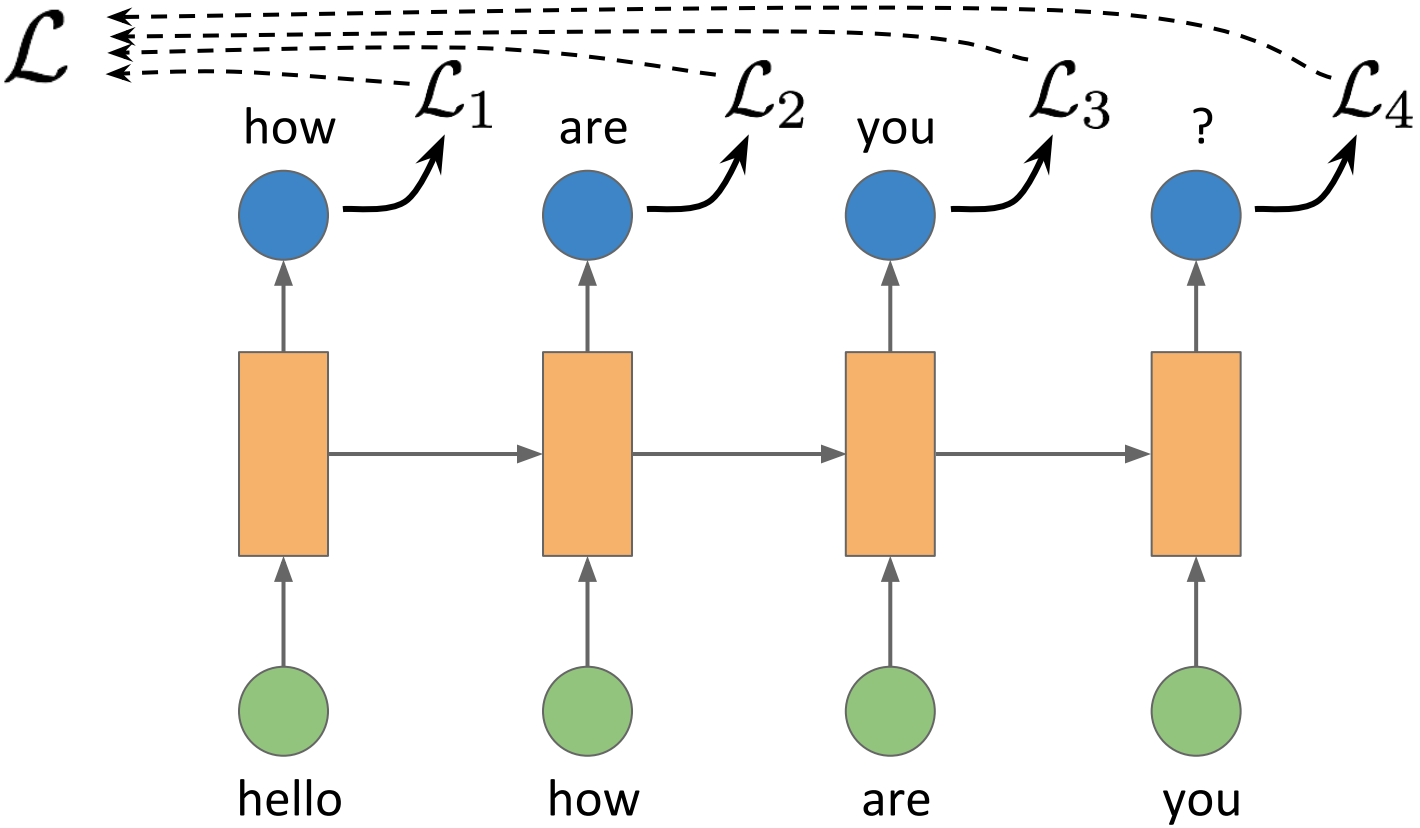
Each of these outputs can be used to get **one loss per timestep**

Loss Computation



We *add* *all of these losses* together to get a single loss for our optimization algorithm

Loss Computation



Individual losses are still calculated as before - e.g. using cross entropy loss

Practical Session

Lecture 5 - SimpleRNN1

Issues with vanilla RNN

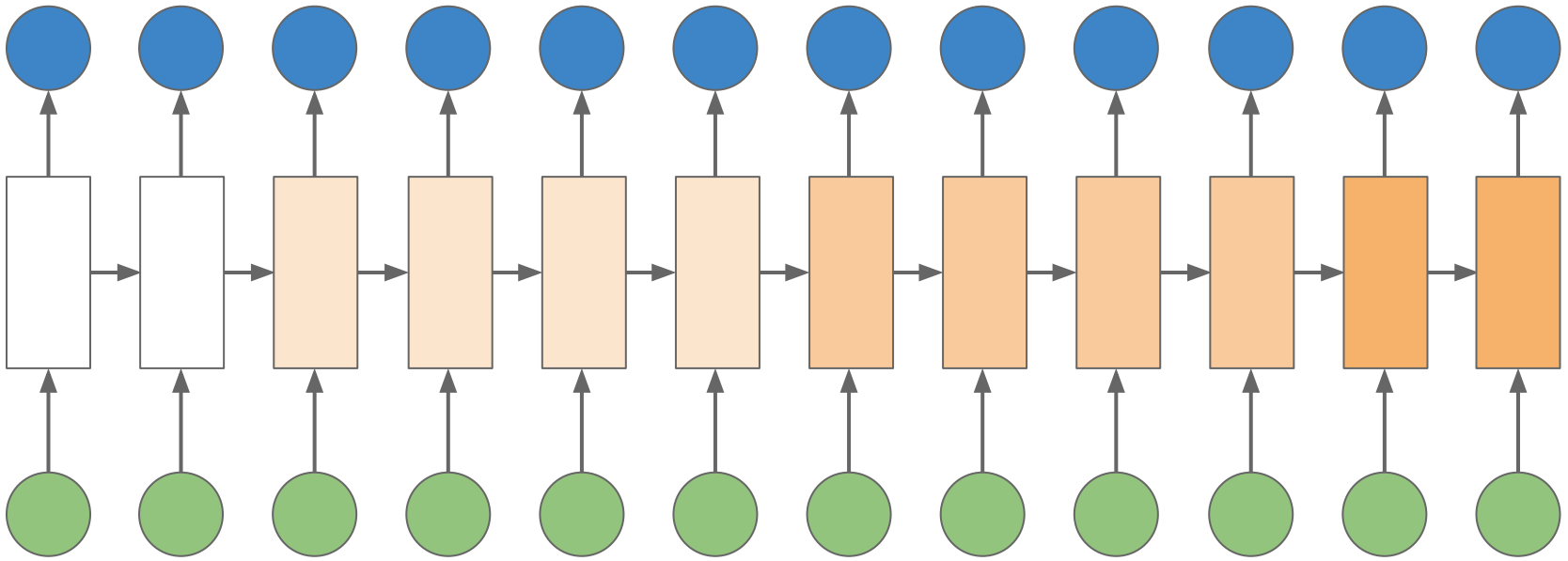
Issues with Vanilla RNN

- Information decay
  - long-term dependencies
- Vanishing gradients
- Exploding gradients

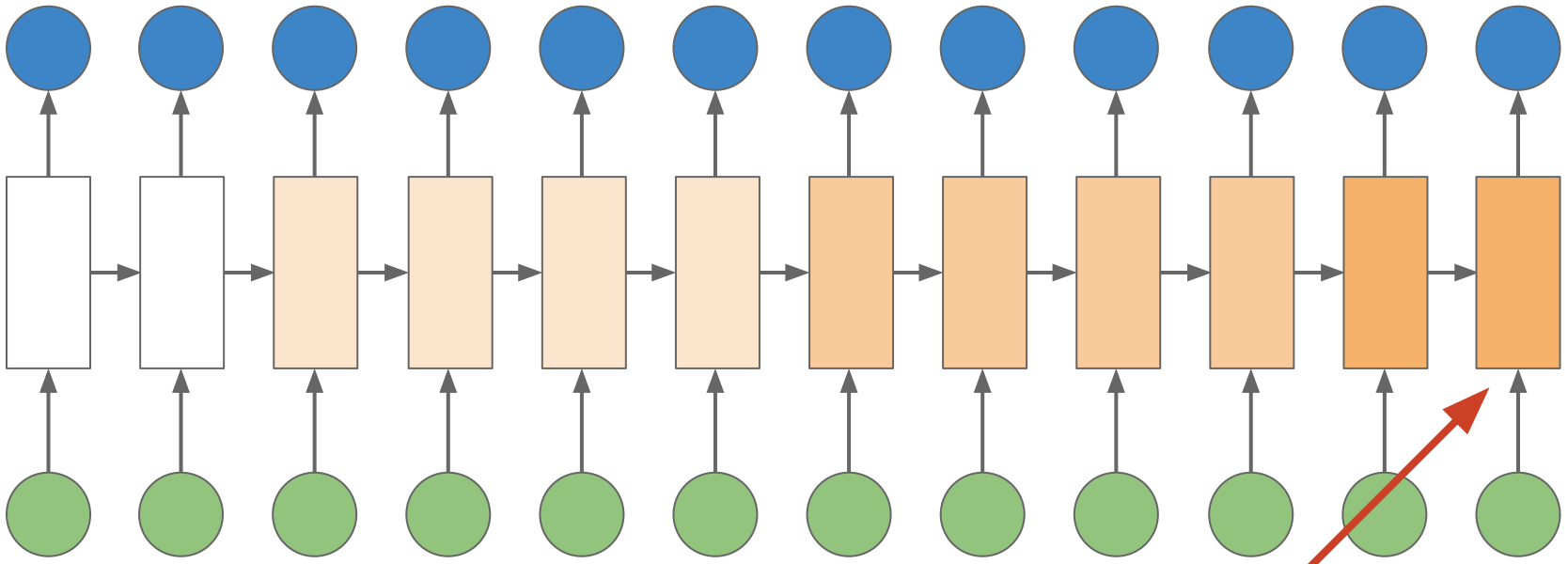
Issues with Vanilla RNN

- Information decay
  - long-term dependencies
- Vanishing gradients
- Exploding gradients

Information decay

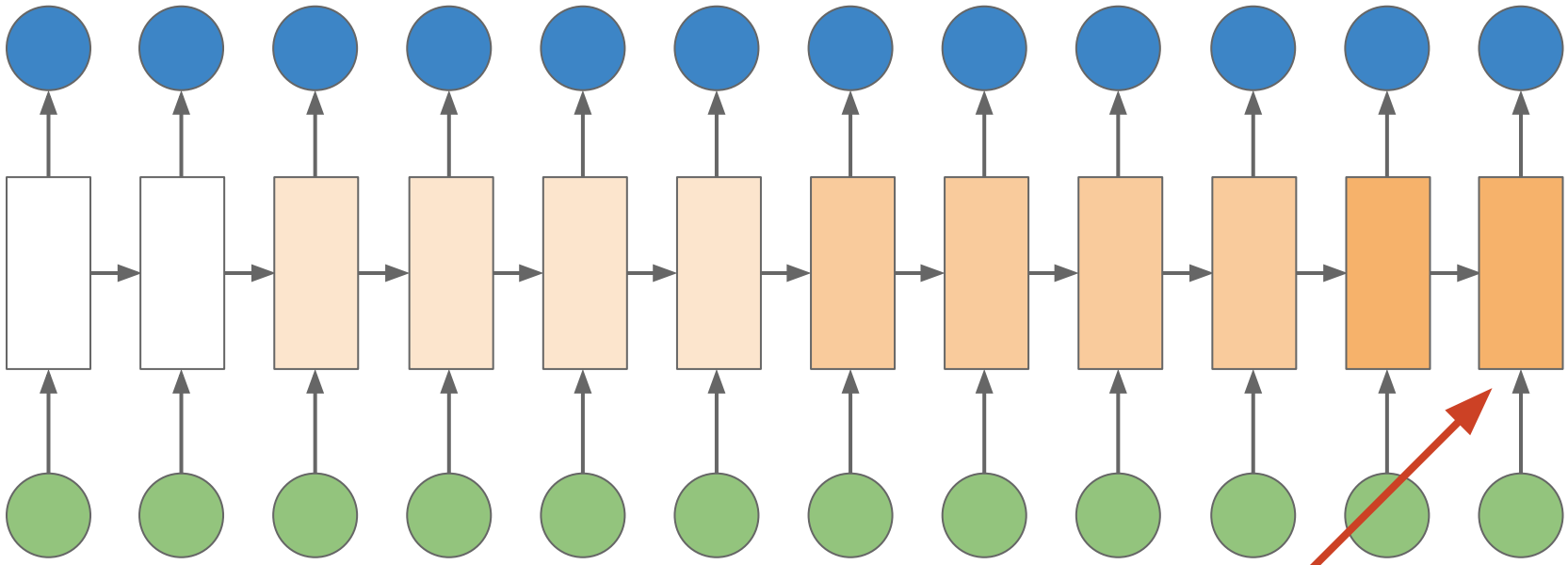


Information decay



The last timestep remembers very little about older timesteps, since it needs to remember information from recent history and the current timestep

Information decay



Remember, the output of the hidden layer is a **fixed length vector**

The network can only remember a limited amount of total information!

Long-term Dependencies

In theory, RNNs are capable of remembering long distance information

Practically, they start forgetting information over long distances as we have seen with the information decay problem

Long-term Dependencies

Words can have long-term dependency on previous words

Ansehen

Anna ***sieht*** sich die Talkshow **an**

If the distance between “an” and “sieht” becomes long, the RNN may forget to correctly learn the relationship

Issues with Vanilla RNN

- Information decay
  - long-term dependencies
- Vanishing gradients
- Exploding gradients

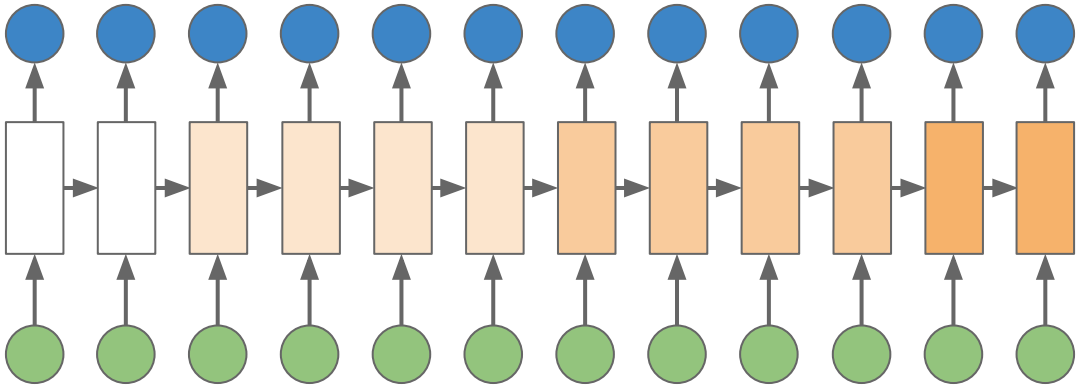
One potential solution:

Long short-term memory (LSTM)

Intuition

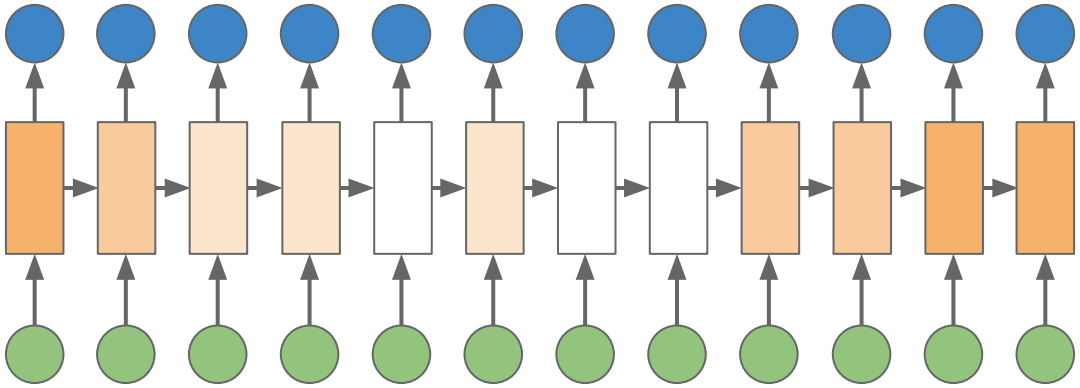
- In real world scenarios with inherited sequence properties, relevant information over long distances is vital
- For example, for an RNN to describe a movie scene, it would need to remember relevant information over longer sequences to describe the current scene correctly

Intuition



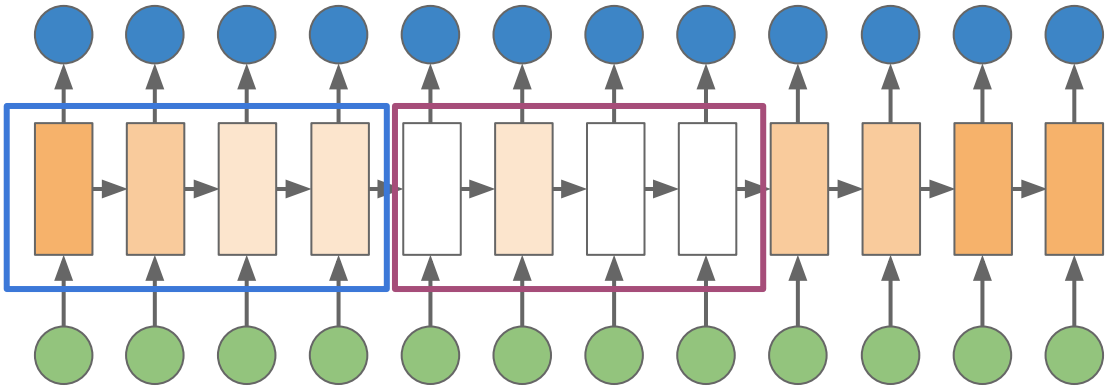
What if at each timestep, we can choose some information to be “important” and tell the network to remember it for longer?

Intuition



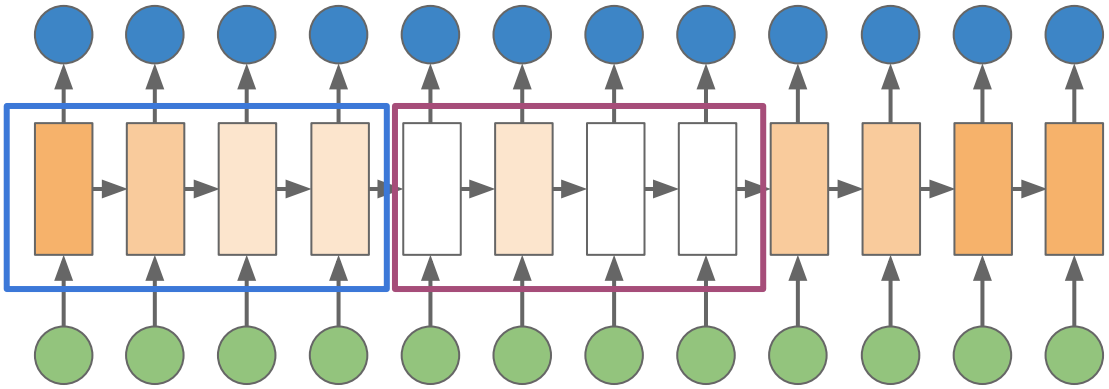
What if at each timestep, we can choose some information to be “important” and tell the network to remember it for longer?

Intuition



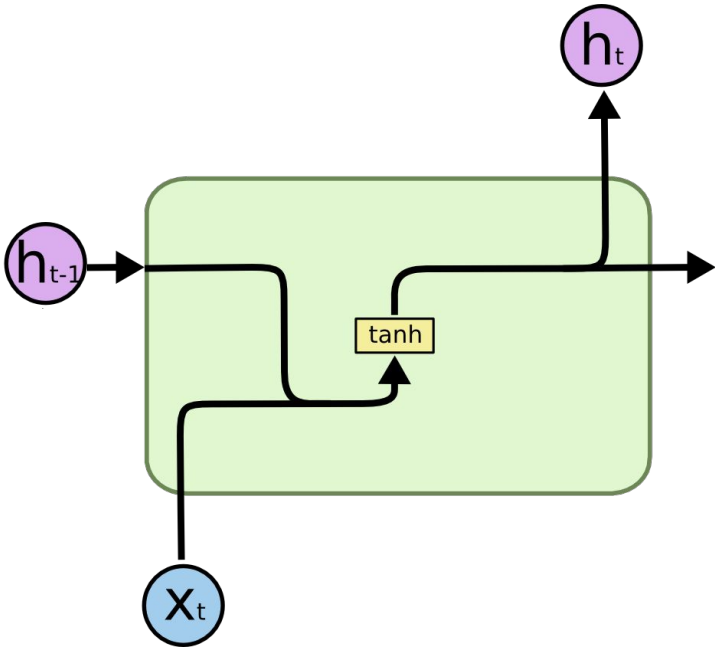
What if at each timestep, we can choose some information to be "important" and tell the network to remember it for longer?

Long Short-term Memory

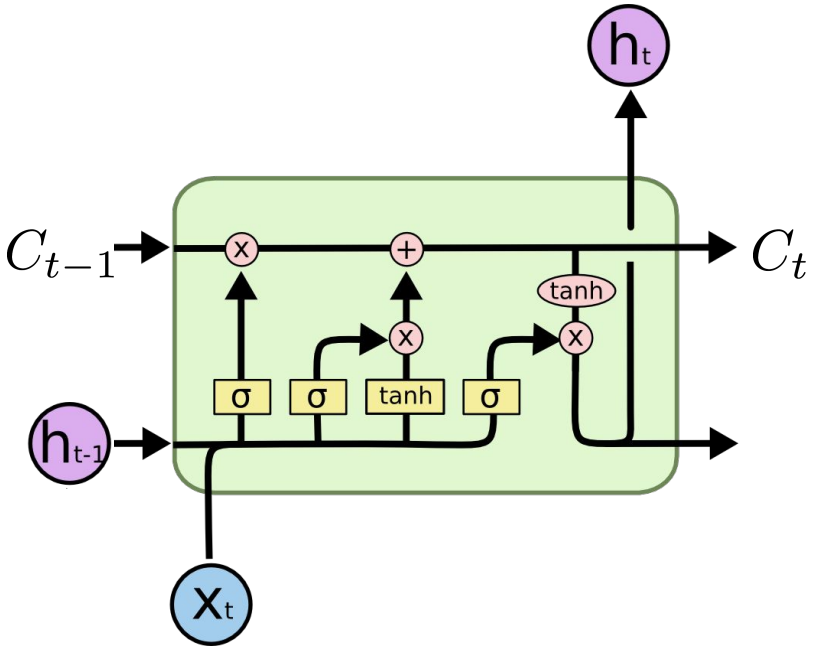


This is what **Long Short-term Memory** units do!

Long Short-term Memory



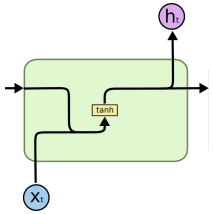
Recurrent Unit

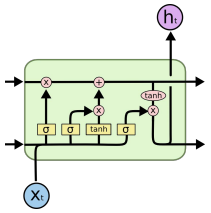


LSTM Unit

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long Short-term Memory





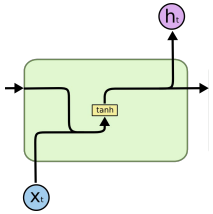
$$h_t = \tanh\left(Wx + W_h h_{t-1} + b\right)$$

Recurrent Unit

LSTM Unit

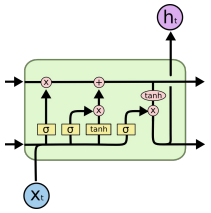
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long Short-term Memory



$$h_t = \tanh\left(Wx + W_h h_{t-1} + b\right)$$

Recurrent Unit



$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$
$$\widetilde{C_t} = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$
$$C_t = f_t \cdot C_{t-1} + i_t \cdot \widetilde{C_t}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t \cdot \tanh(C_t)$$

LSTM Unit

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long Short-term Memory

Consider an example: We are building a language model over some text that has several different people.

Alice studies computational linguistics. She is currently learning about LSTM's. Bob on the other hand studies about cyber security. He is completely confused right now!

Long Short-term Memory

Consider an example: We are building a language model over some text that has several different people.

Alice studies computational linguistics. **She** is currently learning about LSTM's. Bob on the other hand studies about cyber security. He is completely confused right now!

Long Short-term Memory

Consider an example: We are building a language model over some text that has several different people.

Alice studies computational linguistics. She is currently learning about LSTM's. **Bob** on the other hand studies about cyber security. **He** is completely confused right now!

Long Short-term Memory

Alice studies computational linguistics. She is currently learning about LSTM's. Bob on the other hand studies about cyber security. He is completely confused right now!

Our LSTM see's the above text word-by-word, so it needs to remember who we are talking about to use the correct pronouns

Long Short-term Memory

Alice studies computational linguistics. She is currently learning about LSTM's. Bob on the other hand studies about cyber security. He is completely confused right now!

LSTM unit sees **"Alice"** - and from the embeddings it knows that we are talking about a female person

Memory
Cell

Long Short-term Memory

Alice studies computational linguistics. She is currently learning about LSTM's. Bob on the other hand studies about cyber security. He is completely confused right now!

LSTM unit sees **"Alice"** - and from the embeddings it knows that we are talking about a female person

Let's add this information to our **memory cell**!

Memory
Cell

Long Short-term Memory

Alice studies computational linguistics. She is currently learning about LSTM's. Bob on the other hand studies about cyber security. He is completely confused right now!

LSTM unit see's other words, and decides what information should go into the memory cell

Memory Cell

Long Short-term Memory

Alice studies computational linguistics. She is currently learning about LSTM's. Bob on the other hand studies about cyber security. He is completely confused right now!

LSTM unit see's other words, and decides what information should go into the memory cell

Memory Cell

Long Short-term Memory

Alice studies computational linguistics. She is currently learning about LSTM's. Bob on the other hand studies about cyber security. He is completely confused right now!

LSTM unit now has to predict **"She"**. It does so by looking into the memory cell to identify which person we are talking about!

Memory Cell

Long Short-term Memory

Alice studies computational linguistics. She is currently learning about LSTM's. Bob on the other hand studies about cyber security. He is completely confused right now!

LSTM unit see's other words, and decides what information should go into the memory cell

Memory Cell

Long Short-term Memory

Alice studies computational linguistics. She is currently learning about LSTM's. Bob on the other hand studies about cyber security. He is completely confused right now!

LSTM unit see's other words, and decides what information should go into the memory cell

Memory Cell

Long Short-term Memory

Alice studies computational linguistics. She is currently learning about LSTM's. Bob on the other hand studies about cyber security. He is completely confused right now!

LSTM unit see's other words, and decides what information should go into the memory cell

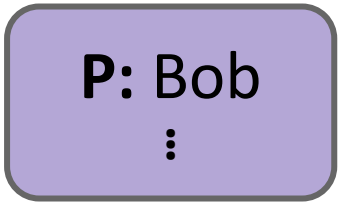
Memory Cell

Long Short-term Memory

Alice studies computational linguistics. She is currently learning about LSTM's. Bob on the other hand studies about cyber security. He is completely confused right now!

LSTM unit sees **"Bob"** - and from the embeddings it knows that we are talking about a male person

Lets *update* this information in our **memory cell**!

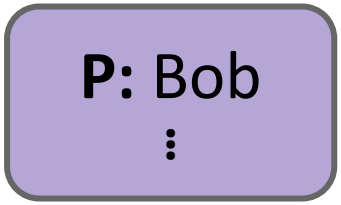


Memory Cell

Long Short-term Memory

Alice studies computational linguistics. She is currently learning about LSTM's. Bob on the other hand studies about cyber security. He is completely confused right now!

LSTM unit now has to predict **"He"**. Again, it does so by looking into the memory cell to identify which person we are talking about!



Memory Cell

Long Short-term Memory

Intuition: We have a **“memory cell”** or **“cell state”** that is passed along the time steps. At each timestep, the unit decides to forget some information from this **cell** and add some new information from the current input!

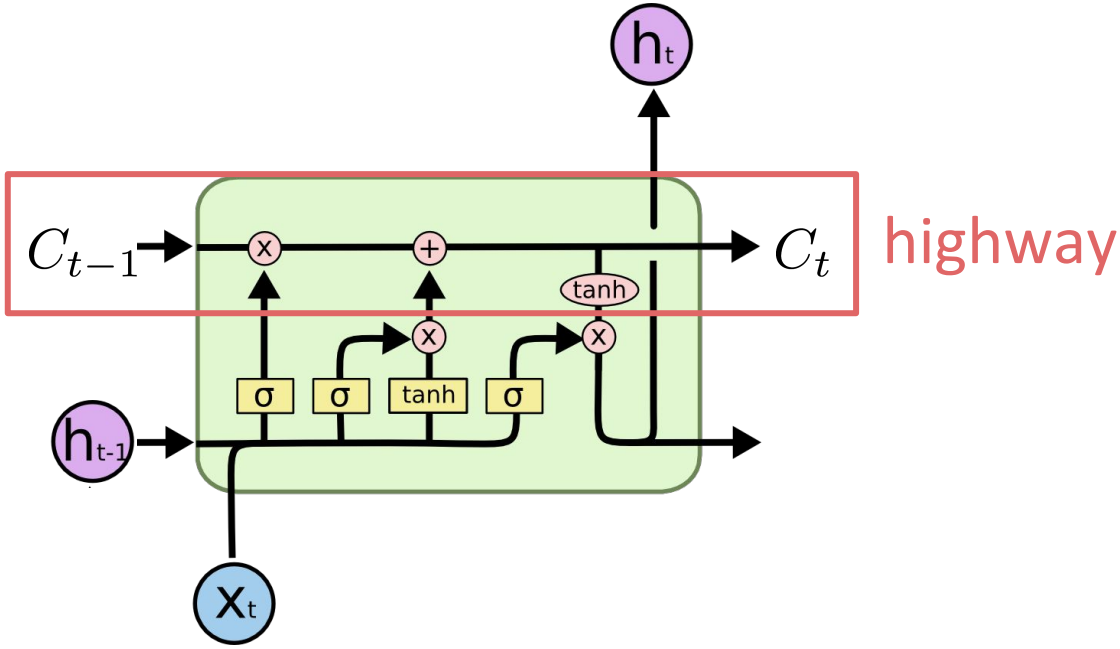
Long Short-term Memory

Intuition: We have a **“memory cell”** or **“cell state”** that is passed along the time steps. At each timestep, the unit decides to forget some information from this **cell** and add some new information from the current input!

This effectively helps us solve both the **Information decay** and the **vanishing gradient** problem

Long Short-term Memory

This effectively helps us solve the **Information decay** and potentially reduces the **vanishing/exploding gradient** problem



LSTM demonstrations

Long Short-term Memory

LSTM’s are super general purpose - you can use them on any kind of sequential data that can be represented as some vectors

Long Short-term Memory

LSTM's are super general purpose - you can use them on any kind of sequential data that can be represented as some vectors

Andrej Karpathy has some really nice demos on character-level language models, i.e. the input to the network is the **next character** instead of a word

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

LSTM Evolution

Here, we will see an network using LSTM units *evolve* over time - remember, all we are doing here is asking the network to predict the next character given the history of characters

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

LSTM Evolution

Here, we will see an network using LSTM units *evolve* over time - remember, all we are doing here is asking the network to predict the next character given the history of characters

qewjhhcvuh rkjghqruhgqrgnqrhlhqlrqrjlcznmcyaklm
adjfadhoirqjnrm, aghouihr;qnrjnjn agyqeg
cvz,cmnv;lhruhm,.nm,czbvugrablgjn,.m adnadfnalkd

Iteration 0

Initially, the output is complete garbage!

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

LSTM Evolution

Here, we will see an network using LSTM units *evolve* over time - remember, all we are doing here is asking the network to predict the next character given the history of characters

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

Iteration 100

A few iterations later - still garbage, but it is starting to learn the concept of “words” and “spaces”

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

LSTM Evolution

Here, we will see an network using LSTM units *evolve* over time - remember, all we are doing here is asking the network to predict the next character given the history of characters

"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."

Iteration 300

The model is now learning about periods and quotes.

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

LSTM Evolution

Here, we will see an network using LSTM units *evolve* over time - remember, all we are doing here is asking the network to predict the next character given the history of characters

> we counter. He stutn co des. His stanted out one ofler that concossions and was to gearang reay Jotrets and with fre colt otf paitt thin wall. Which das stimn

Iteration 500

Some simple words like “We”, “He”, “His” are spelt correctly!

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

LSTM Evolution

Here, we will see an network using LSTM units *evolve* over time - remember, all we are doing here is asking the network to predict the next character given the history of characters

> 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.

Iteration 700

Some structure of English is starting to appear...

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

LSTM Evolution

Here, we will see an network using LSTM units *evolve* over time - remember, all we are doing here is asking the network to predict the next character given the history of characters

"Kite vouch!" he repeated by her
door. "But I would be done and quarts, feeling, then, son is people...."

Iteration 1200

The model has learned longer words and some punctuation

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

LSTM Evolution

Here, we will see an network using LSTM units *evolve* over time - remember, all we are doing here is asking the network to predict the next character given the history of characters

> "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.

Iteration 2000

Much better outputs than what we started with...

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

LSTM Evolution

Here, we will see a network using LSTM units *evolve* over time - remember, all we are doing here is asking the network to predict the next character given the history of characters

> "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.

Iteration 2000

Much better outputs than what we started with...

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

LSTM Examples

A model trained on Shakespeare

PANDARUS:
Alas, I think he shall be come approached and the day
When little srain 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.

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

LSTM Examples

A model trained on Wikipedia sources

Naturalism and decision for the majority of Arab countries' capitalide was grounded by the Irish language by **[[John Clair]]**, **[[An Imperial Japanese Revolt]]**, associated with Guangzham's sovereignty. His generals were the powerful ruler of the Portugal in the [[Protestant Immineners]], which could be said to be directly in Cantonese Communication, which followed a ceremony and set inspired prison, training. The emperor travelled back to [[Antioch, Perth, October 25|21]] to note, the Kingdom of Costa Rica, unsuccessful fashioned the [[Thrales]], [[Cynth's Dajoard]], known in western [[Scotland]], near Italy to the conquest of India with the conflict. Many governments recognize the military housing of the [[Civil Liberalization and Infantry Resolution 265 National Party in Hungary]], that is sympathetic to be to the [[Punjab Resolution]] (PJS) [**http://www.humah.yahoo.com/guardian.cfm/7754800786d17551963s89.htm** Official economics Adjoint for the Nazism, Montgomery was swear to advance to the resources for those Socialism's rule, was starting to signing a major tripad of aid exile.]]

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

LSTM Examples

A book on Algebraic geometry!

For $\bigoplus_{n=1,\ldots,m}$ where $\mathcal{L}_{m_\bullet} = 0$, hence we can find a closed subset $\mathcal{H}$ in $\mathcal{H}$ and any sets $\mathcal{F}$ on X, U is a closed immersion of S, then $U \to T$ is a separated algebraic space.

Proof. Proof of (1). It also start we get

$$S = \mathrm{Spec}(R) = U \times_X U \times_X U$$

and the compariciely in the fibre product covering we have to prove the lemma generated by $\coprod Z \times_U U \to V$. Consider the maps M along the set of points Sch_{fppf} and $U \to U$ 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 $\mathrm{Spec}(R') \to S$ is smooth or an

$$U = \bigcup U_i \times_{S_i} U_i$$

which has a nonzero morphism we may assume that f_i is of finite presentation over S. We claim that $\mathcal{O}_{X,x}$ is a scheme where $x, x', s'' \in S'$ such that $\mathcal{O}_{X,x'} \to \mathcal{O}'_{X',x'}$ is separated. By Algebra, Lemma **??** we can define a map of complexes $\mathrm{GL}_{S'}(x'/S'')$ and we win. □

To prove study we see that $\mathcal{F}|_U$ is a covering of $\mathcal{X}'$, and $\mathcal{T}_i$ is an object of $\mathcal{F}_{X/S}$ for $i > 0$ and $\mathcal{F}_p$ exists and let $\mathcal{F}_i$ be a presheaf of $\mathcal{O}_X$-modules on $\mathcal{C}$ as a $\mathcal{F}$-module. In particular $\mathcal{F} = U/\mathcal{F}$ we have to show that

$$\widetilde{M}^\bullet = \mathcal{I}^\bullet \otimes_{\mathrm{Spec}(k)} \mathcal{O}_{S,s} - i_X^{-1}\mathcal{F})$$

is a unique morphism of algebraic stacks. Note that

$$\mathrm{Arrows} = (Sch/S)^{opp}_{fppf}, (Sch/S)_{fppf}$$

and

$$V = \Gamma(S, \mathcal{O}) \longmapsto (U, \mathrm{Spec}(A))$$

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 $X_{spaces,étale}$ 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* $\underline{Proj}_X(\mathcal{A}) = \mathrm{Spec}(B)$ *over* U *compatible with the complex*

$$Set(\mathcal{A}) = \Gamma(X, \mathcal{O}_{X,\mathcal{O}_X}).$$

When in this case of to show that $\mathcal{Q} \to \mathcal{C}_{Z/X}$ *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 closed subschemes are catenary. If* T *is surjective we may assume that* T *is connected with residue fields of* S*. Moreover there exists a closed subspace* $Z \subset X$ *of* X *where* U *in* X' *is proper (some defining as a closed subset of the uniqueness it suffices to check the fact that the following theorem*

(1) f *is locally of finite type. Since* $S = \mathrm{Spec}(R)$ *and* $Y = \mathrm{Spec}(R)$.

Proof. This is form all sheaves of sheaves on X. But given a scheme U and a surjective étale morphism $U \to X$. Let $U \cap U = \coprod_{i=1,\ldots,n} U_i$ be the scheme X over S at the schemes $X_i \to X$ and $U = \lim_i X_i$. □

The following lemma surjective restrocomposes of this implies that $\mathcal{F}_{x_0} = \mathcal{F}_{x_0} = \mathcal{F}_{\mathcal{X},\ldots,0}$.

Lemma 0.2. *Let* X *be a locally Noetherian scheme over* S, $E = \mathcal{F}_{X/S}$. *Set* $\mathcal{I} = \mathcal{J}_1 \subset \mathcal{I}'_n$. *Since* $\mathcal{I}^n \subset \mathcal{I}^n$ *are nonzero over* $i_0 \leq \mathfrak{p}$ *is a subset of* $\mathcal{J}_{n,0} \circ \overline{A}_2$ *works.*

Lemma 0.3. *In Situation* **??**. *Hence we may assume* $\mathfrak{q}' = 0$.

Proof. We will use the property we see that $\mathfrak{p}$ is the mext functor (**??**). On the other hand, by Lemma **??** we see that

$$D(\mathcal{O}_{X'}) = \mathcal{O}_X(D)$$

where K is an F-algebra where δ_{n+1} is a scheme over S. □

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

LSTM Examples

A model trained on Linux source code

```
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
  int error;
  if (fd == MARN_EPT) {
    /*
     * The kernel blank will coeld it to userspace.
     */
    if (ss->segment < mem_total)
      unblock_graph_and_set_blocked();
    else
      ret = 1;
    goto bail;
  }
  segaddr = in_SB(in.addr);
  selector = seg / 16;
  setup_works = true;
  for (i = 0; i < blocks; i++) {
    seq = buf[i++];
    bpf = bd->bd.next + i * search;
    if (fd) {
      current = blocked;
    }
  }
  rw->name = "Getjbbregs";
  bprm_self_clearl(&iv->version);
  regs->new = blocks[(BPF_STATS << info->historidac)] | PFMR_CLOBATHINC_SECONDS << 12;
  return segtable;
}
```

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Advantages and Disadvantages of RNN

Advantages

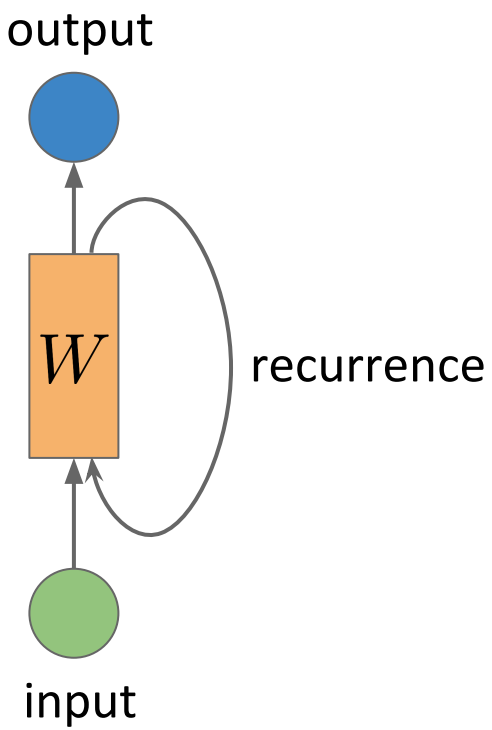
- Can process any input length
- Model size is independent of the sequence length

Disadvantages

- High Computation time

RNNs in Practice

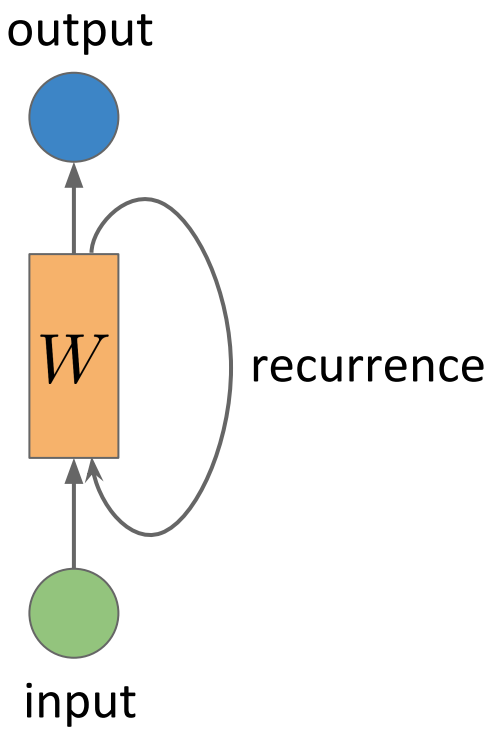
So far, we have only talked about the recurrent layer in our models



RNNs in Practice

However, recurrent layers are seldom used all by themselves - but instead are preceded by normal feed-forward layers for the input and outputs

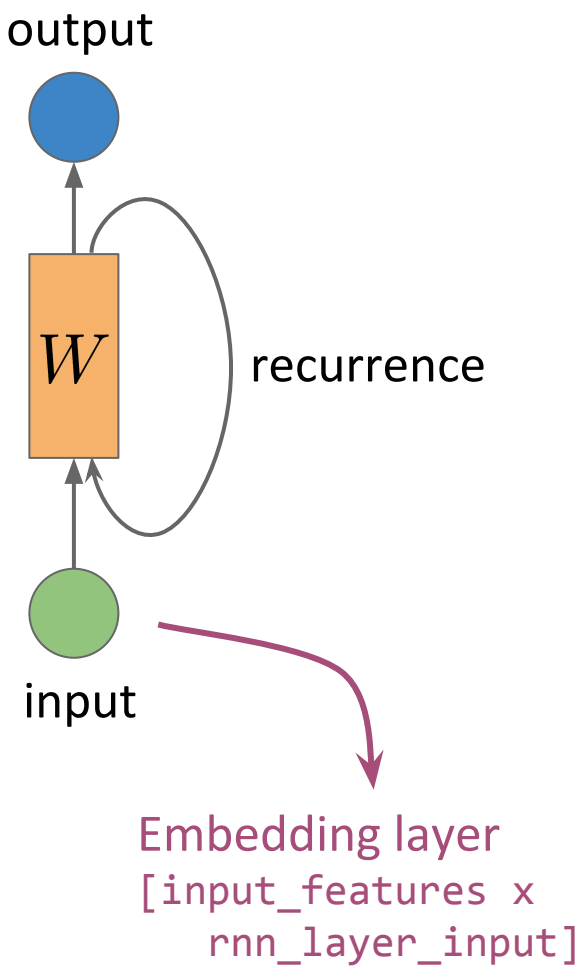
(represented as the green and blue circles here)



RNNs in Practice

However, recurrent layers are seldom used all by themselves - but instead are preceded by normal feed-forward layers for the input and outputs

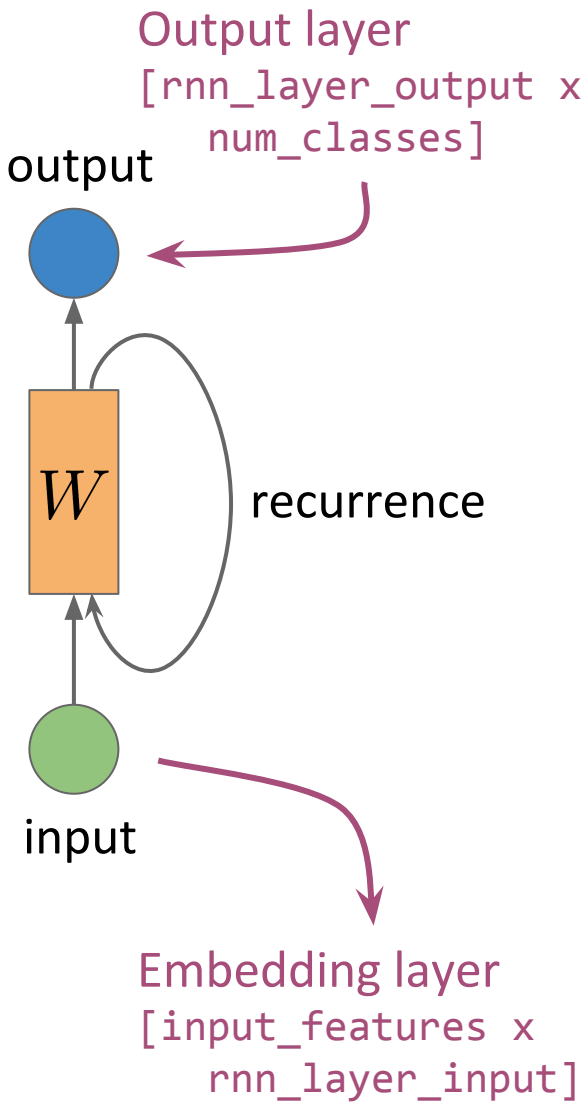
(represented as the green and blue circles here)



RNNs in Practice

However, recurrent layers are seldom used all by themselves - but instead are preceded by normal feed-forward layers for the input and outputs

(represented as the green and blue circles here)



Applications of RNNs

From now on, RNN refers to LSTM

Practical Session

 LSTM