Class core values

1. Be **respect**ful to yourself and others
2. Be **confident** and believe in yourself
3. Always do your **best**
4. Be **cooperative**
5. Be **creative**
6. Have **fun**
7. Be **patient** with yourself while you learn
8. Don’t be shy to **ask “stupid” questions**
9. Be **inclusive** and **accepting**

Week 5, Lecture 1

Learning on sequences: RNNs

Image from Priyal Walpita, medium

Learning Objectives

1. Describe the main challenges with sequence inputs
2. Explain the basic concepts of a recurrent neural network
3. Define the limitations of RNNs
4. Describe embedding and its biases
5. Apply keras to implement a simple RNN module
6. Tune the model based on knowledge of the concepts

Types of input data for proteins

1. Simple input

protein$_1$	25 kDa	pI=7.5	310 residues	…	2.5 hr half-life	Stability$_1$
protein$_2$	10 kDa	pI=4	50 residues	…	10 hr half-life	Stability$_2$
protein$_3$	100 kDa	pI=8	1200 residues	…	2 hr half-life	Stability$_3$

…

Types of input data for proteins

1. Simple input — SVM, Random Forest, dense neural net
2. 2D image — CNN

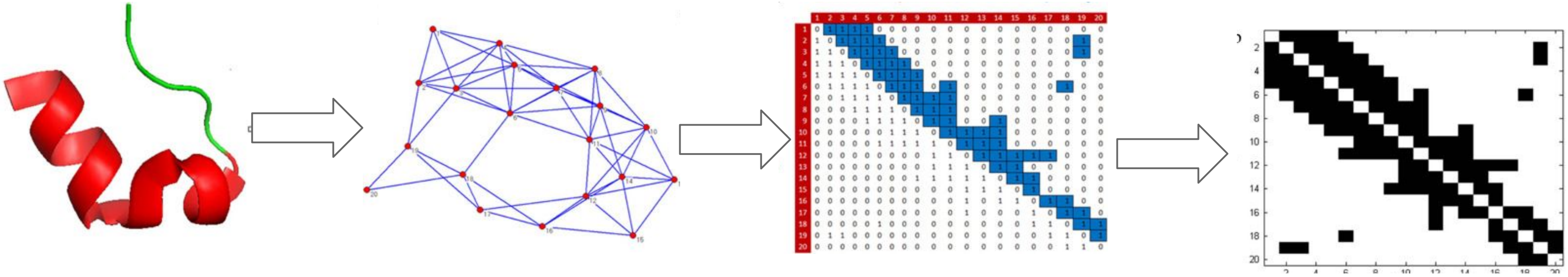


Image from *Physcia A*, **2017**, volume 456, 782-791

Types of input data for proteins

1. Simple input — SVM, Random Forest, dense neural net
2. 2D image — CNN
3. String of amino acids

p_1	MGLTDILGFNREFDILAV…SPLFG	s_1
p_2	MLKPTRVNMSERCGHITDENVCSR…TLVRF	s_2
p_3	MIKRTVIHGRDFRWNYTSPL…GMNSWQ	s_3
…		

Features: charge, pKa, size, functional groups, hydrogen bond status, …

Types of input data for proteins

1. Simple input — SVM, Random Forest, dense neural net
2. 2D image — CNN
3. String of amino acids — Natural language processing

p_1 MGLTDILGFNREFDILAV…SPLFG s_1

p_2 MLKPTRVNMSERCGHITDENVCSR…TLVRF s_2

p_3 MIKRTVIHGRDFRWNYTSPL…GMNSWQ s_3

…

Features: charge, pKa, size, functional groups, hydrogen bond status, …

Natural language processing, a big area in computer science

Understanding human language (spoken or written)

Natural language processing, a big area in computer science

Understanding human language (spoken or written)

- Speech recognition
  - ASR, speech to text
- Natural language understanding
  - Voice activation, commands to robots, text categorization
- Natural language generation
  - Generating forecasts, automated response

Computers don't understand words

Apple
Orange
Cow
Building
Scientist

One way to help computers understand words is by one-hot encoding

Apple	1000000
Orange	0100000
Cow	0010000
Building	0001000
Scientist	0000100

One-hot encoding doesn’t retain the relationship between words

Apple	1000000
Orange	0100000
Cow	0010000
Building	0001000
Scientist	0000100

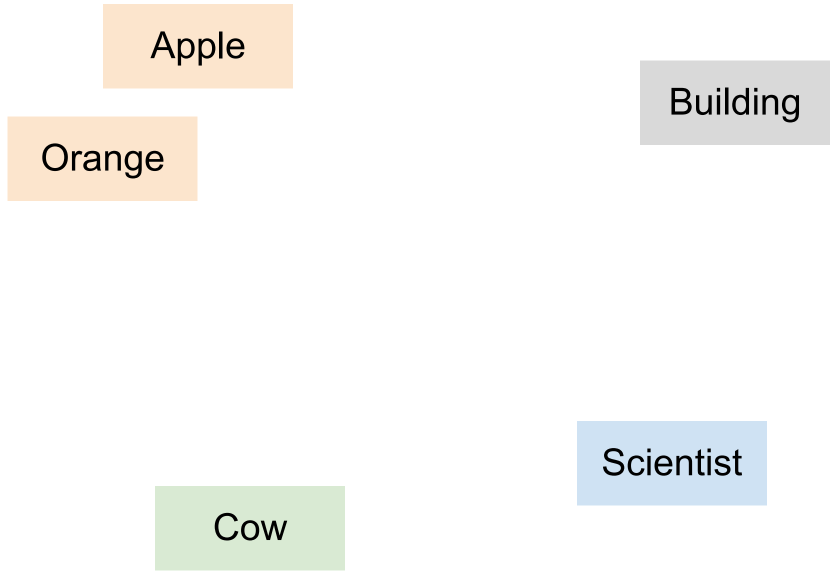
| Apple - Orange | = | Apple - Building |

One-hot encoding is not feasible for the many many words we have

Apple	1000000
Orange	0100000
Cow	0010000
Building	0001000
Scientist	0000100

The solution is word embedding

Apple	1000000
Orange	0100000
Cow	0010000
Building	0001000
Scientist	0000100



One way to get the embedding is by training on the fly

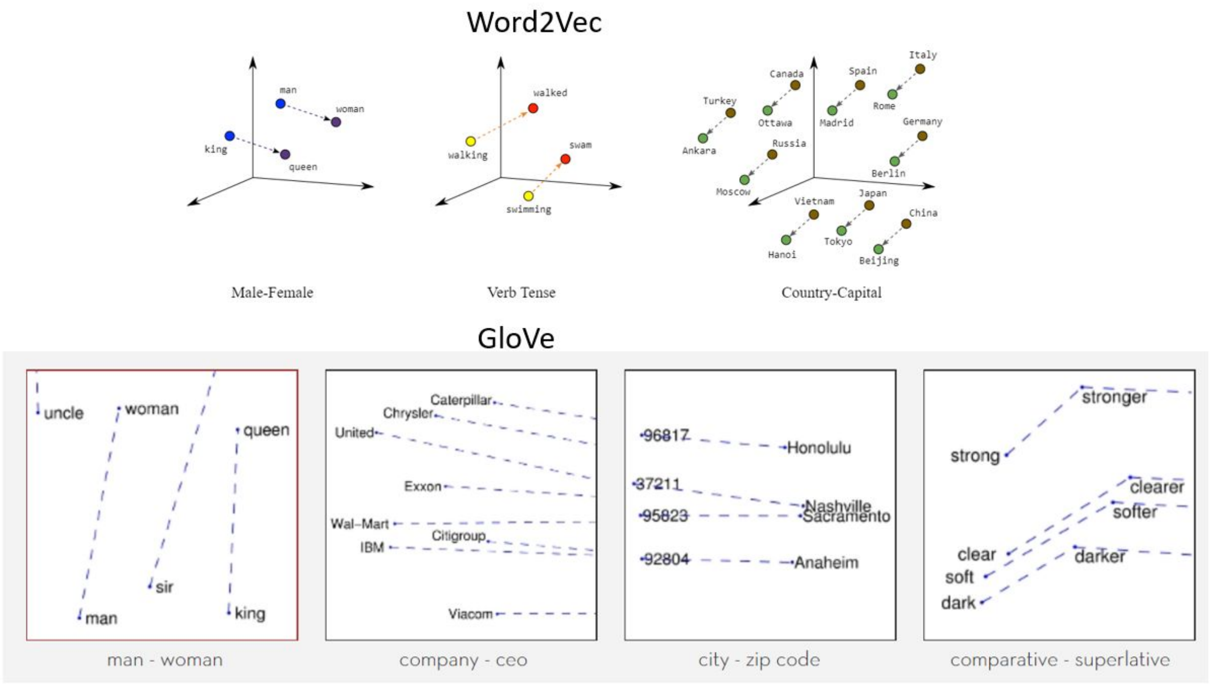
$$\underbrace{\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{One-hot vector}} \times \underbrace{\begin{bmatrix} 8 & 2 & 1 & 9 \\ 6 & 5 & 4 & 0 \\ 7 & 1 & 6 & 2 \\ 1 & 3 & 5 & 8 \\ 0 & 4 & 9 & 1 \end{bmatrix}}_{\text{Embedding Weight Matrix}} = \underbrace{\begin{bmatrix} 1 & 3 & 5 & 8 \end{bmatrix}}_{\text{Hidden layer output}}$$

You can also use a pre-trained embedding matrix – often a better solution

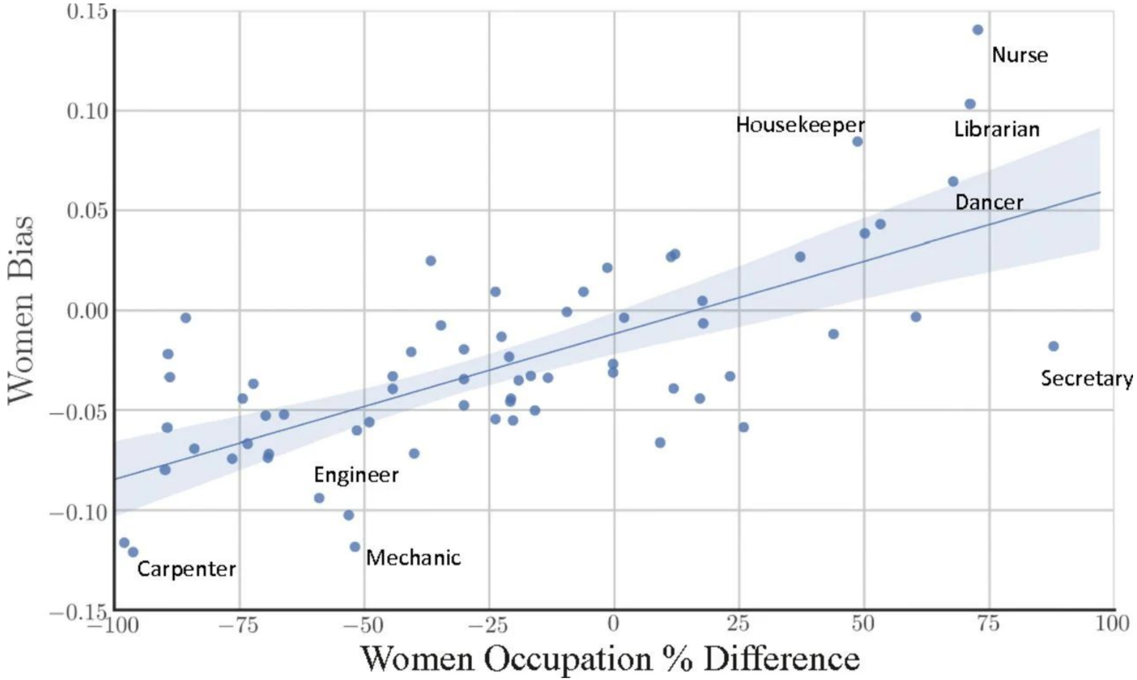
$$\underset{\text{One-hot vector}}{\begin{bmatrix}0 & 0 & 0 & 1 & 0\end{bmatrix}} \times \underset{\text{Embedding Weight Matrix}}{\begin{bmatrix}8 & 2 & 1 & 9\\ 6 & 5 & 4 & 0\\ 7 & 1 & 6 & 2\\ 1 & 3 & 5 & 8\\ 0 & 4 & 9 & 1\end{bmatrix}} = \underset{\text{Hidden layer output}}{\begin{bmatrix}1 & 3 & 5 & 8\end{bmatrix}}$$

Image from Hemant Ranvir, medium

Word2vec and GloVe are two best known methods for creating word embedding



Training on existing corpus of text carries over the biases we have

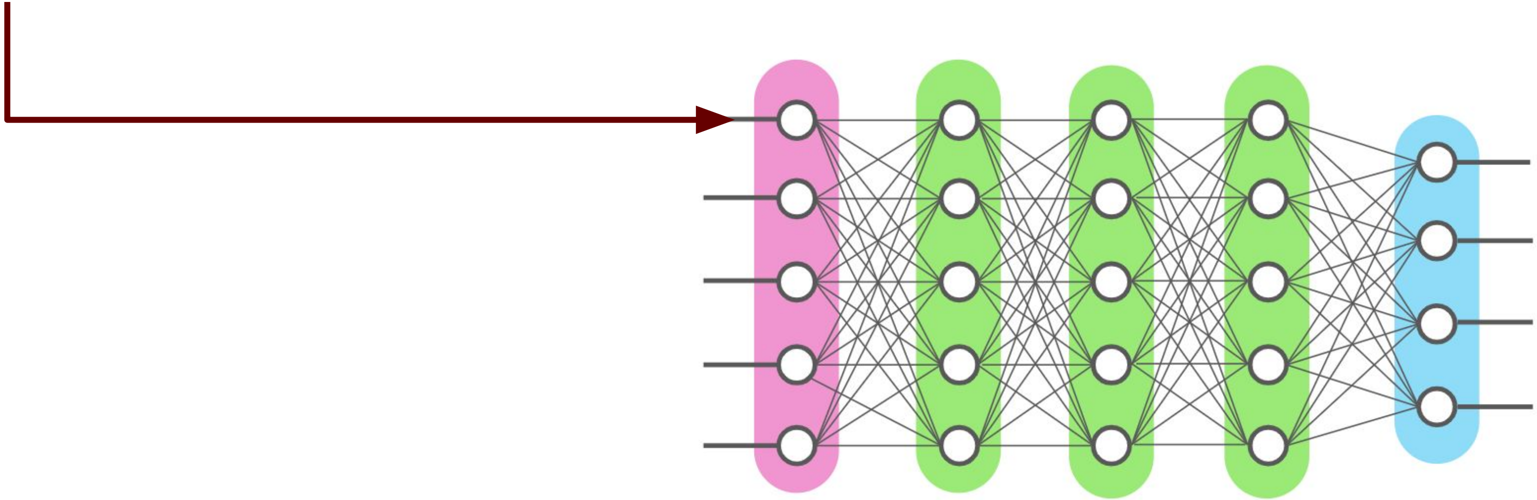


Language as input to a neural net

The weather is great .

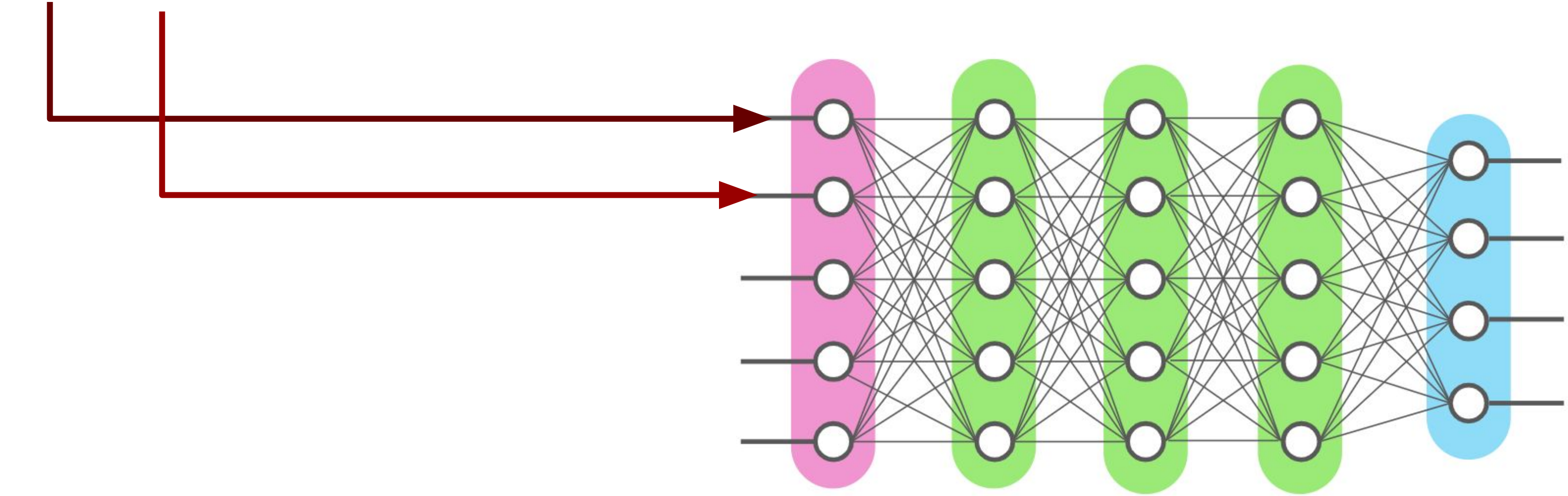
Language as input to ANNs

The weather is great .

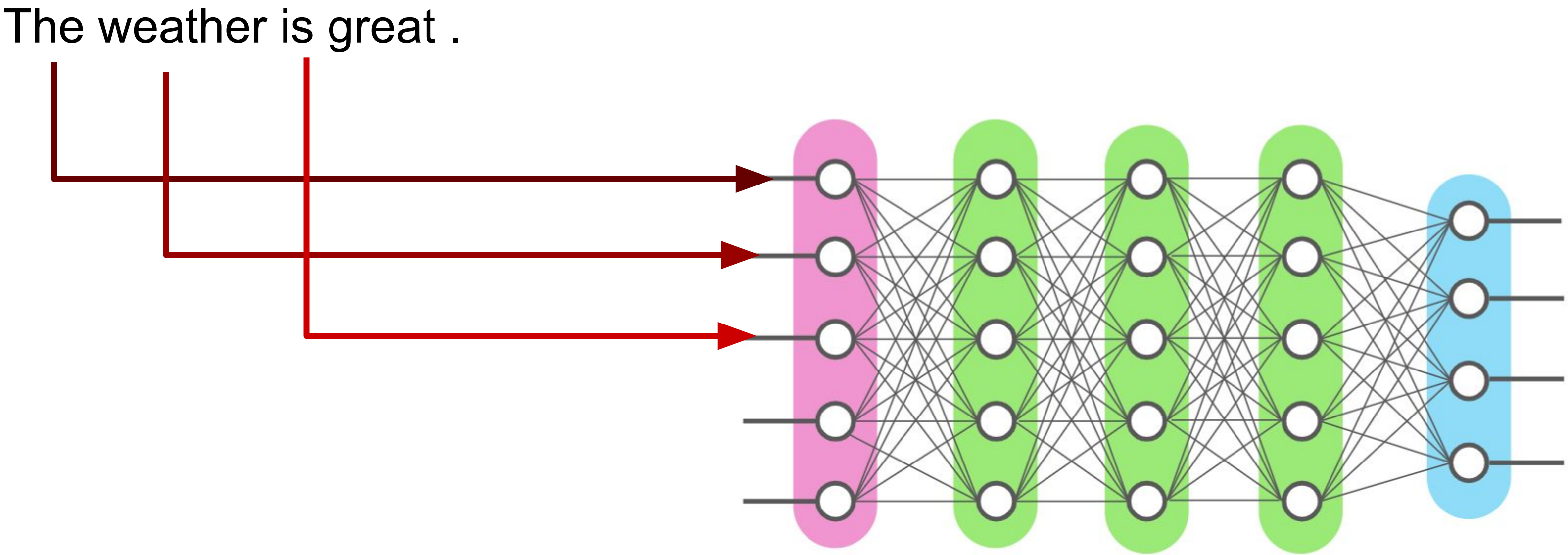


Language as input to a neural net

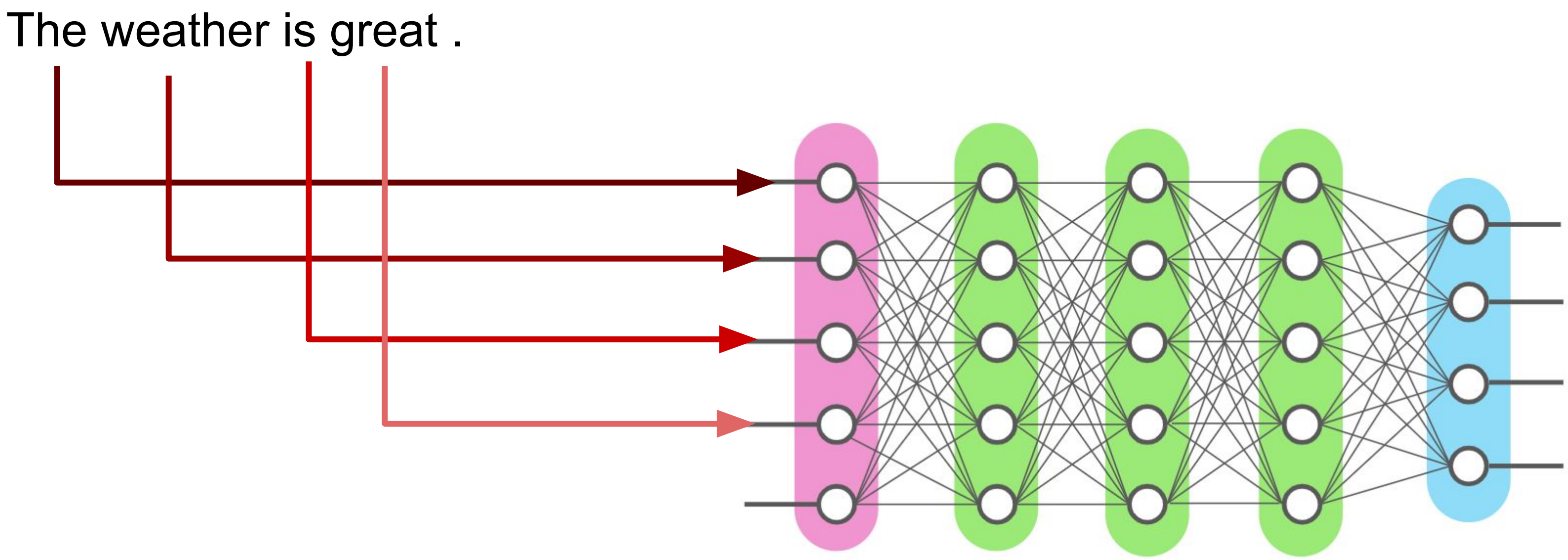
The weather is great .



Language as input to a neural net

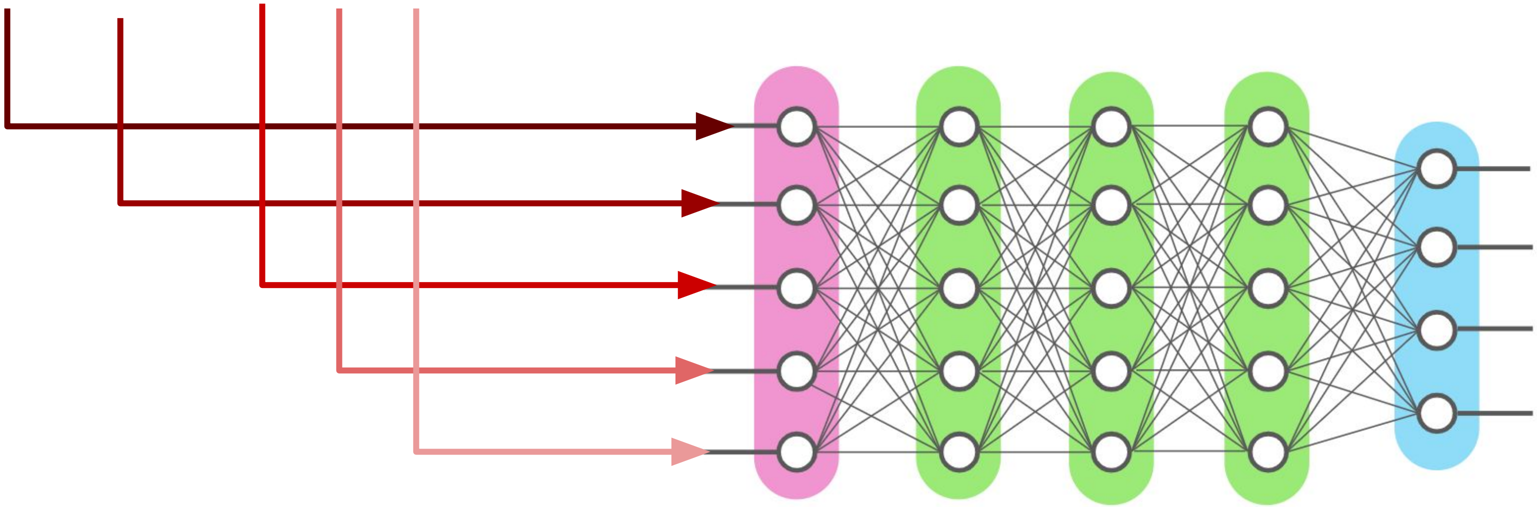


Language as input to a neural net

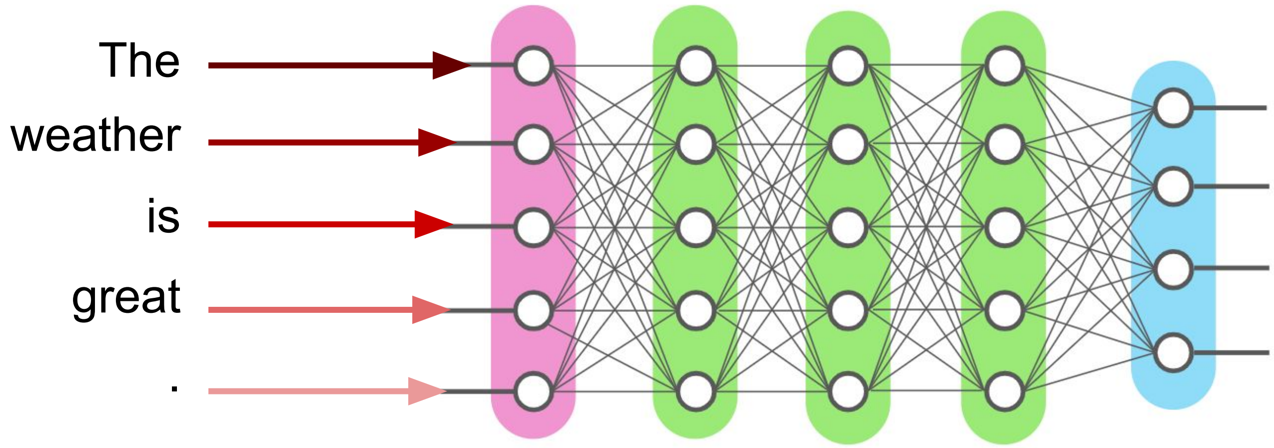


Language as input to a neural net

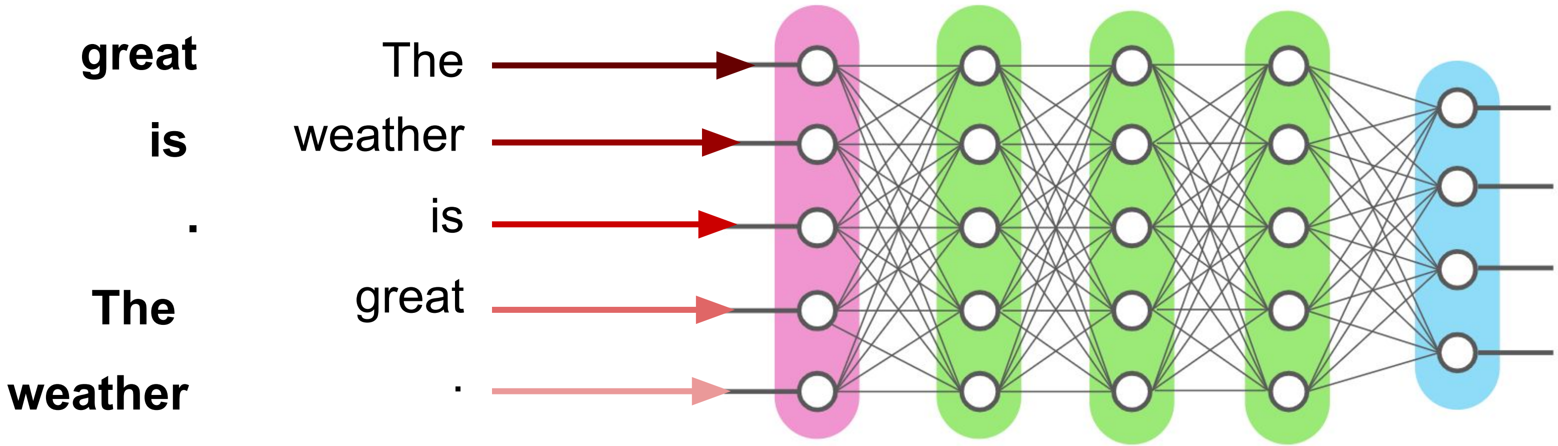
The weather is great .



Language as input to a neural net



Language as input to a neural net

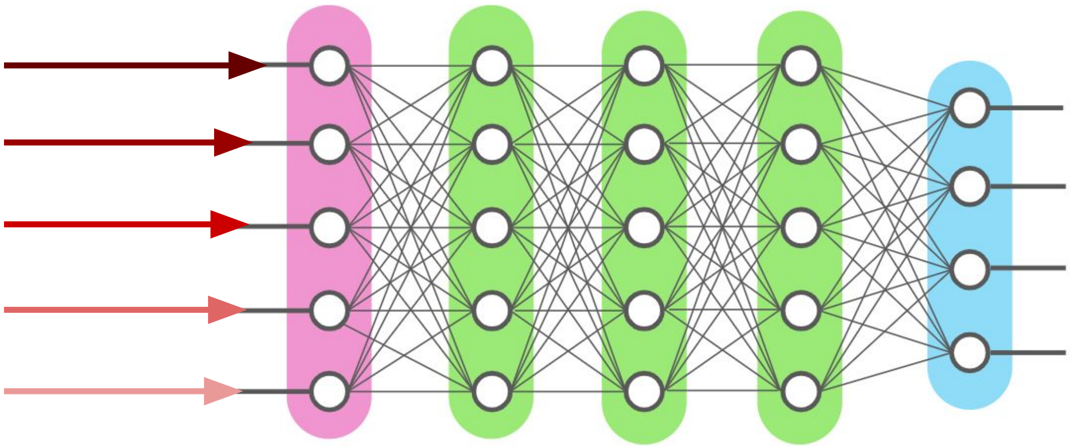


Language as input to a neural net

The weather is finally sunny!

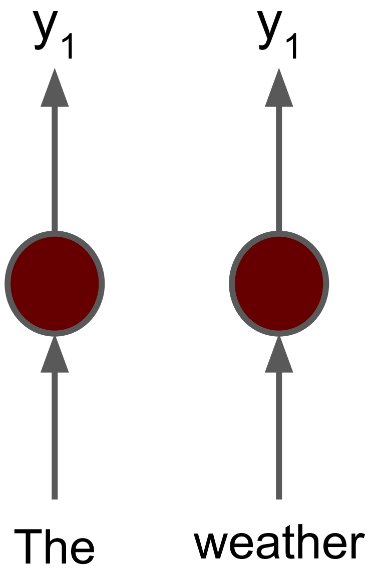
The
weather
is
finally
sunny
!

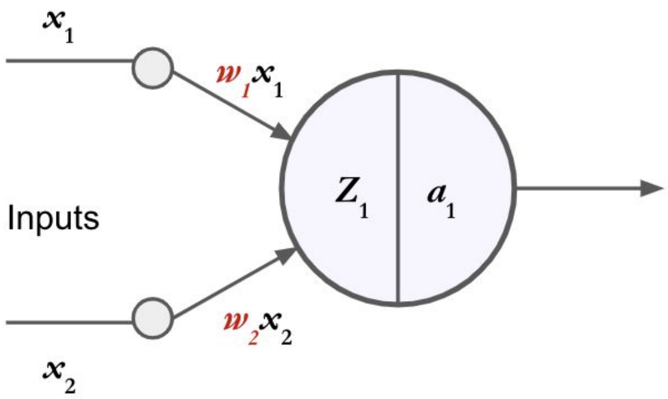
The
weather
is
great
.



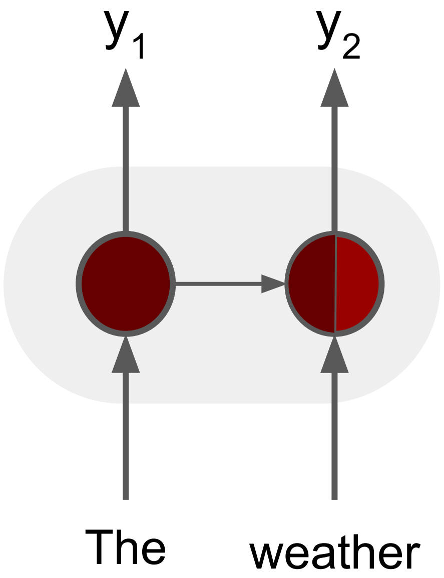
Recurrent neural nets (RNNs) were developed to address these limitations with ANNs

In ANNs each input is independent from the others

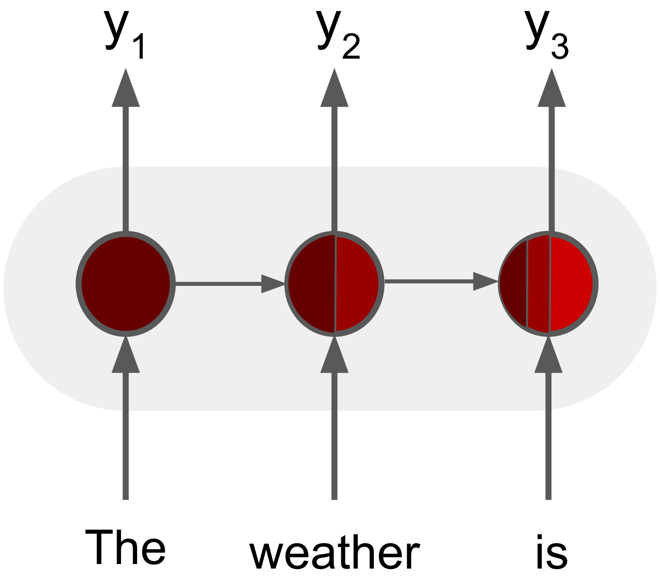




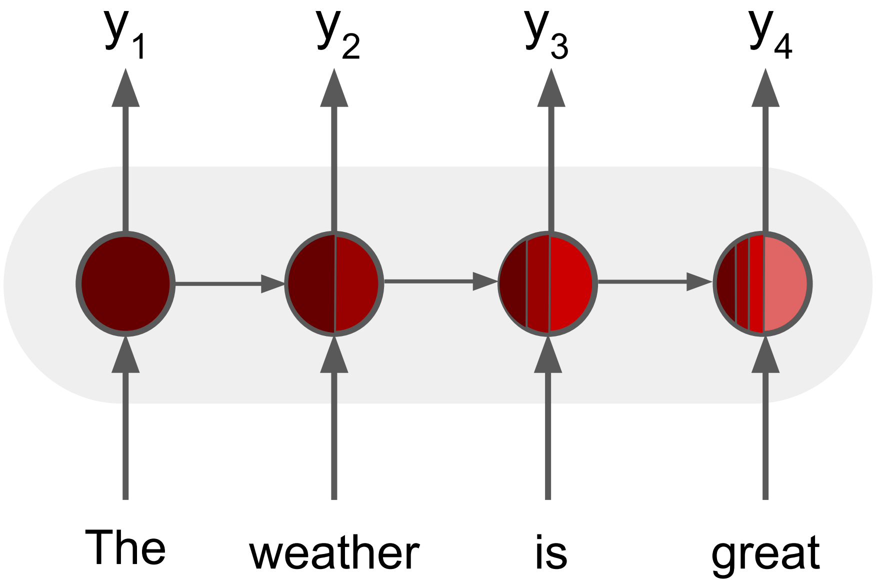
In RNNs, the output of the previous step is fed to the next step



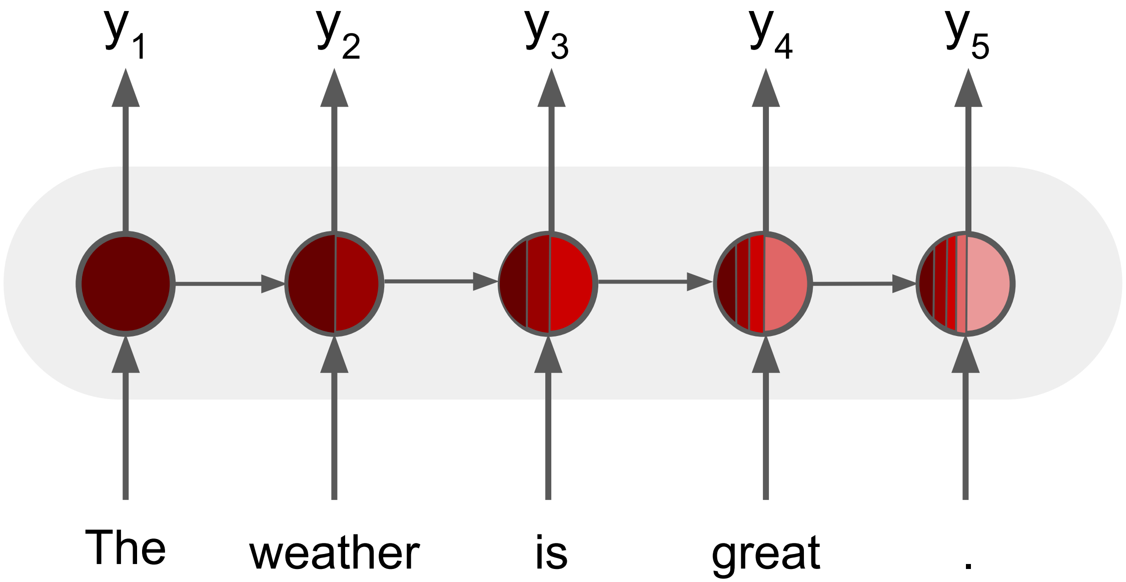
This allows the network to keep a memory of the previous steps



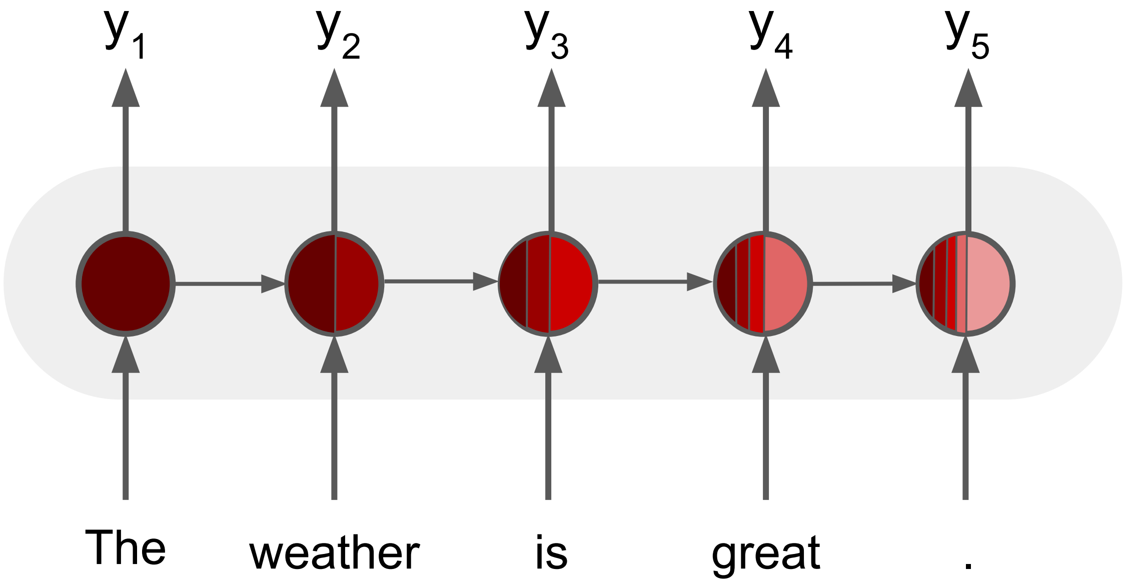
This allows the network to keep a memory of the previous steps



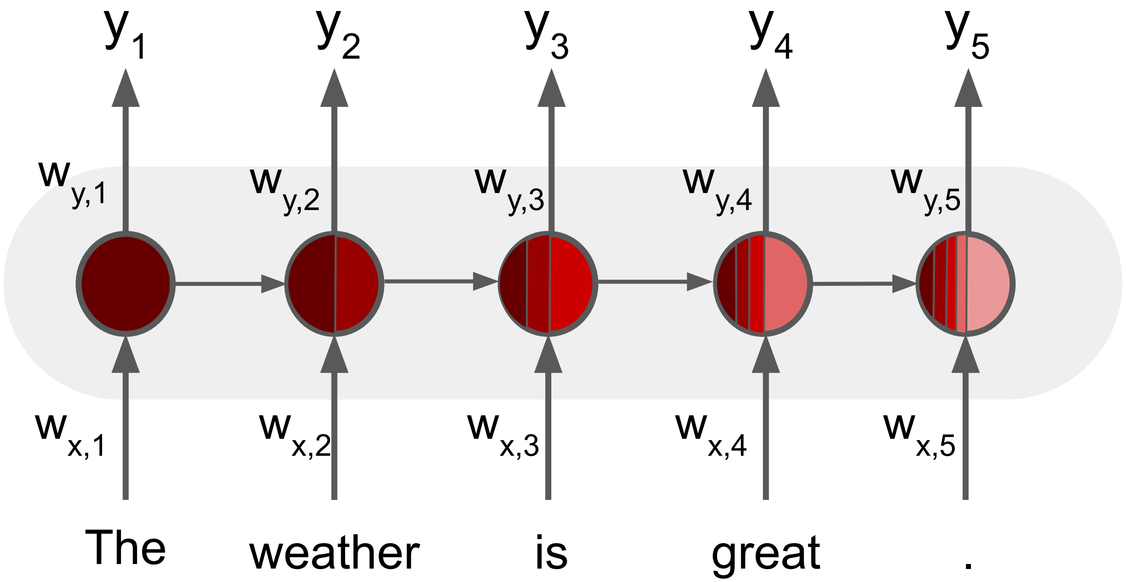
This allows the network to keep a memory of the previous steps



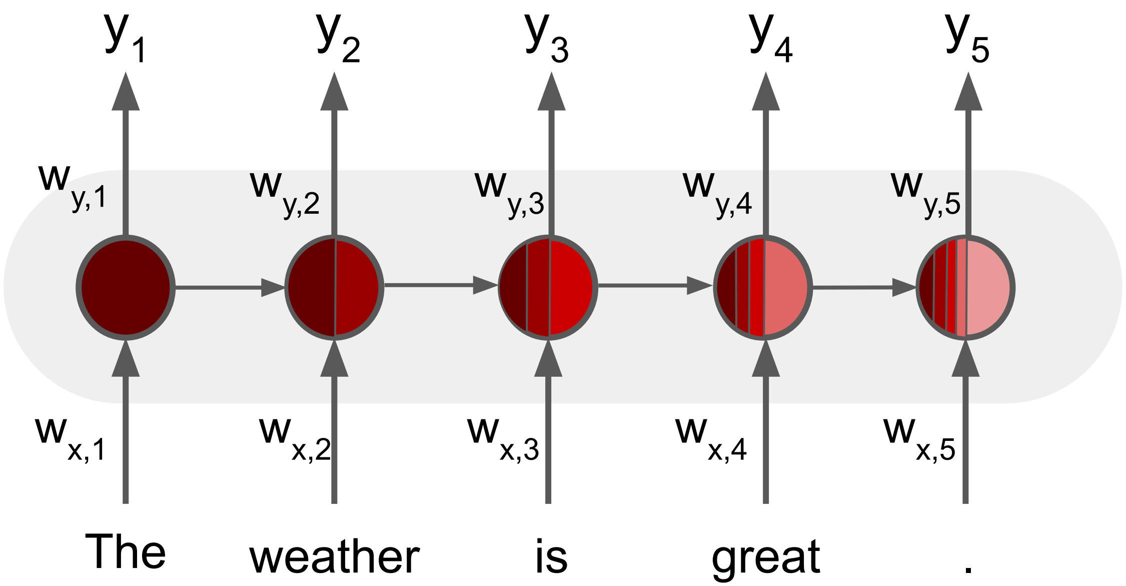
What if the sentences have different sizes?



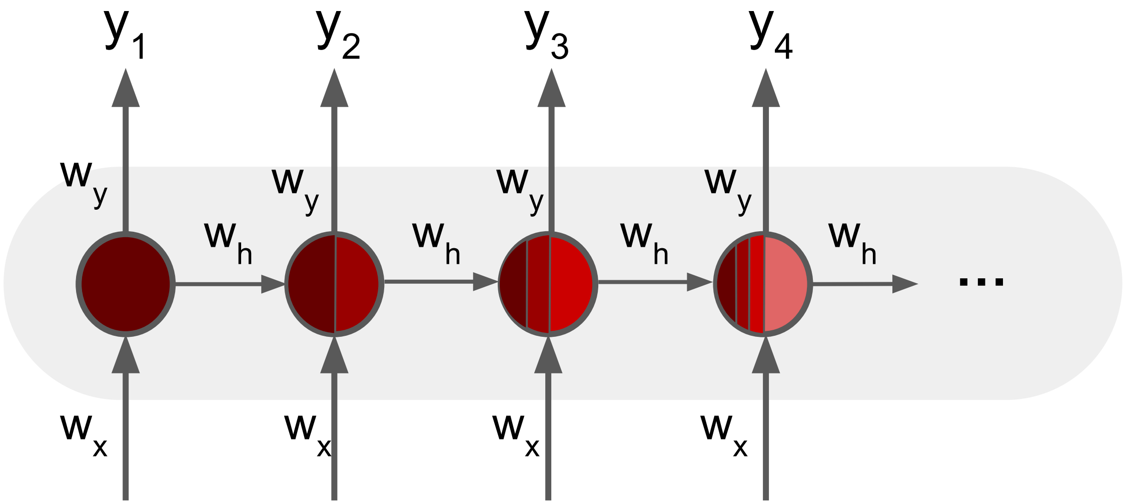
If each layer has different weights …



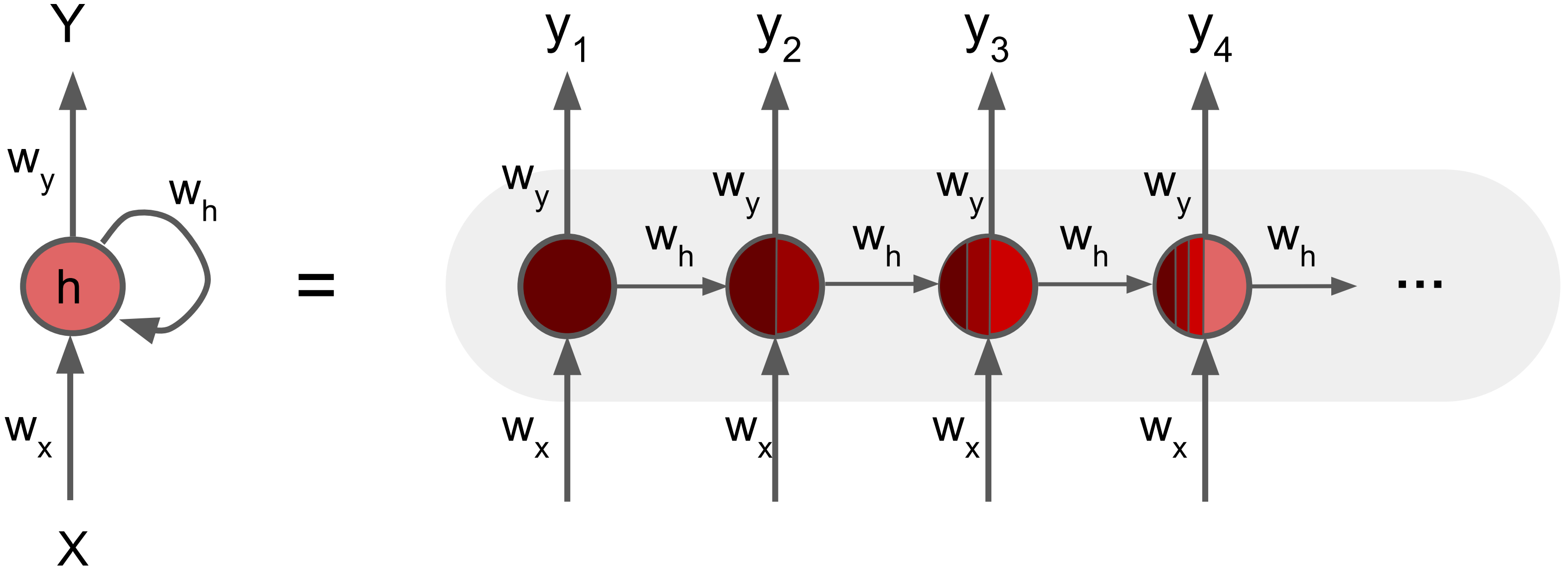
If each layer has different weights, we couldn't change the size



Parameter sharing in RNNs allow for adopting different lengths

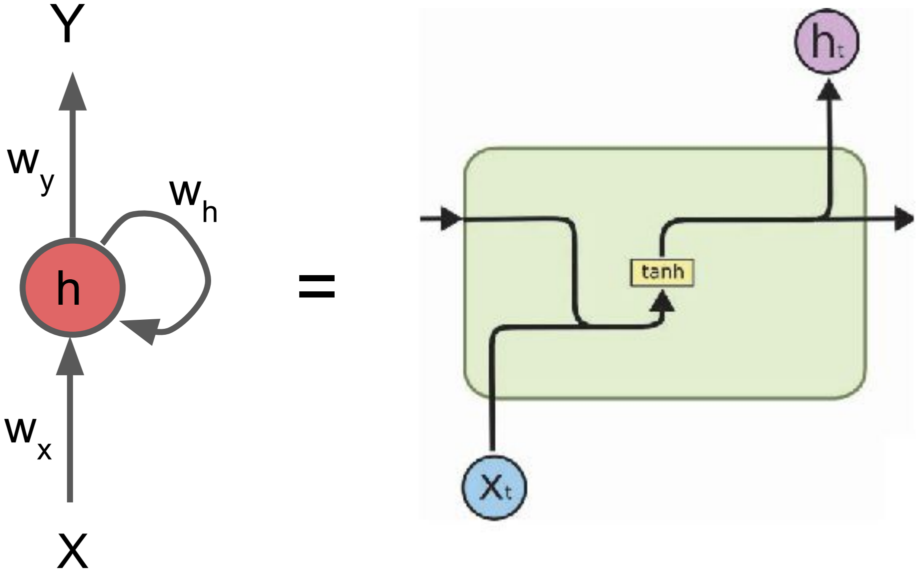


Parameter sharing in RNNs allow for adopting different lengths

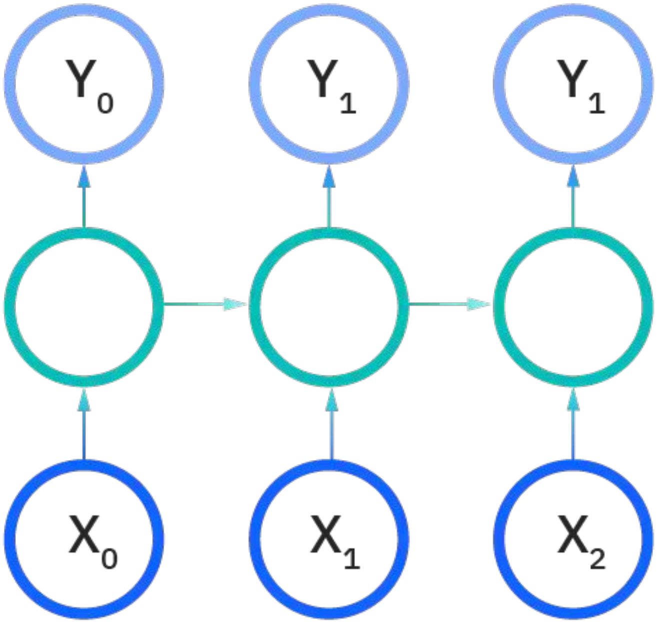


Rolled RNN

Parameter sharing in RNNs allow for adopting different lengths

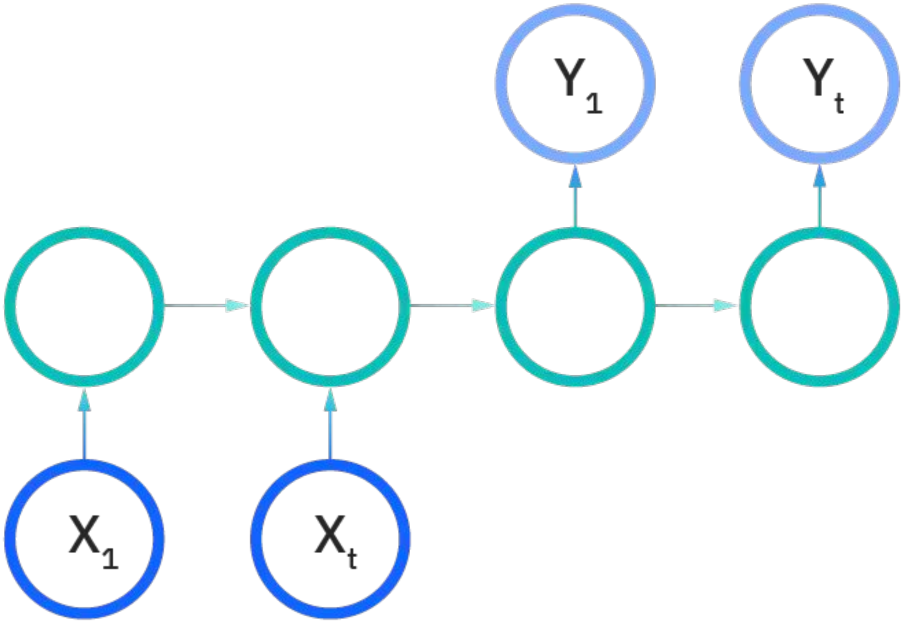


Types of RNN – many-to-many



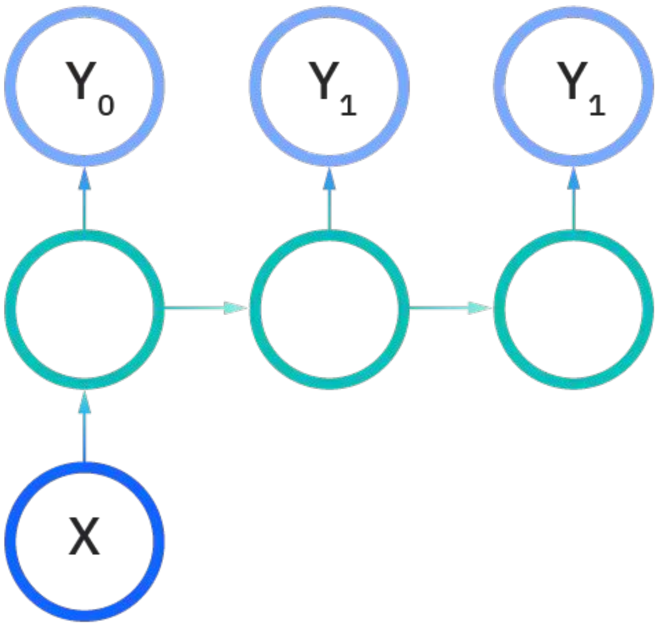
- Machine translation
- Time series prediction
- Sentence completion
- …

Types of RNN – many-to-many



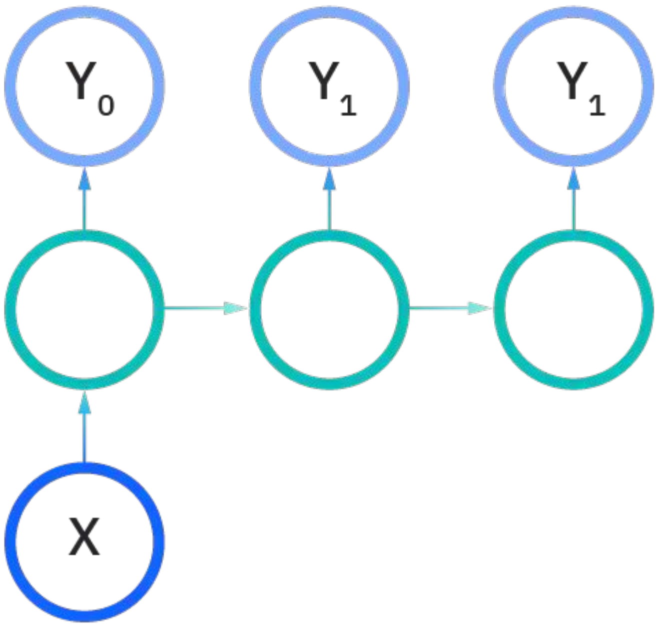
- Forecast prediction

Types of RNN – One-to-many



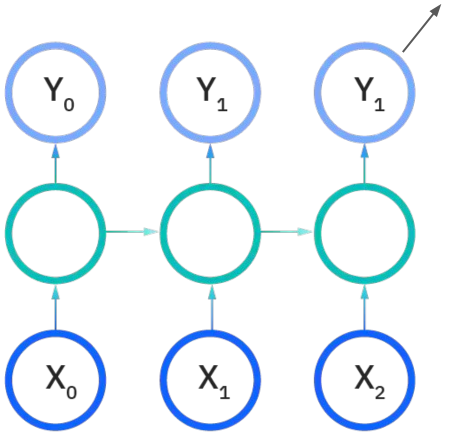
- Music generation
- Text generation

Types of RNN – One-to-many

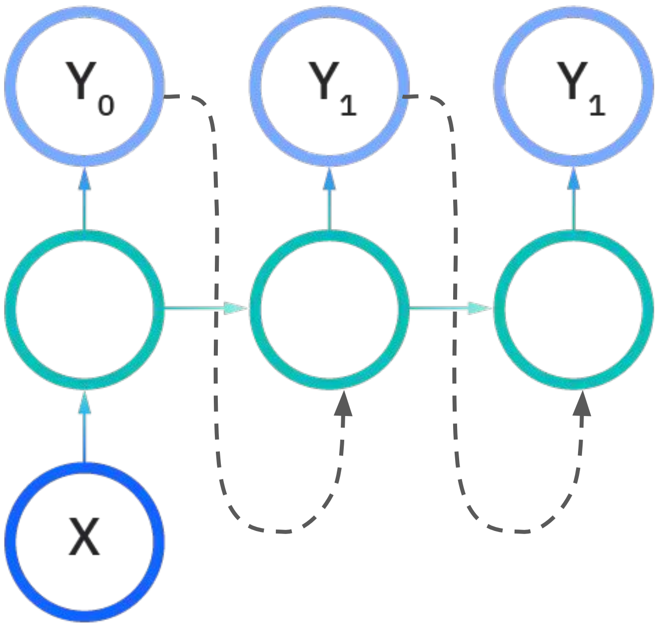


- Music generation
- Text generation

Step 1: Training

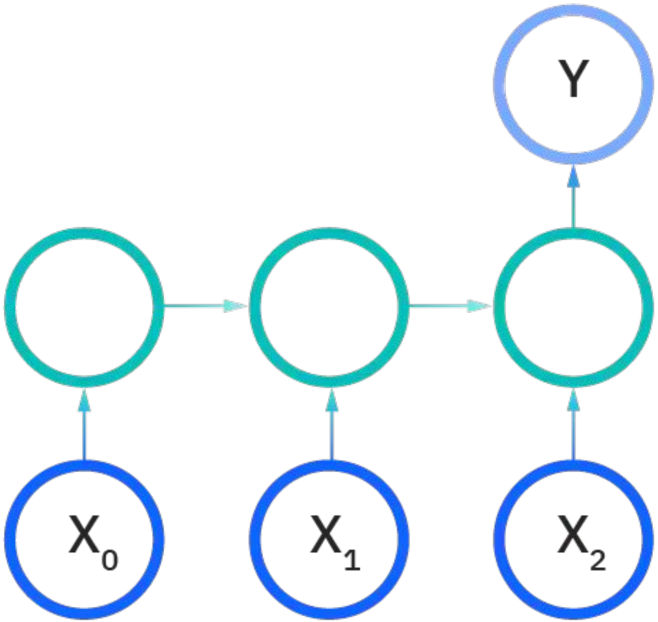


Types of RNN – One-to-many



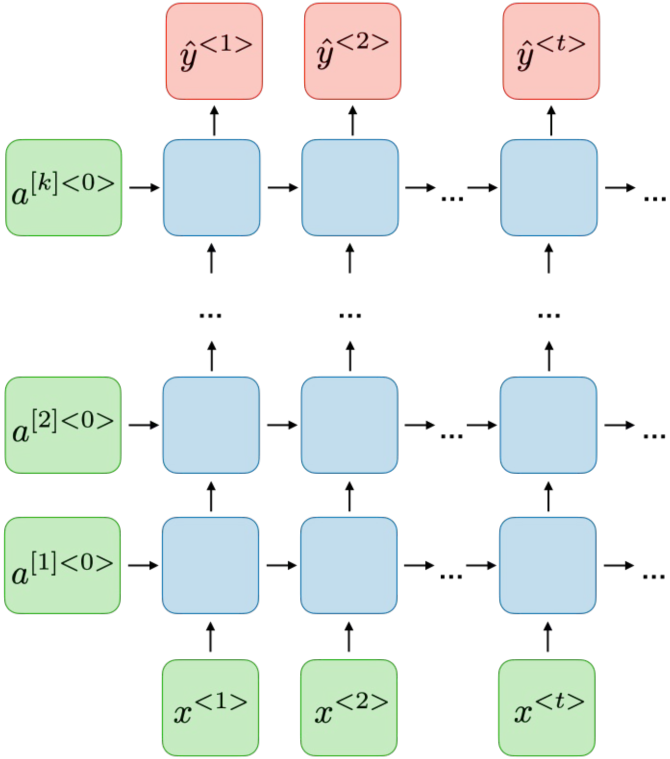
- Music generation
- Text generation

Types of RNN – many-to-one

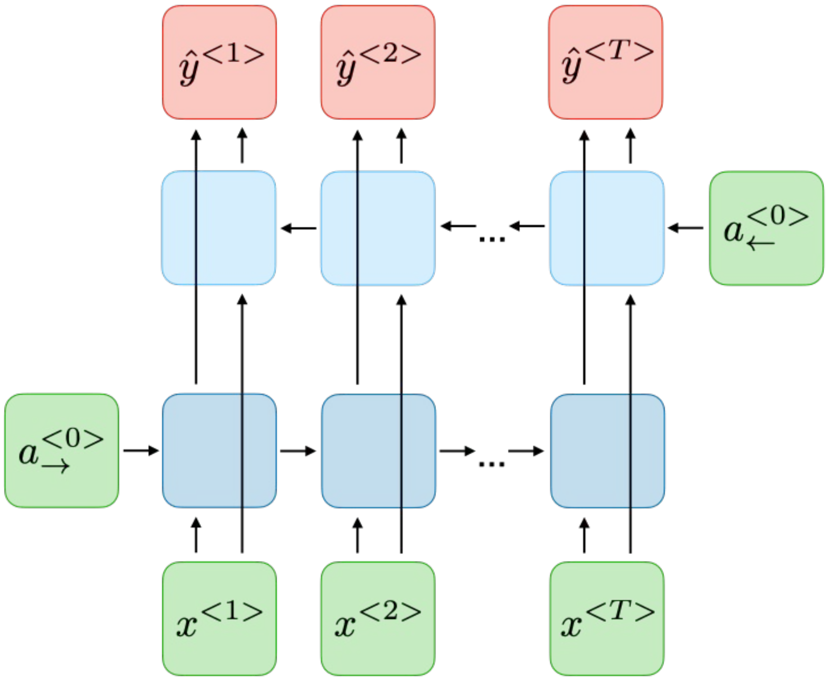


- Sentiment detection

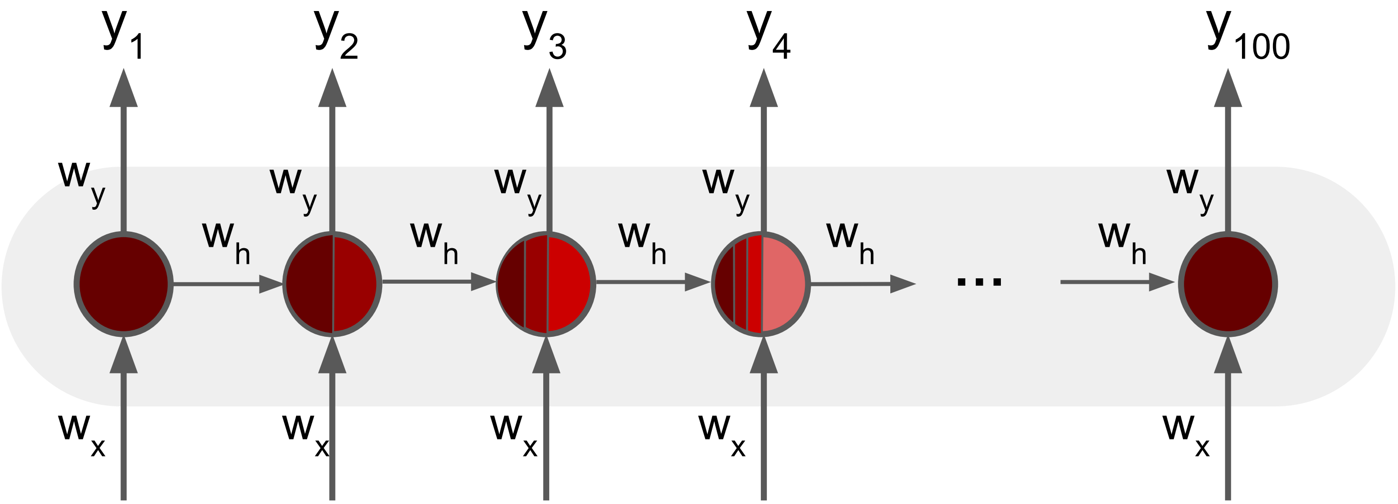
Architectures of RNN – Deep RNNs



Architectures of RNN – bidirectional

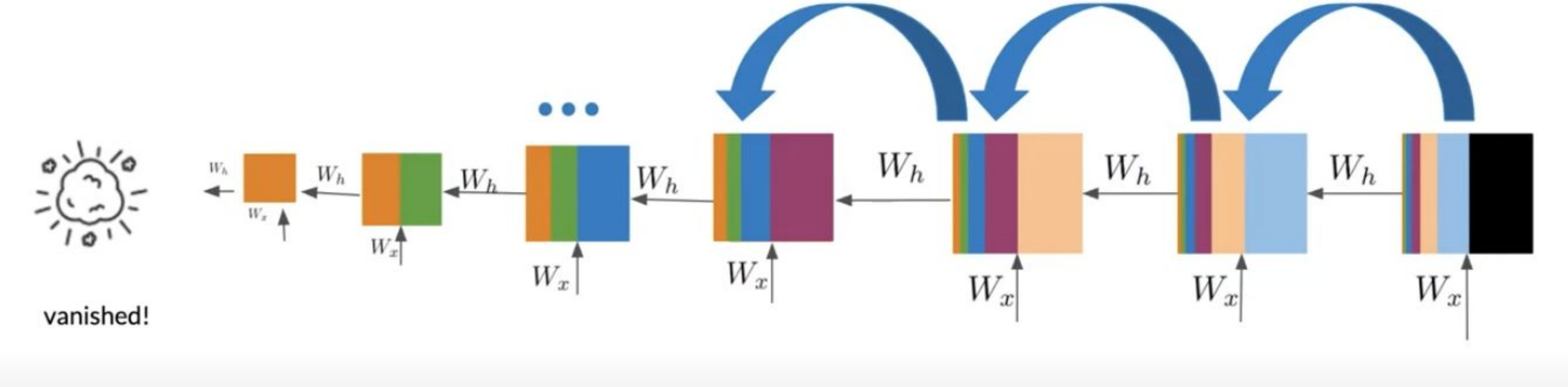


RNN and vanishing/exploding gradients

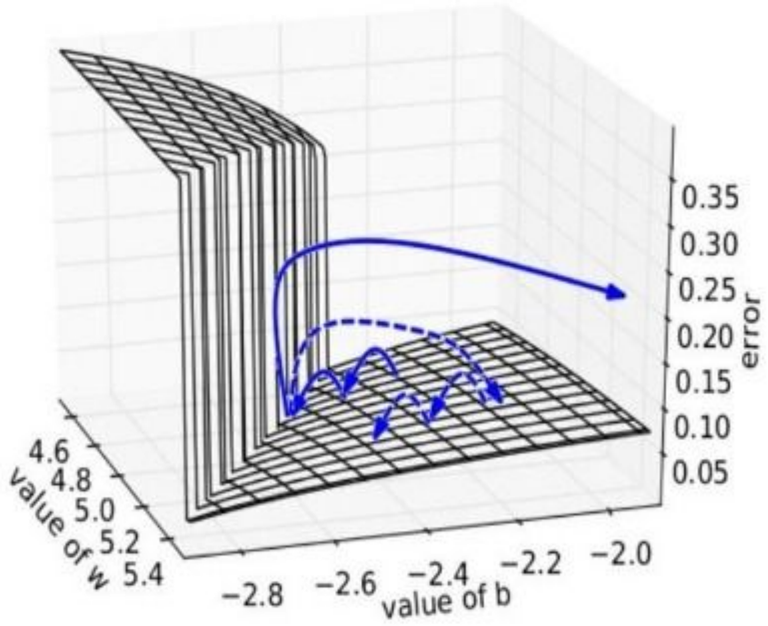


= 100 layer ANN

RNN and **vanishing**/exploding gradients



RNN and vanishing/**exploding** gradients



Solutions to vanishing/exploding gradients

1. Initializing with identity matrix and ReLU (identity RNN)

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Solutions to vanishing/exploding gradients

1. Initializing with identity matrix and ReLU (identity RNN)
2. Gradient clipping

```
if gradient > 25:
    gradient = 25
```

Solutions to vanishing/exploding gradients

1. Initializing with identity matrix and ReLU (identity RNN)
2. Gradient clipping
3. Skip connections

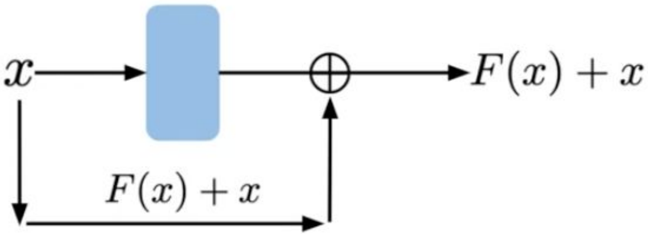


Image from machine learning TV: RNNs and vanishing gradients

To predict, RNNs need to remember the text

The sky is ____

To predict, RNNs need to remember the text

The sky is ___

To predict, RNNs need to remember the text

The sky is blue

To predict, RNNs need to remember the text

The sky is blue

I live in France. I love this city. I speak ___

To predict, RNNs need to remember the text

The sky is blue

I live in France. I love this city. I speak ____

To predict, RNNs need to remember the text

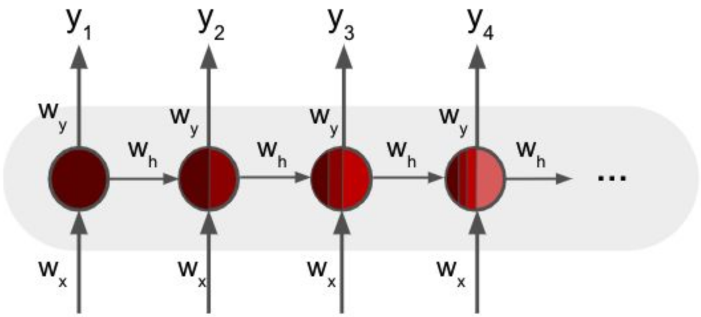
The sky is blue

I live in France. I love this city. I speak French

Due to limitations of the architecture, remembering context from past is challenging

The sky is blue

I live in France. I love this city. I speak French



Next lecture:
LSTMs and GRUs