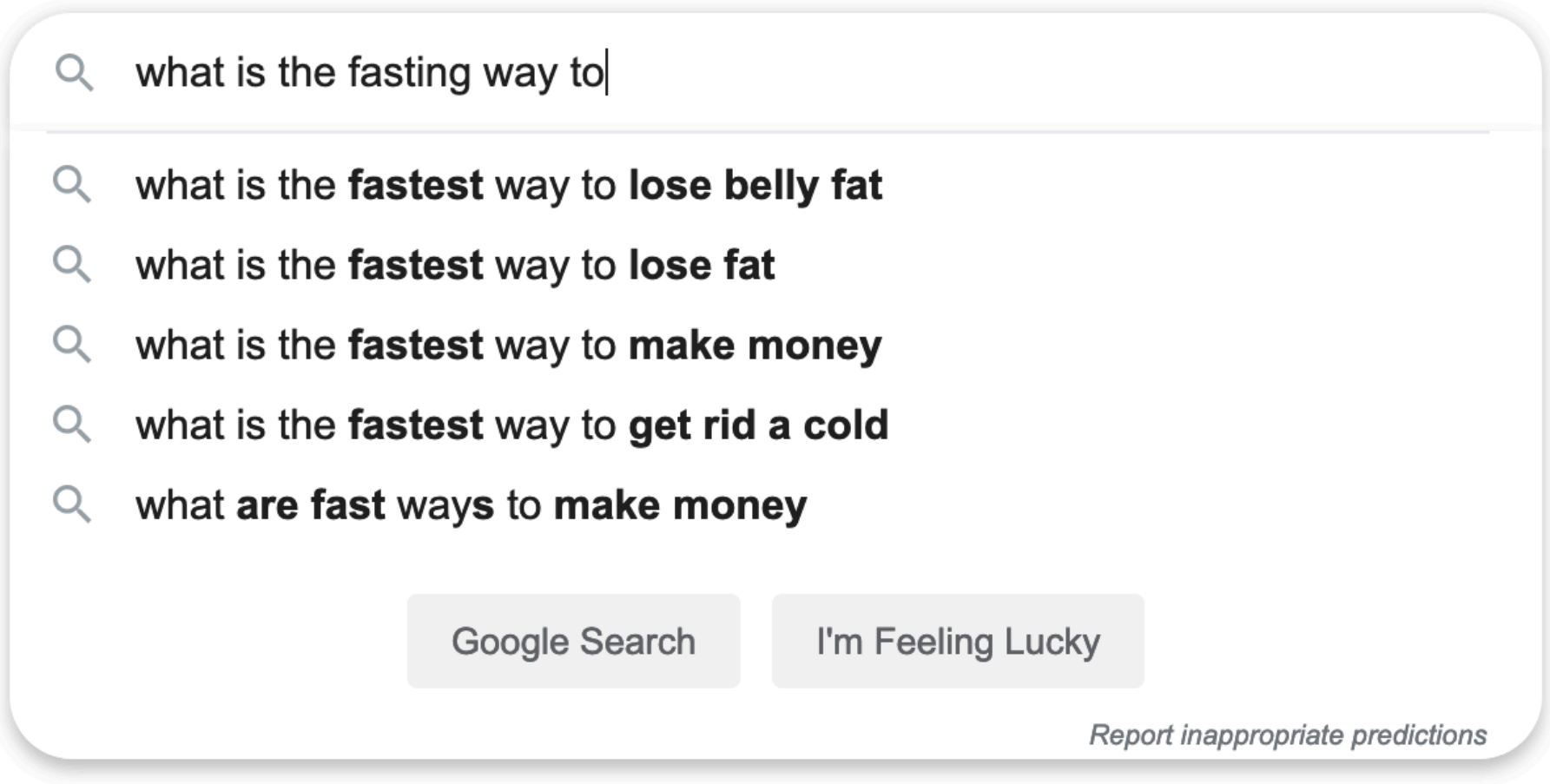
Text Generation

SCHOOL OF INFOCOMM

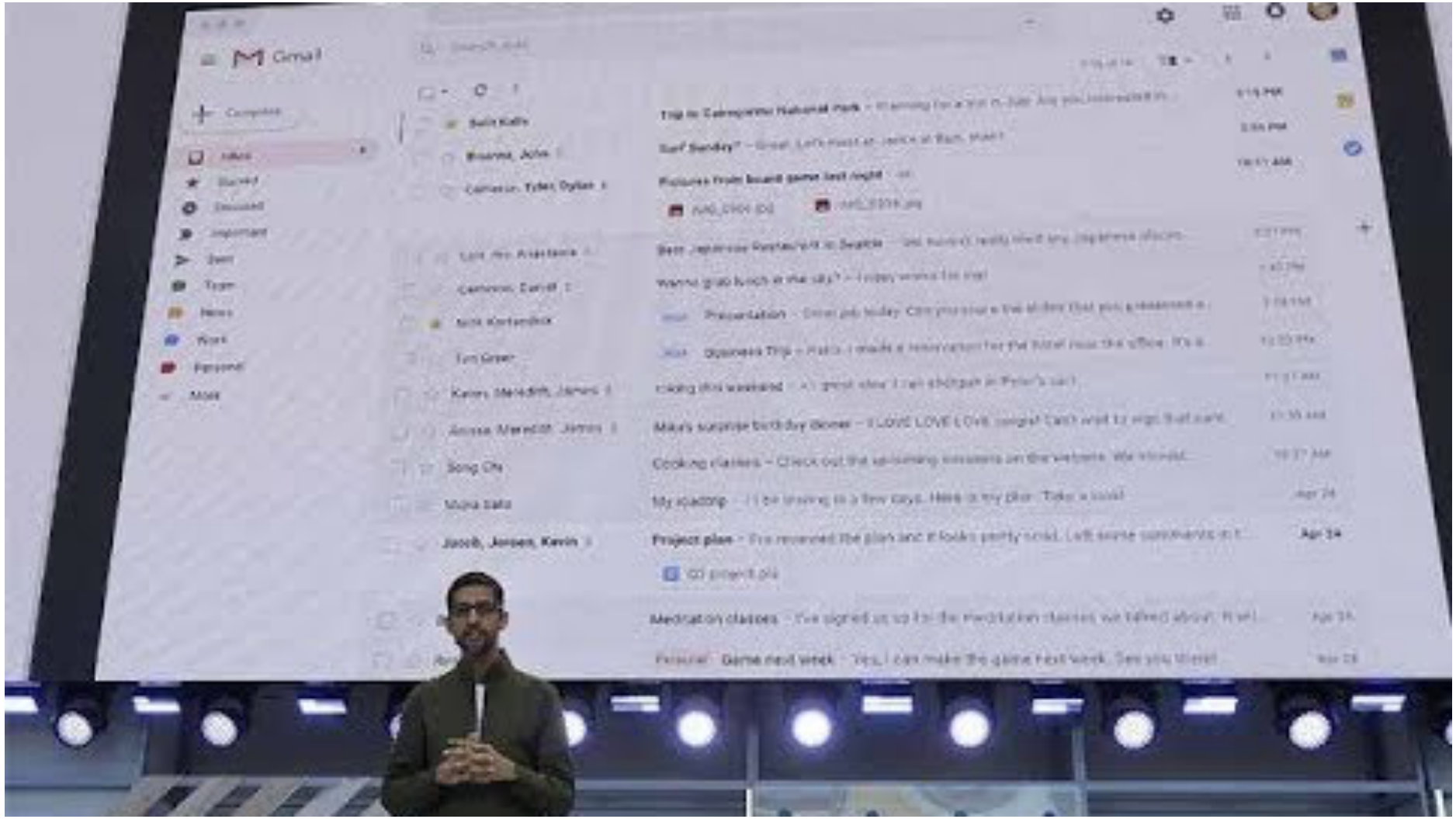
Text Generation

- If you can predict the next character in a sequence of characters, you can spell word.
- If you can predict the next word in a sentence, you can predict the word after that, and the next after that and so on.
- If you can predict a sequence of meaningful words, you can possibly get meaningful sentences.
- If you can get a collection of meaningful sentences, you can possibly form coherent paragraphs.
- If you can form coherent paragraphs, you can generate AI powered fakenews / songs / storiesa !

Auto-complete a search



Write Email Faster



https://youtu.be/nZ-C8I-8BZw

Language transaction

Goal: Translate a "source" text from one language to target "text" in another language

Approaches for Text Generation

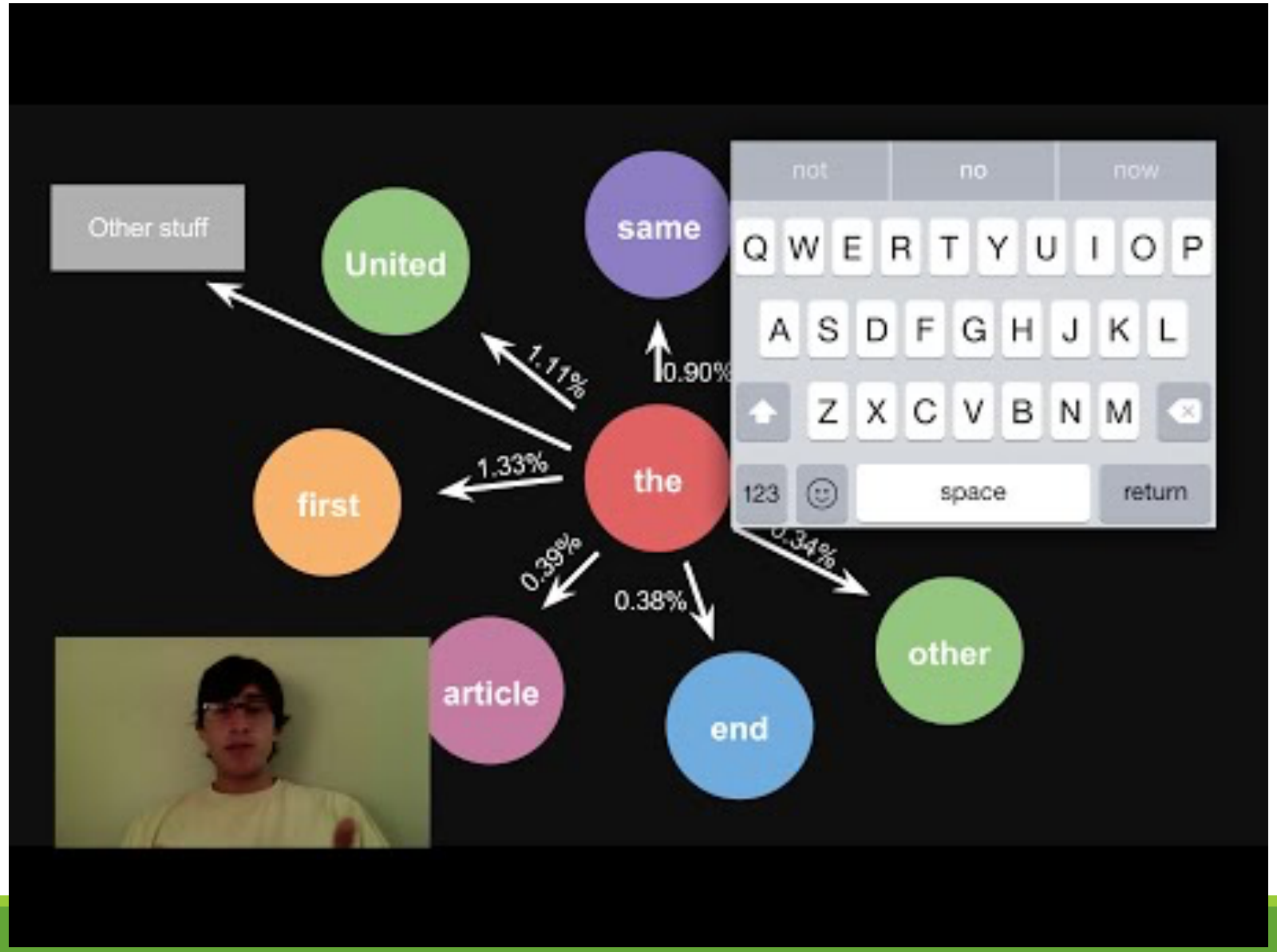
Language modeling is the task of predicting the next word(s) in a sentence, based on the previous words.

Stochastic based approach: These models use statistical techniques and certain linguistic rules to learn the **probability distribution** of words. *Hidden Markov Models (HMM)*

Neural netwok approach: Direct application of neural network to surpass limitation of statistical language models

- *Recurrent Neural Network (RNN)*
- *Attention Mechanism*

Markov Chains



https://youtu.be/56mGTszb_iM

How can we build a probabilistic understanding of this text?

"Both the brown fox and the brown dog slept."

Source: Intel AI Developer Program

“Both the brown fox and the brown dog slept.”

Let’s build a dictionary that tracks the “current” two words, and what word comes after them. We can treat the two words as the “current state” and the next word as a “next available” state.

Source: Intel AI Developer Program

“Both the brown fox and the brown dog slept.”

{ (Both, the): [brown] }

Source: Intel AI Developer Program

"Both the brown fox and the brown dog slept."

{ (Both, the): [brown],
  (the, brown): [fox], }

Source: Intel AI Developer Program

“Both the brown fox and the brown dog slept.”

{ (Both, the): [brown],
(the, brown): [fox],
(brown,fox): [and], }

"Both the brown fox and the brown dog slept."

{ (Both, the): [brown],
(the, brown): [fox],
(brown,fox): [and],
(fox, and): [the], }

Source: Intel AI Developer Program

"Both the brown fox and the brown dog slept."

{ (Both, the): [brown],
(the, brown): [fox],
(brown,fox): [and],
(fox, and): [the],
(and, the): [brown],}

Source: Intel AI Developer Program

"Both the brown fox and the brown dog slept."

{ (Both, the): [brown],
(the, brown): [fox, dog],
(brown,fox): [and],
(fox, and): [the],
(and, the): [brown],}

Now if we are in a current state of "the brown", both "fox" and "dog" are equally likely to occur!

So we have a state dependent chain of probabilities.

We can use Markov Chains here!

Source: Intel AI Developer Program

Text Generation with Markov Chains

{ (Both, the): [brown],
(the, brown): [fox, dog],
(brown,fox): [and],
(fox, and): [the],
(and, the): [brown],
(brown, dog): [slept] }

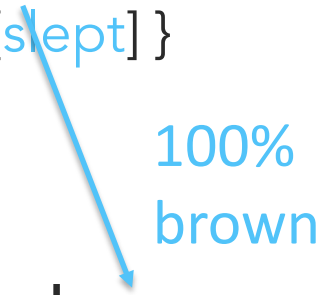
Let's start with a sentence seed of "and the" and this dictionary of word relationships and roll some dice.

"and the "

Source: Intel AI Developer Program

Text Generation with Markov Chains

{ (Both, the): [brown],
(the, brown): [fox, dog],
(brown,fox): [and],
(fox, and): [the],
(and, the): [brown],
(brown, dog): [slept] }



"and the brown "

Let's start with a sentence seed of "and the" and this dictionary of word relationships and roll some dice.

Source: Intel AI Developer Program

Text Generation with Markov Chains

{ (Both, the): [brown],
(the, brown): [fox, dog],
(brown,fox): [and],
(fox, and): [the],
(and, the): [brown],
(brown, dog): [slept] }

Let's start with a sentence seed of "and the" and this dictionary of word relationships and roll some dice.

50% dog, 50% fox.
ROLL DICE.

"and the brown fox "

Source: Intel AI Developer Program

Text Generation with Markov Chains

{ (Both, the): [brown],
(the, brown): [fox, dog],
(brown,fox): [and],
(fox, and): [the],
(and, the): [brown],
(brown, dog): [slept] }

Let's start with a sentence seed of "and the" and this dictionary of word relationships and roll some dice.

Because of our limited selection of words, we are now stuck completing the sentence.

"and the brown fox and the brown dog slept"

Source: Intel AI Developer Program

Example output with a bigger corpus

Input: A few H.P. Lovecraft Books
Markov System: Two word current state.

```
the moon, and the newcomers and there are things about the lock
on the deck of a monstrous odour...senses transfigured...
boarding at that moment it struck me, known all along the banks
were bent and dirty, he was shown the thing vanished with him.
the dusk of the blasphemously stupendous bulk of our ascent of
the real article, when marsh suddenly swerved from abstractions
to the town and on april 2nd was driven by some remoter advancing
bulk--and then came the final throaty rattle came. dr houghton of
aylesbury, who had reigned as a beginning of a laugh. the horror
```

Source: Intel AI Developer Program

Exercise 1

Text generation using Markov Chain

Refer to Jupyter notebook:
ex1-text_generation_using_markovchains.ipynb

Difference Sentences , Same Bag of Words

HARRY IS CRAZY ABOUT SALLY BUT NOT ADELINE

ADELINE IS CRAZY ABOUT HARRY BUT NOT SALLY

SALLY IS CRAZY ABOUT HARRY BUT NOT ADELINE

Each of these statements will become the same bag of words with the same count and TFIDF

[ABOUT, ADELINE, BUT, CRAZY, HARRY, IS, NOT, SALLY)

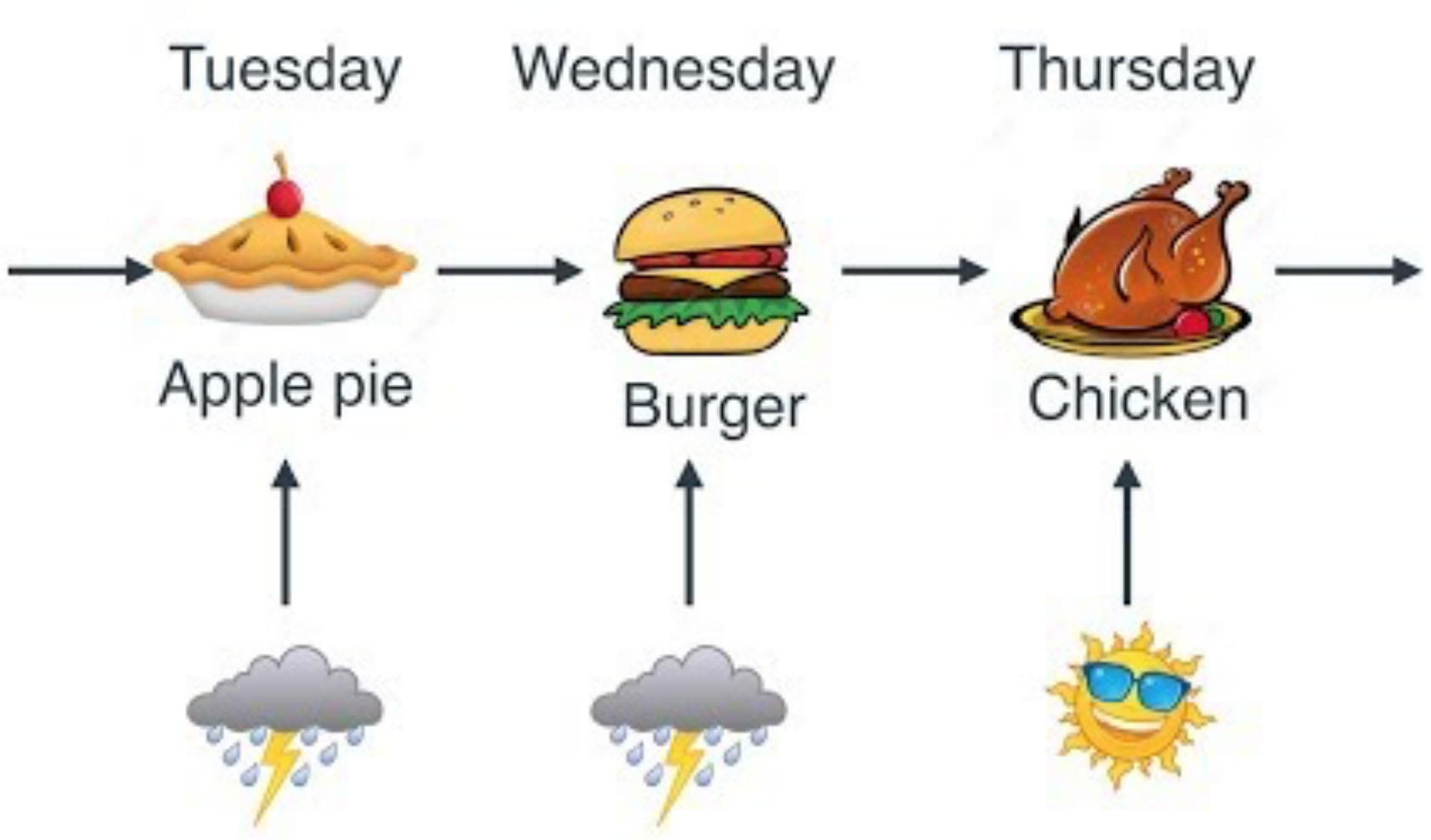
How Do Human Read (I)?

THEY WANTED TO PET THE DOG WHOSE FUR IS BROWN

- Read forward until FUR → knows that the fur belongs the dog

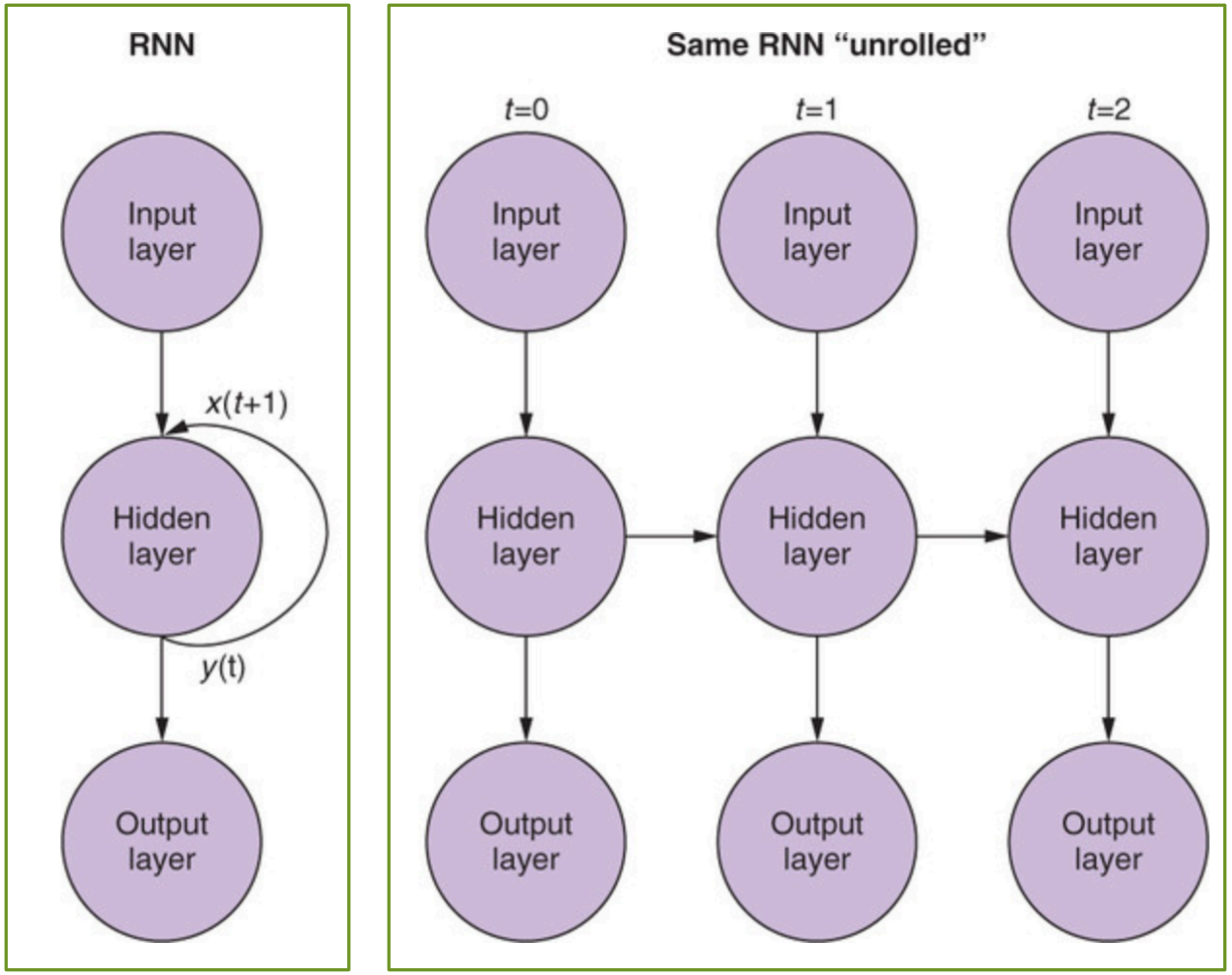
Considering that natural language is a sequence of words, symbols and characters, what has been revealed in the natural course of reading is important for understanding the sequence

RNN – A short introduction



https://youtu.be/UNmqTiOnRfg

Recurrent Neural Network (RNN)



In RNN, the input is sent into the network over a sequence of steps over time

Each step is known as a "time step"

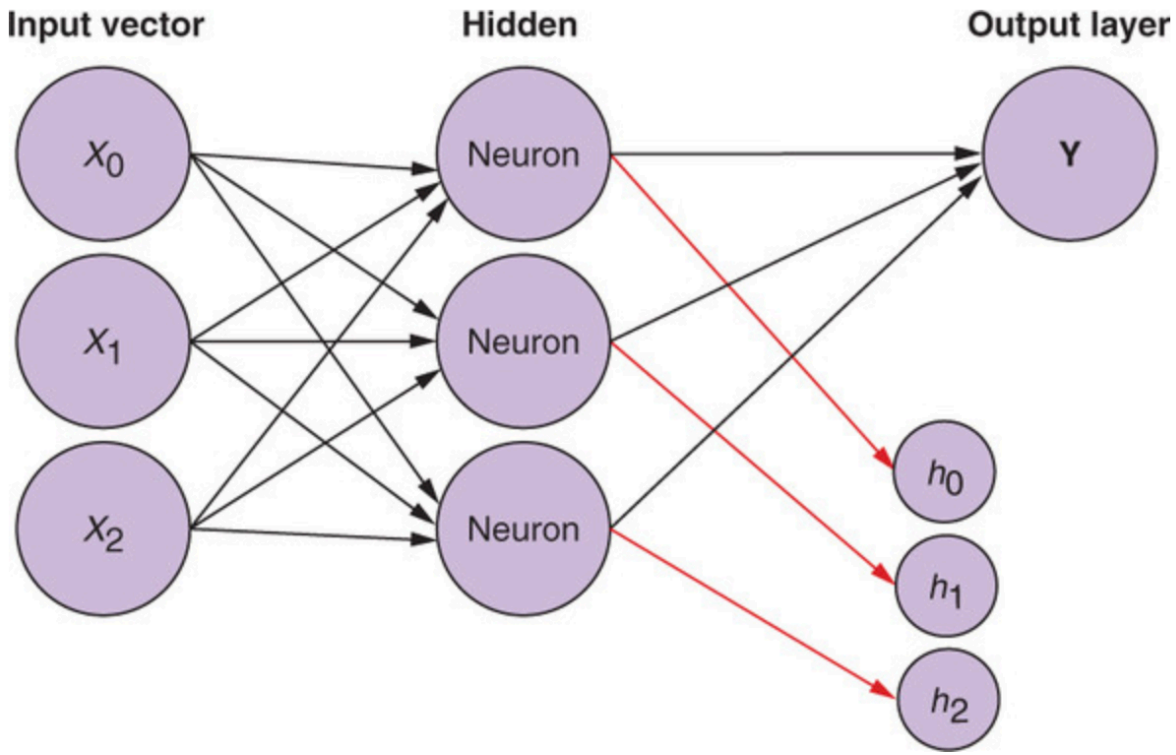
Output from time step (t) is feedback into the **same hidden layer** at the next time step (t+1).

At the time step (t+1), the network has some information on what was processed in the prior time step

Each vertical is the same network with the same weights

Figure from: Natural Language Processing in Action, Manning Publications

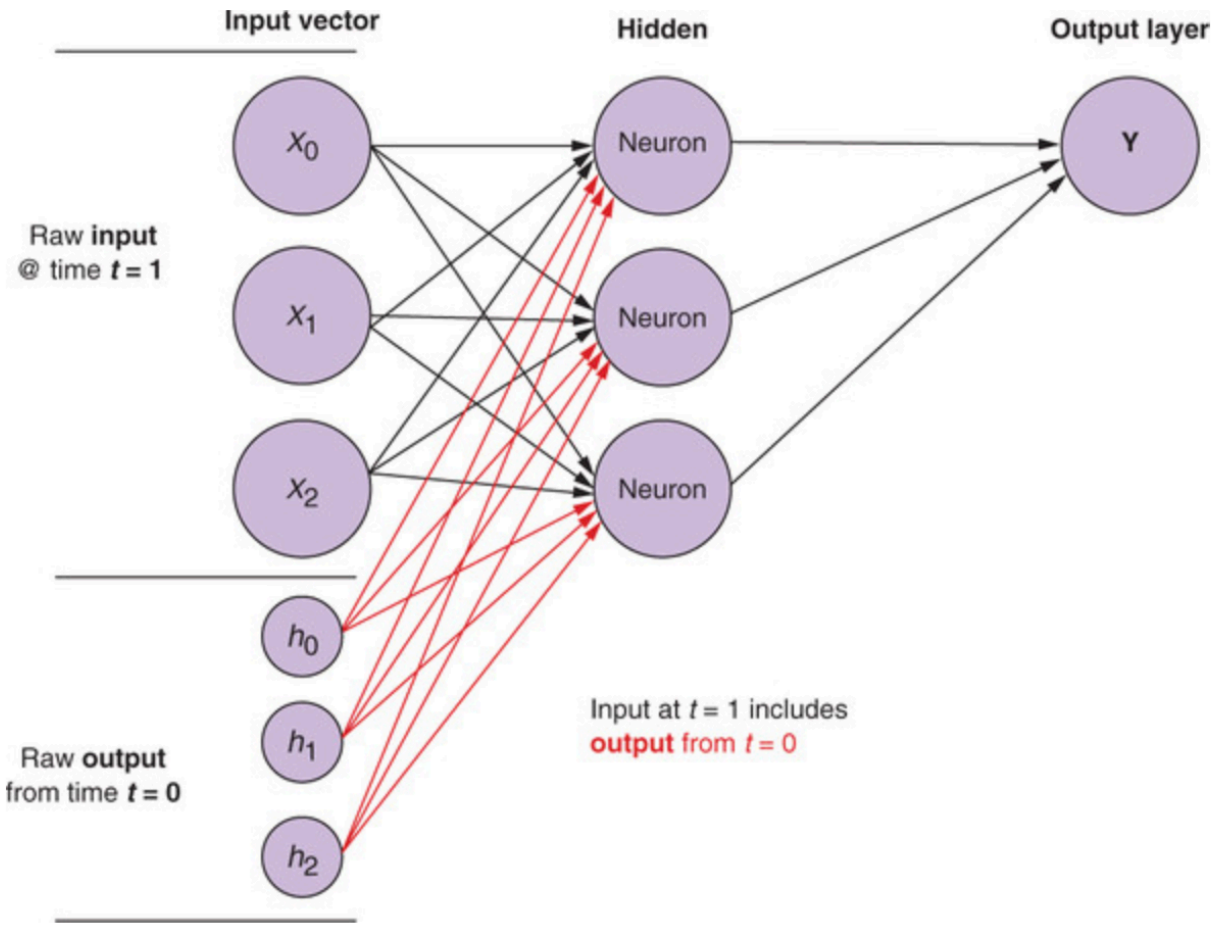
Time step t = 0



(h0, h1, h2) contains information about (X0, X1, X2) and (Y)

Figure from: Natural Language Processing in Action, Manning Publications

Time step t = 1



(h0, h1, h2) from the prior time step is feed into the neurons, this time together with new (X0, X1, X2) and generate another output (Y)

Figure from: Natural Language Processing in Action, Manning Publications

Time Step and Text

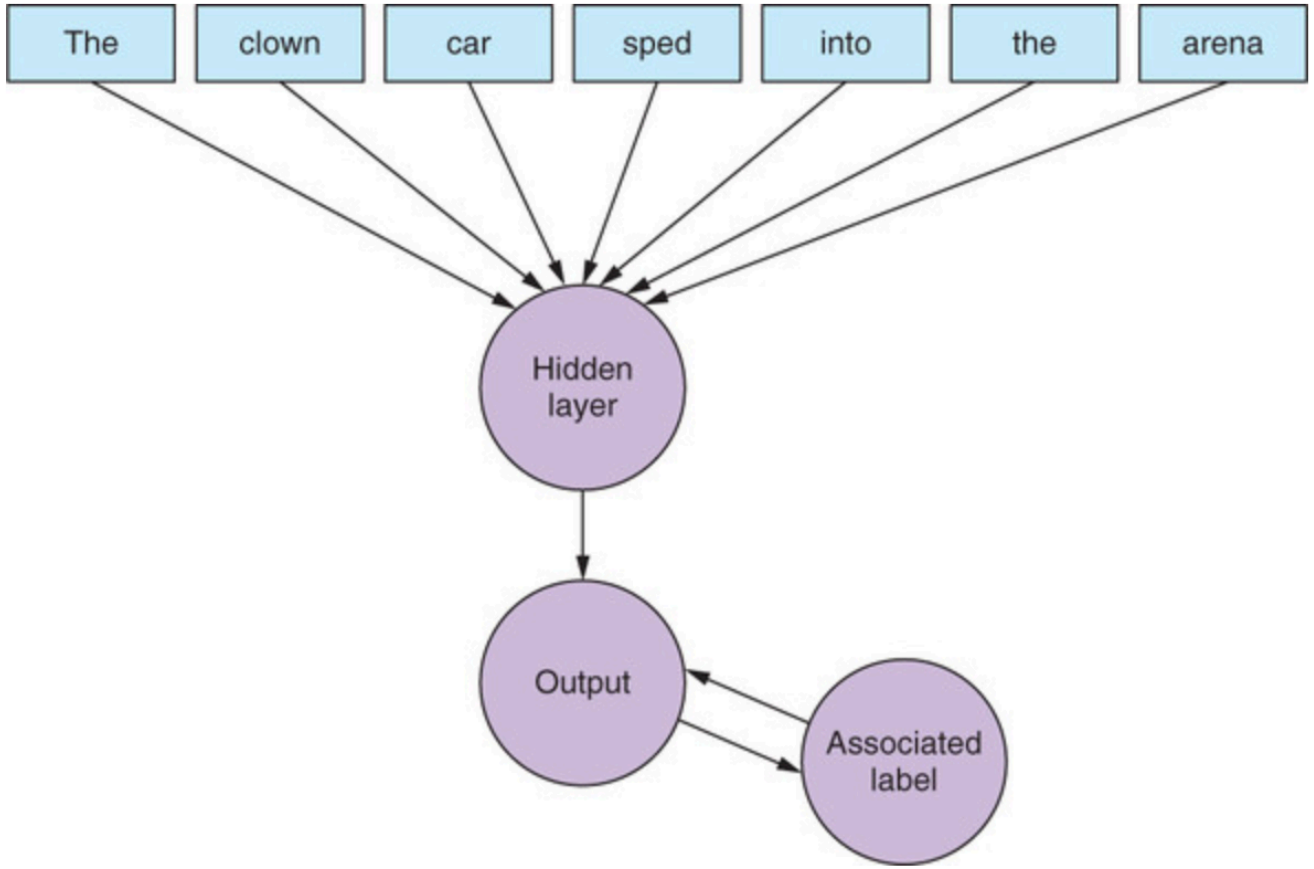
Suppose a sentence is tokenized into words

Instead of feeding the entire text into the network, we pass into the network one token at a time

Each token input is equivalent to one time step

Each time step is the same as the token sequence index

CNN



Consider a document classification problem.

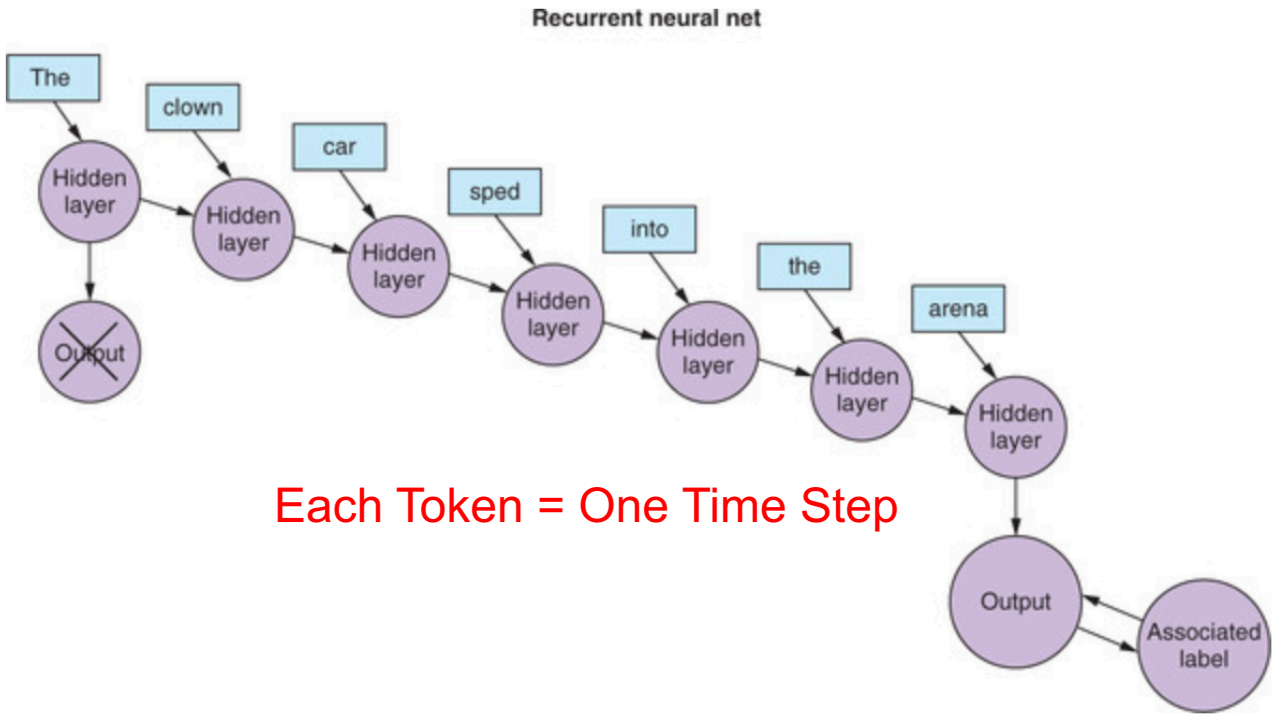
The sentence is tokenized into words

We feed all the tokens into the hidden layout at one time and output a label

This is a bag of words approach. No information of the order of words.

Figure from: Natural Language Processing in Action, Manning Publications

RNN



Consider a document classification problem

Suppose that sentence is tokenized into words

We feed into the network one token at a time

At each time step, a new token is processed together with information from the prior token

Until the last token, we are still working with the same sentence

The network can now output a different label by considering the how the tokens are sequenced

Figure from: Natural Language Processing in Action, Manning Publications

Backpropagation

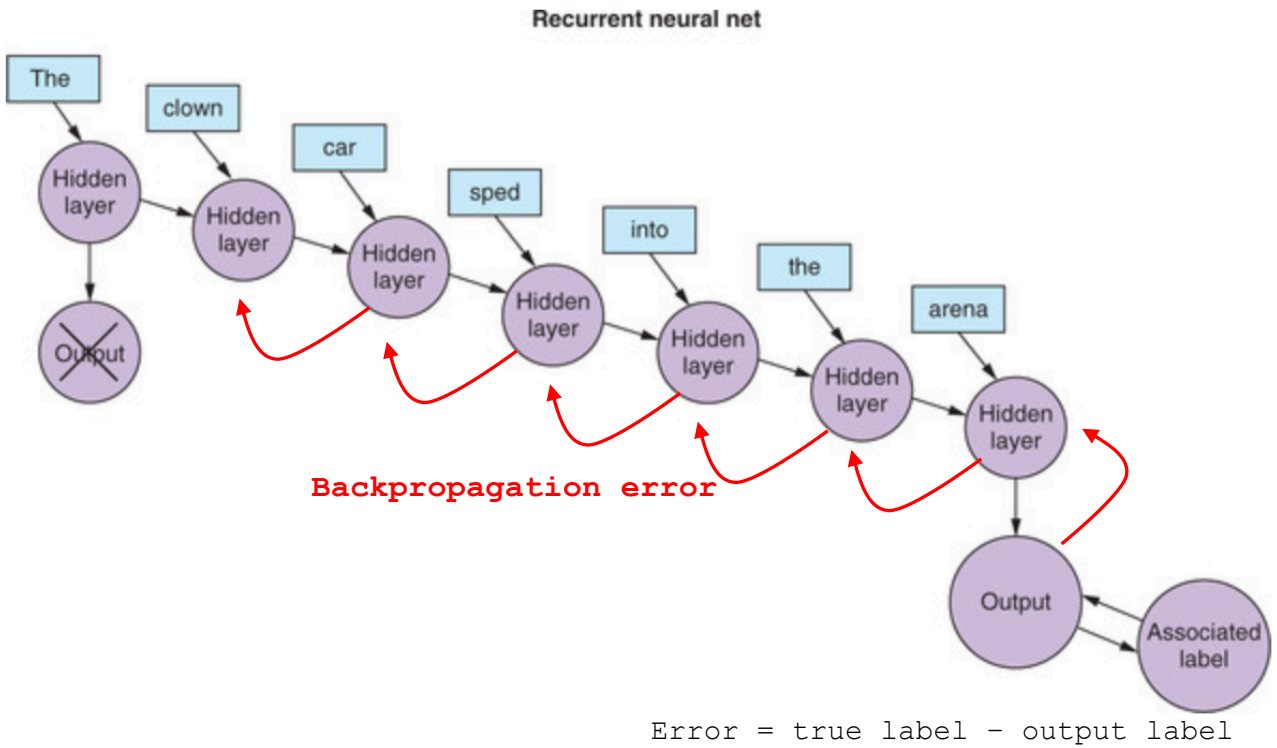


Figure from: Natural Language Processing in Action, Manning Publications

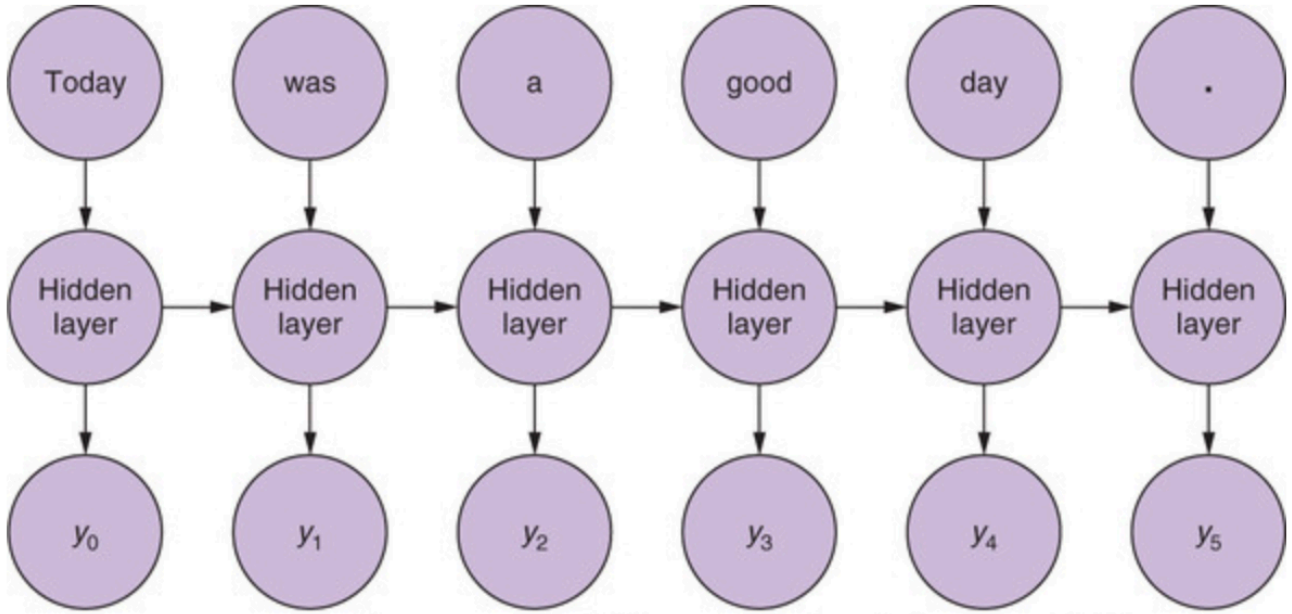
The error is calculated by comparing the difference between the generated label (output) and actual label.

At the final time step (token = arena),adjustment to the weights are calculated such that error is minimized but not applied yet.

The calculated weight adjustment is back propagate to the layer in the past time step (token = the). At this time, the inputs to the hidden layer were different, so further adjustments to the weights are calculated.

This continues until the backpropagation reaches the first time step (Token = The). The aggregated changes are then finally applied to the weight in the hidden layer.

What if we care for the intermediate output?



The earlier example ignores all intermediate output and is only interested in the final output

There are use cases where the intermediate output sequence matters

e.g. predicting next word, translation

Figure from: Natural Language Processing in Action, Manning Publications

Backpropagation

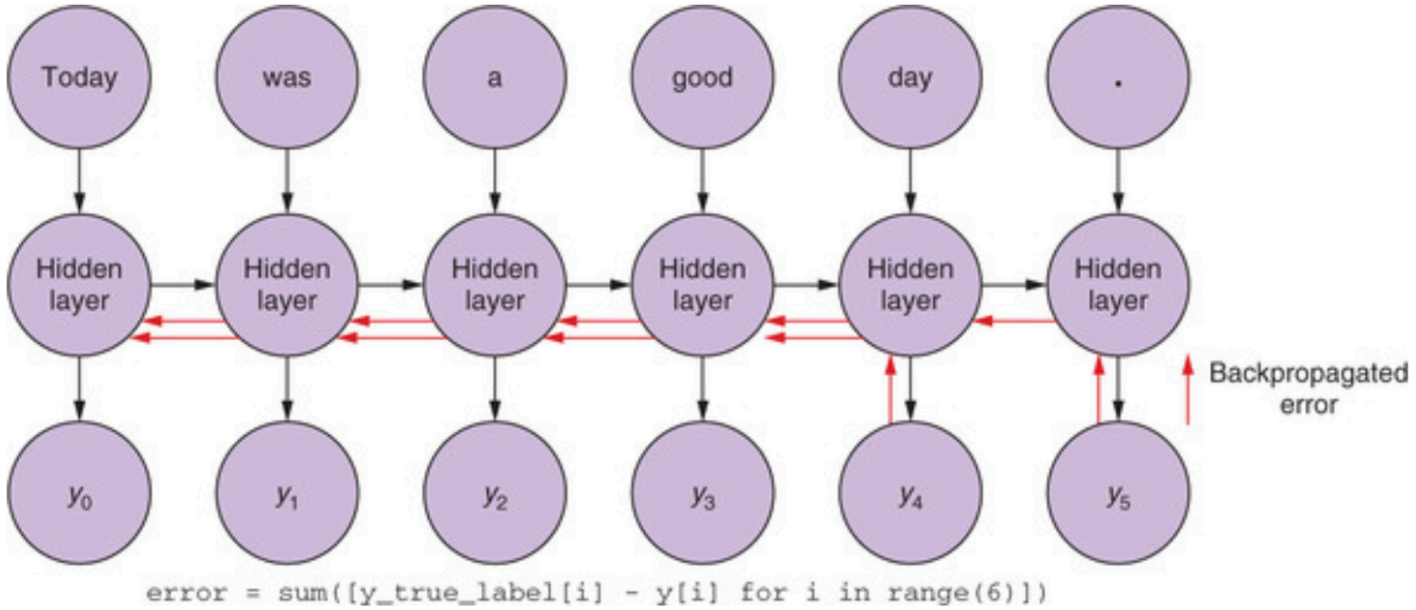


Figure from: Natural Language Processing in Action, Manning Publications

In this case, the error takes into account the total of all the errors between the actual labels and output y.

Starting with the final time step (token = .), the weights adjustment are calculated by working backwards from y5 to y0.

The next backpropagation starts with prior time step (token = day). Weight adjustments are calculated by working backwards from y4 to y0.

This process continues until the start of the sequence

The aggregated weight changes are then finally applied to the weight in the hidden layer.

How Do Human Read (II)?

THEY WANTED TO PET THE DOG WHOSE FUR IS BROWN

- Read forward until FUR → knows that the fur belongs the dog
- Continue reading to the end
- Refer back to the earlier part of the statement and conclude the dog has brown fur

Considering that natural language is a sequence of words, symbols and characters, what has been revealed in the natural course of reading is important for understanding the sequence

What comes after is also important for deepening the understanding the sequence

Bidirectional RNN

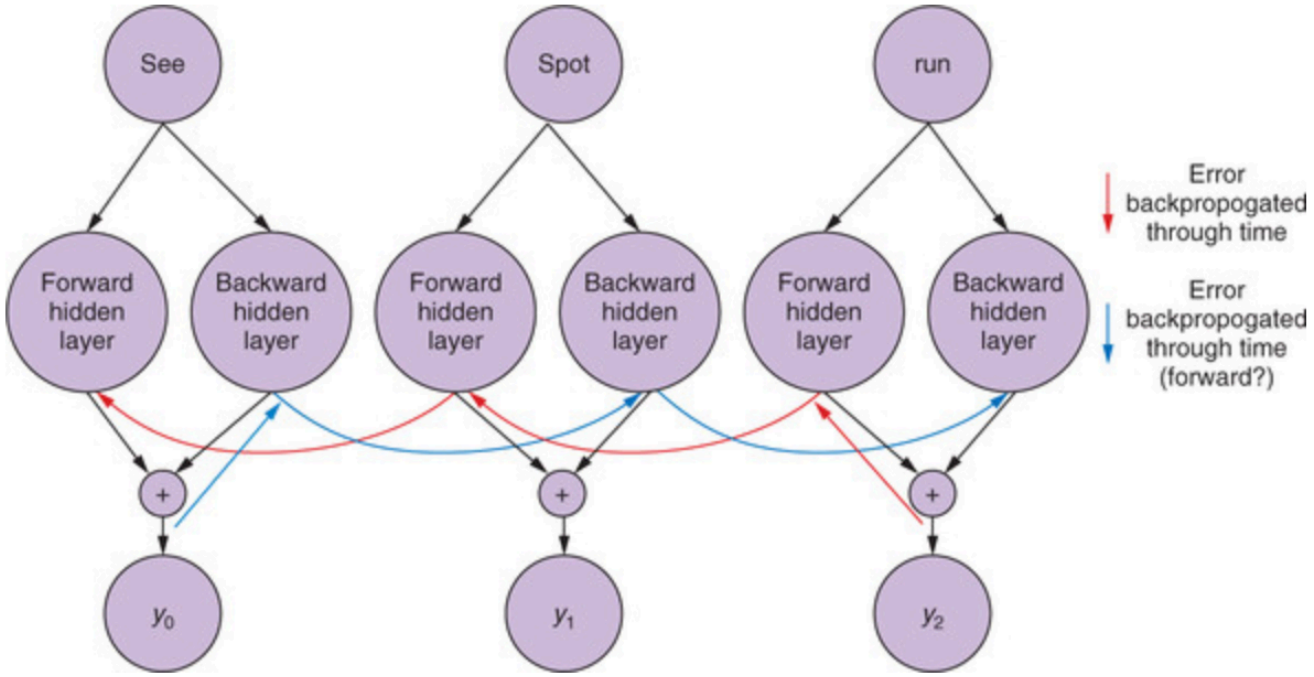


Figure from: Natural Language Processing in Action, Manning Publications

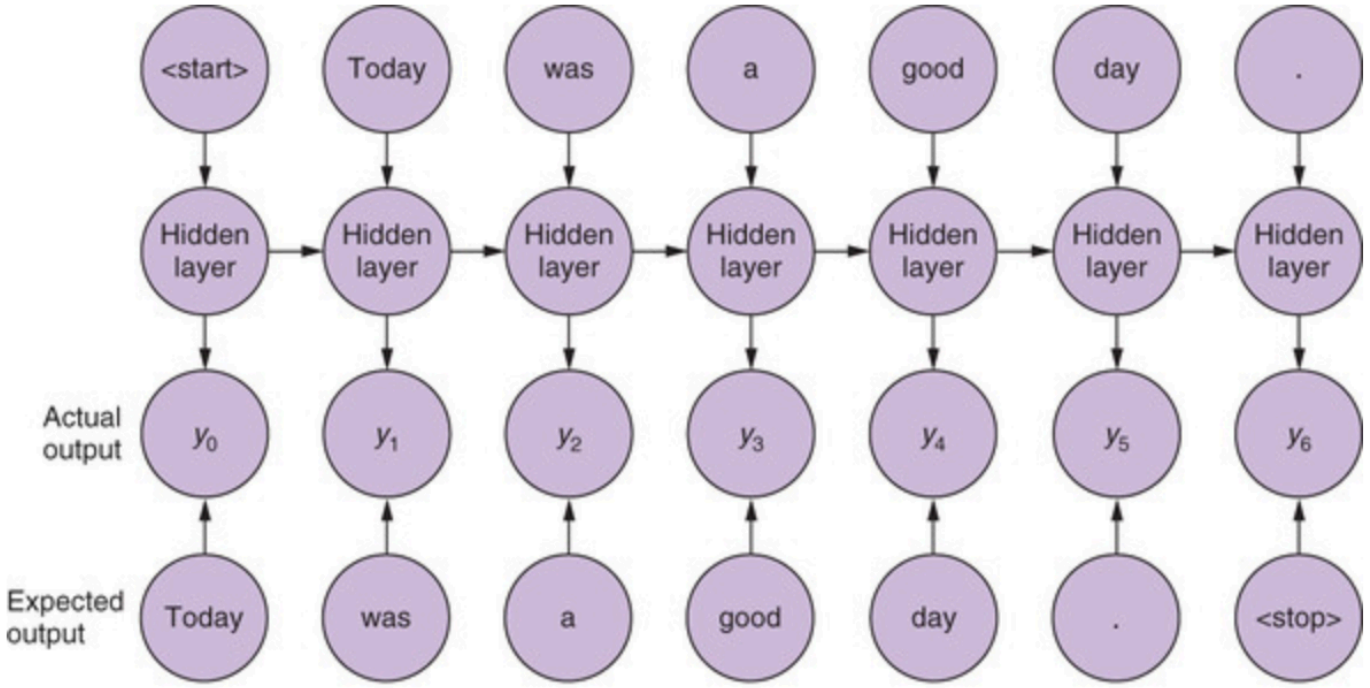
Two RNN layers are used to process the sequence of text

The first layer is the forward hidden layer, where the input tokens are feed in the way it is naturally read (e.g. See, Spot, Run)

The second layer is the backward hidden later. The input tokens are feed backwards starting with the last word (Run, Spot, See)

The output from both the forward and backwards layers for the **same** token are concatenated to generate the output

Language Model – Predicting the next word(s)



Expected output is the next token in the sample. Shown here on word level.

Training goal: For each token in the sample, you want the RNN to learn to predict the next token.

We can turn the input statement into the expected outcome by offsetting it

Today	was	a	good	day
⬇	⬇	⬇	⬇	
was	a	good	day	.

Figure from: Natural Language Processing in Action, Manning Publications

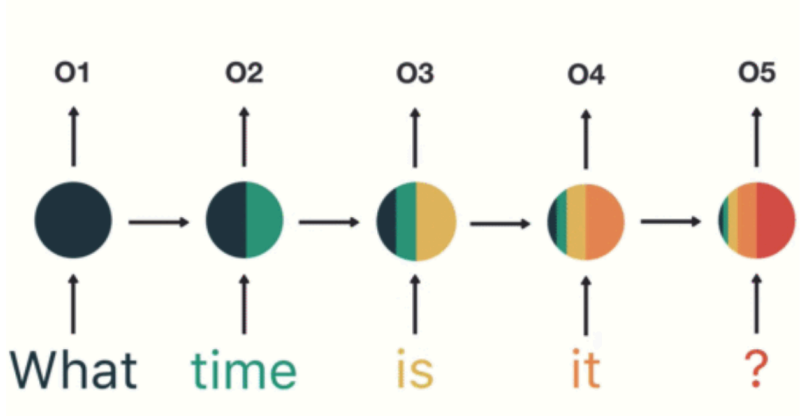
RNN limitations

In theory, RNN can remember forever

But as the network moves forward in time, "smaller bits" of the earlier input is remembered
> *At time step 3 (token = it) information about 'time' is almost lost*

During backpropagation, error signals from the later time step do not reach the earlier time step resulting in gradually smaller adjustments (the vanishing gradient problem)

In practice, LSTM (Long Short Term Memory) cells are used instead of RNN



Is RNN/LSTM Effective?

This method has been used to:

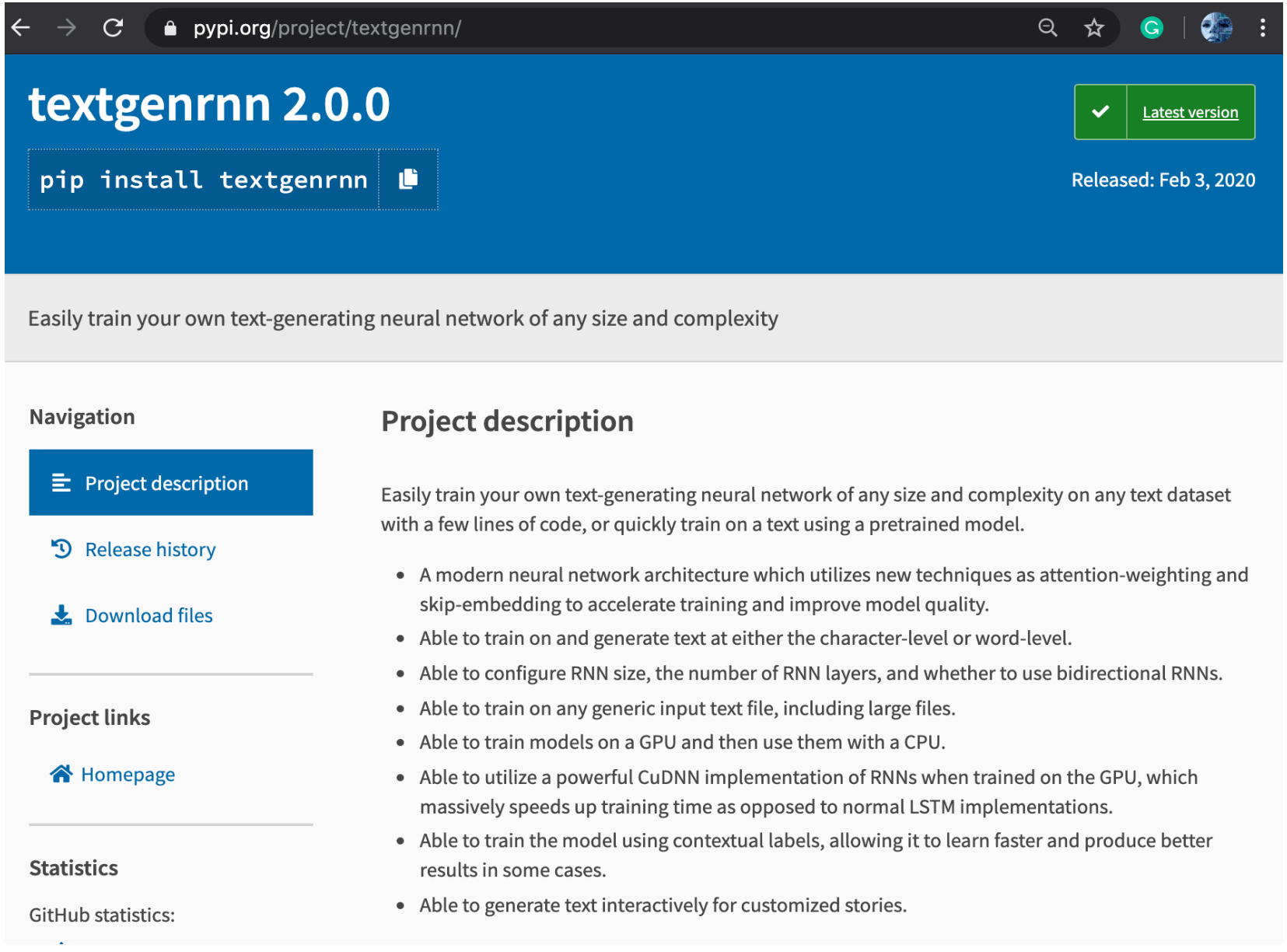
- Write fake Shakespeare plays (including character names and appropriate line breaks)
- Write fake Wikipedia articles in Markdown
- Write fake math proofs in LaTeX
- Learn C++ syntax (while writing mostly gibberish code)
- Generate baby names

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

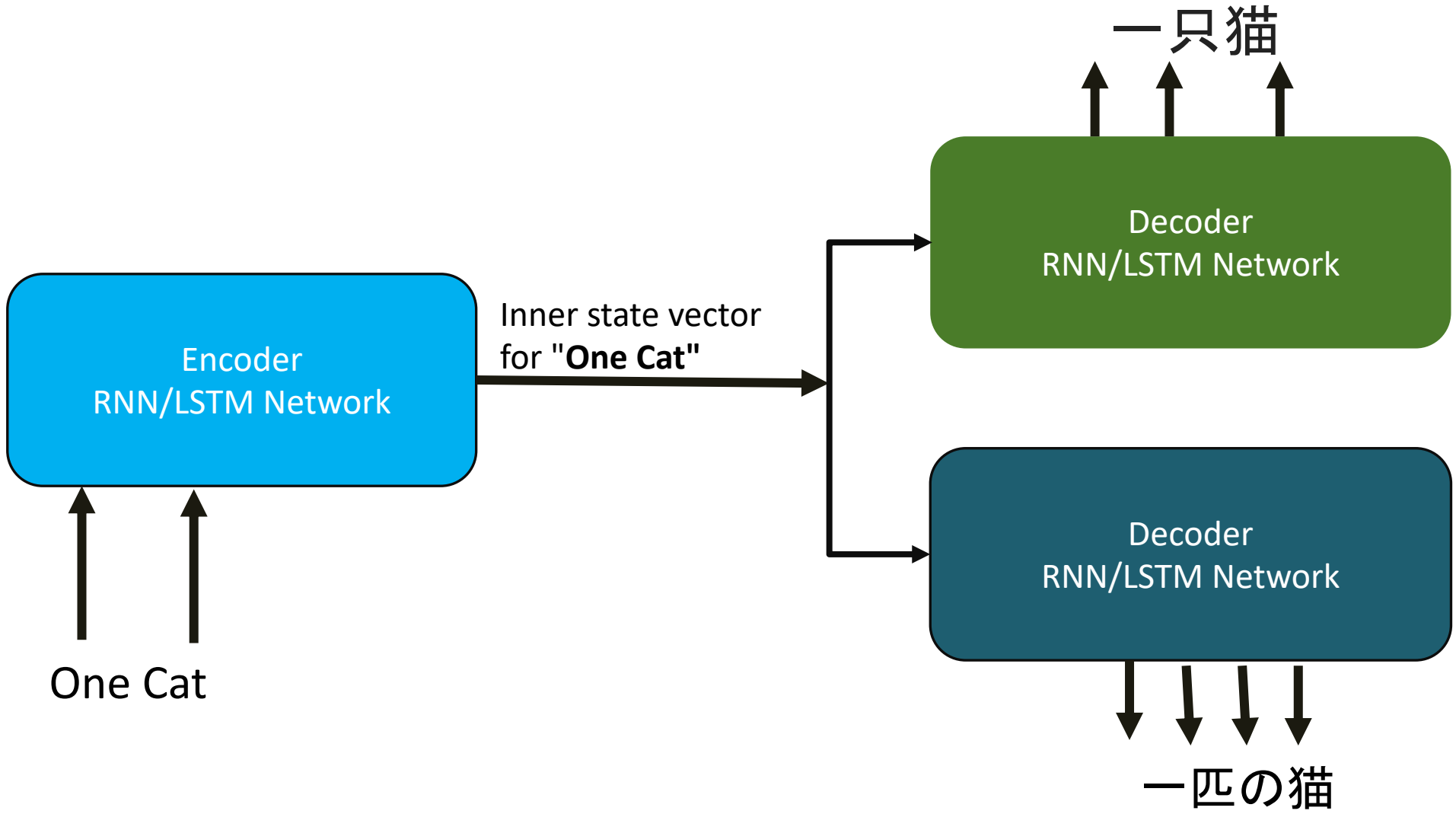
Exercise 2

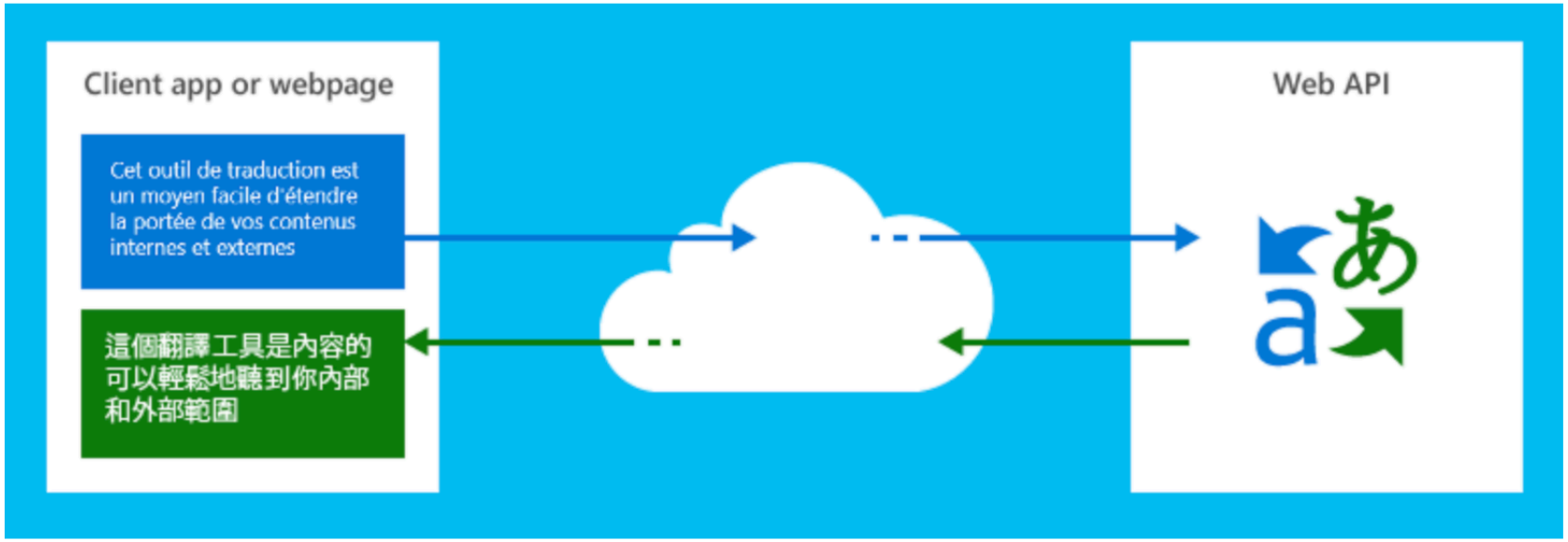
Text Generation using RNN/LSTM

Refer to Jupyter notebook:
ex2-text_generation_using_rnn_lstm.ipynb



Encoder/Decoder model with RNN/LSTM





The Microsoft Translator Text API and Microsoft Speech services, part of the **Cognitive Services** collection of APIs, are machine translation services from Microsoft.

Continuous improvements to translation are important. However, performance improvements have plateaued with SMT technology since the mid-2010s. By leveraging the scale and power of Microsoft's AI supercomputer, specifically the Microsoft Cognitive Toolkit, Microsoft Translator now offers neural network (**LSTM**) based translation that enables a new decade of translation quality improvement.

https://www.microsoft.com/en-us/translator/business/machine-translation/#nmt

Attention

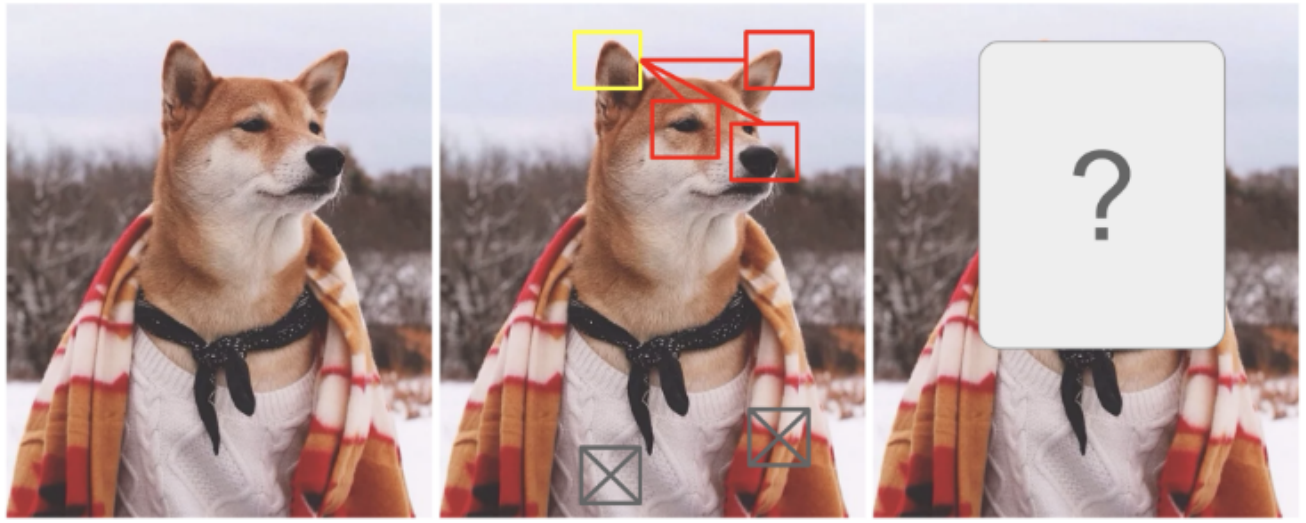


Fig. 1. A Shiba Inu in a men's outfit. The credit of the original photo goes to Instagram @mensweardog.

Visual attention allows us to focus on a certain region with "high resolution" (i.e. look at the pointy ear in the yellow box) while perceiving the surrounding image in "low resolution" (i.e. the snowy background and the outfit), and then adjust the focal point.

Ref: https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html

Attention

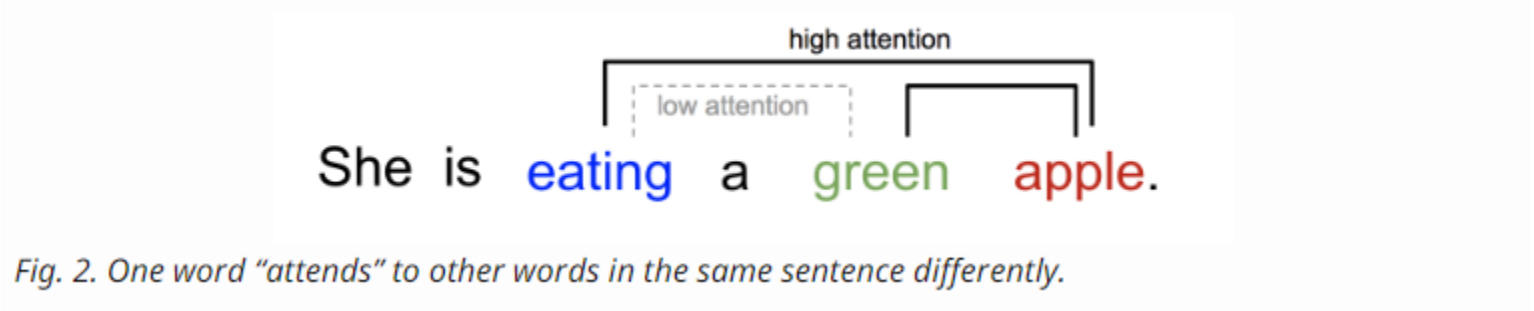


Fig. 2. One word "attends" to other words in the same sentence differently.

For text attention, we can explain the relationship between words in one sentence or close context. When we see "eating", we expect to encounter a food word very soon. The green and red terms describe the food, but green probably not so much with "eating" directly.

Ref: https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html

Attention

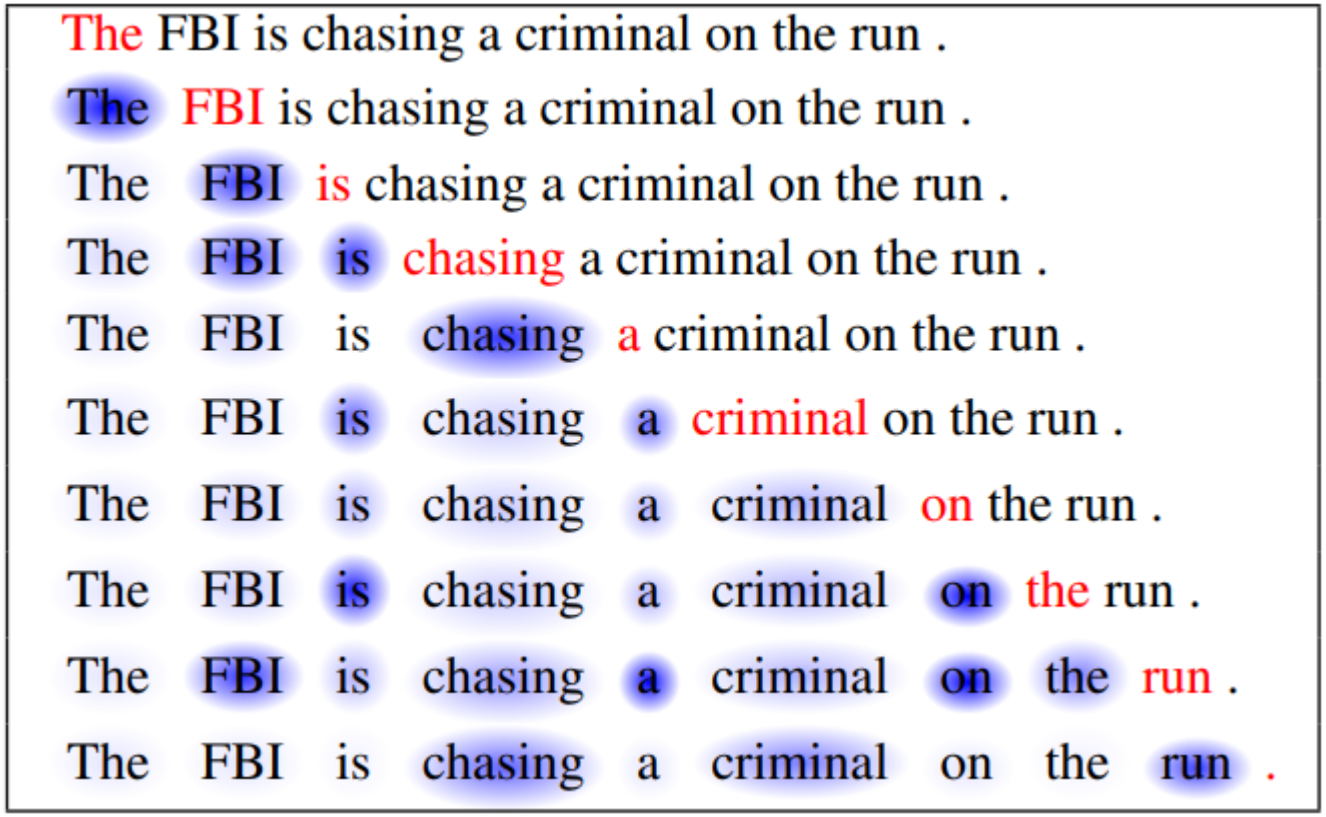


Image from: https://arxiv.org/pdf/1601.06733.pdf

Recent Development – Transformer model

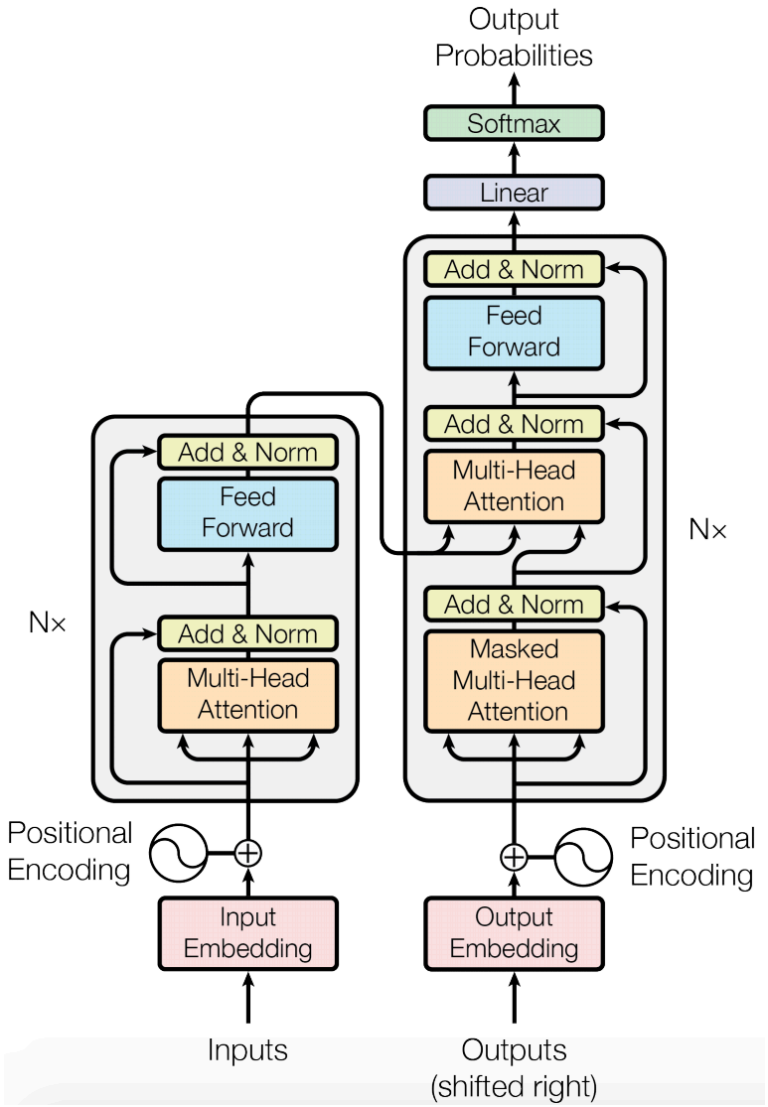
Attention Is All You Need

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin

(Submitted on 12 Jun 2017 (v1), last revised 6 Dec 2017 (this version, v5))

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks in an encoder-decoder configuration. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

https://arxiv.org/abs/1706.03762



Transformer model

- Like RNN/LSTM, the tranformer model is designed to handle tasks where the order of the text is important
- Does not require the sequence to be processed in order
- Facilities more parallelization and the use of larger set of data
- Makes possible transfer learning and development of **pre-trained models**
  - BERT (Bidirectional Encoder Representations from Transformers)
  - GPT (Generative Pretrained Transformer)

Pre-trained Model – GPT / GPT-2

"We've trained a large-scale **unsupervised** language model which generates coherent paragraphs of text, achieves state-of-the-art performance on many language modeling benchmarks, and performs rudimentary reading comprehension, machine translation, question answering, and summarization - all without task-specific training." ***Open AI research group***

- GPT-2 is trained with a simple objective: predict the next word, given all of the previous words within some text
- Transformer based language model with 1.5 billion parameters, 8 million web pages

Recycling is good for the world.

Recycling is NOT good for the world. It is bad for the environment, it is bad for our health, and it is bad for our economy. I'm not kidding. Recycling is not good for the environment. It is destructive to the earth and it is a major contributor to global warming. Recycling is not good for our health. It contributes to obesity and diseases like heart disease and cancer. Recycling is bad for our economy. It increases the cost of a product, and in turn, the price of everything that is made with that product. Recycling is not good for our nation. We pay a tremendous price for the privilege of having the world's most advanced and efficient recycling system. Recycling is a huge, colossal waste of time, energy, money, and resources. And THAT is why we need to get back to basics and get back to basics in our recycling efforts. One of the best ways to start is to look at the process of creating a paper product. When you make a paper product, it is basically a long chain of materials. Everything from the raw materials (wood, cardboard, paper, etc.), to the reagents (dyes, solvents, etc.) to the printing equipment (chemicals, glue, paper, ink, etc.), to the packaging, to the packaging materials (mercury, chemicals, etc.) to the processing equipment (heating, cooling, etc.), to the packaging materials, to the packaging materials that are shipped overseas and to the packaging materials that are used in the United States. Each step along the way creates tons of waste that we constantly have to clean up. The process of making a paper product is a very wasteful one. But the end result is something that all of us need to consume. And if we want to keep the recycling process running efficiently, then we really need to think about each and every step that goes into making a paper product.

In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

Pérez and the others then ventured further into the valley. "By the time we reached the top of one peak, the water looked blue, with some crystals on top," said Pérez.

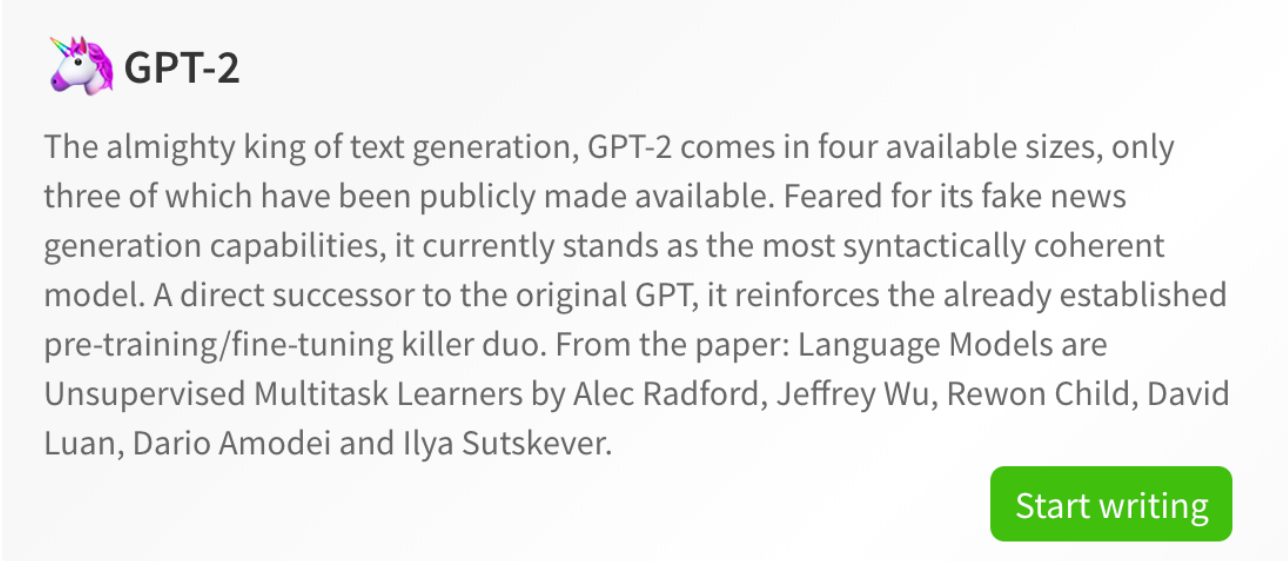
Pérez and his friends were astonished to see the unicorn herd. These creatures could be seen from the air without having to move too much to see them – they were so close they could touch their horns.

While examining these bizarre creatures the scientists discovered that the creatures also spoke some fairly regular English. Pérez stated, "We can see, for example, that they have a common 'language,' something like a dialect or dialectic."

Exercise 3

Use AI to write an appeal letter

https://transformer.huggingface.co/



Write a 300 words letter starting with:

Dear dog owners, please pick up your dog's poo whenever it completes its business.

References

- Intel AI Developer Program, https://software.intel.com/en-us/ai
- Hidden Markov Models Suitable for Text Generation , https://pdfs.semanticscholar.org/cb06/a81281a394a3eddd9e2cb8f409490c858782.pdf
- Natural Language Generation at Google Research (AI Adventures), https://youtu.be/MNvT5JekDpg
- The Unreasonable Effectiveness of Recurrent Neural Networks, http://karpathy.github.io/2015/05/21/rnn-effectiveness
- Natural Language Processing in Action, Manning Publications.
- Attention is all you need, https://arxiv.org/abs/1706.03762
- The illustrated Transformer, http://jalammar.github.io/illustrated-transformer/
- OpenAI GPT-2: Understanding Language Generation through Visualization , https://towardsdatascience.com/openai-gpt-2-understanding-language-generation-through-visualization-8252f683b2f8