

Intro to Neural Networks

BA865 – Mohannad Elhamod

Language Modeling

What is language modeling anyway?...

Web search engine / ...

I saw a cat|

I saw a cat on the chair

I saw a cat running after a dog

I saw a cat in my dream

I saw a cat book

[Lena-votta](#)

Send

▼

To

Cc

Add a subject

Greetings,

I would like to

Tab

Mohannad Elhamod

Clinical Assistant Professor

Boston University | Questrom School of Business

elhamod@bu.edu

QUESTROM

MEANS

BUSINESS.

What is language modeling anyway?...

I grabbed the **branch** and broke it.

I went to the **branch** and deposited some money.

Context matters!

What is language modeling anyway?...

- I went to ____.
- I woke up at 7 am and went to ____.
- I woke up at 7 am, packed my book and notebook, and went to ____.

More context leads to more certainty

What is language modeling anyway?...

I went to the **branch** and deposited some money.

I went to the **bank** and deposited some money.

I went to the **ATM** and deposited some money.

Words which frequently appear in **similar contexts** have **similar meaning**.

[Lena-volta](#)

Formalizing our thoughts

- It seems we process language sequentially**.
- So, language modeling is the chaining of word probabilities.

$P(\text{I saw a cat on } \dots) =$

$P(\text{I}) \cdot P(\text{saw}|\text{I}) \cdot P(\text{a}|\text{I saw}) \cdot P(\text{cat}|\text{I saw a}) \cdot P(\text{on}|\text{I saw a cat}) \cdot \dots$

Probability of I saw a cat on

[Lena-volta](#)

- How do we calculate these probabilities?

counting...

$$P(\text{cat}) = \frac{N(\text{"cat" in corpus})}{N(\text{all words in corpus})}$$

$$P(\text{cat} | \text{my}) = \frac{N(\text{"my cat" in corpus})}{N(\text{"my" in corpus})}$$

Can you foresee any problem with this calculation?...

N-grams

Instead, let's just use a context of *fixed-length*.

$P(\text{I saw a cat on a mat}) =$

- $P(\text{I})$
- $P(\text{saw} \mid \text{I})$
- $P(\text{a} \mid \text{I saw})$
- $P(\text{cat} \mid \text{I saw a})$
- $P(\text{on} \mid \text{I saw a cat})$
- $P(\text{a} \mid \text{I saw a cat on})$
- $P(\text{mat} \mid \text{I saw a cat on a})$

- $n=3$ (trigram model): $P(y_t \mid y_1, \dots, y_{t-1}) = P(y_t \mid y_{t-2}, y_{t-1})$,
- $n=2$ (bigram model): $P(y_t \mid y_1, \dots, y_{t-1}) = P(y_t \mid y_{t-1})$,
- $n=1$ (unigram model): $P(y_t \mid y_1, \dots, y_{t-1}) = P(y_t)$.

[Lena-volta](#)

N-grams

Context is like a sliding window into the past.

Hugging Face is a startup based in New York City and Paris

$p(\text{word})$

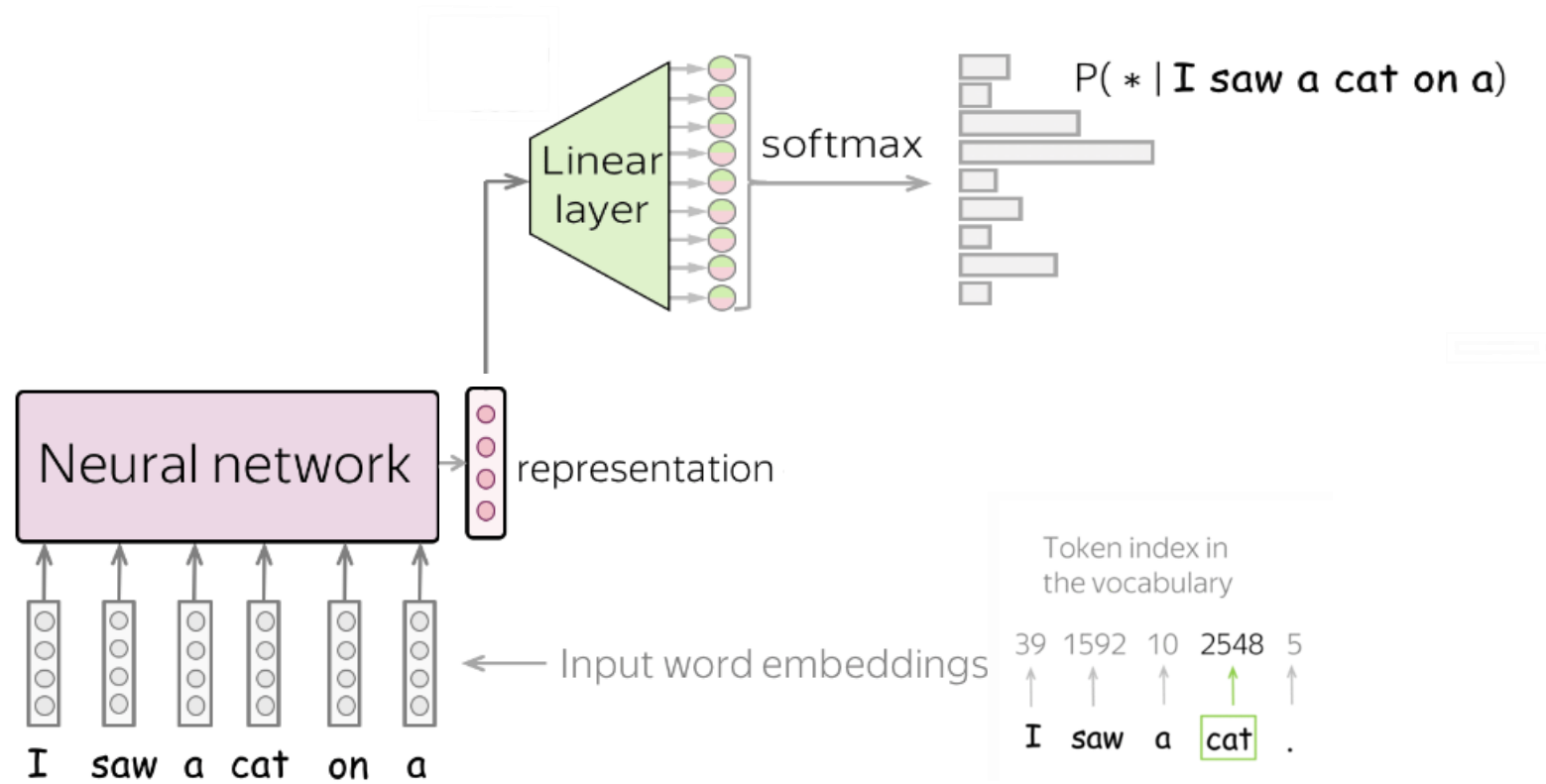
[Huggingface](#)

Context size

- I went to the beach...
 - My wife sat next to me. She was replying to some emails, and...
 - the bird stole our sandwich. Then...
 - it started raining suddenly and ____.
-
- Longer context: predictable outcome.
 - Shorter context: Too unpredictable.

Neural Networks for Language Modeling

General Model Architecture

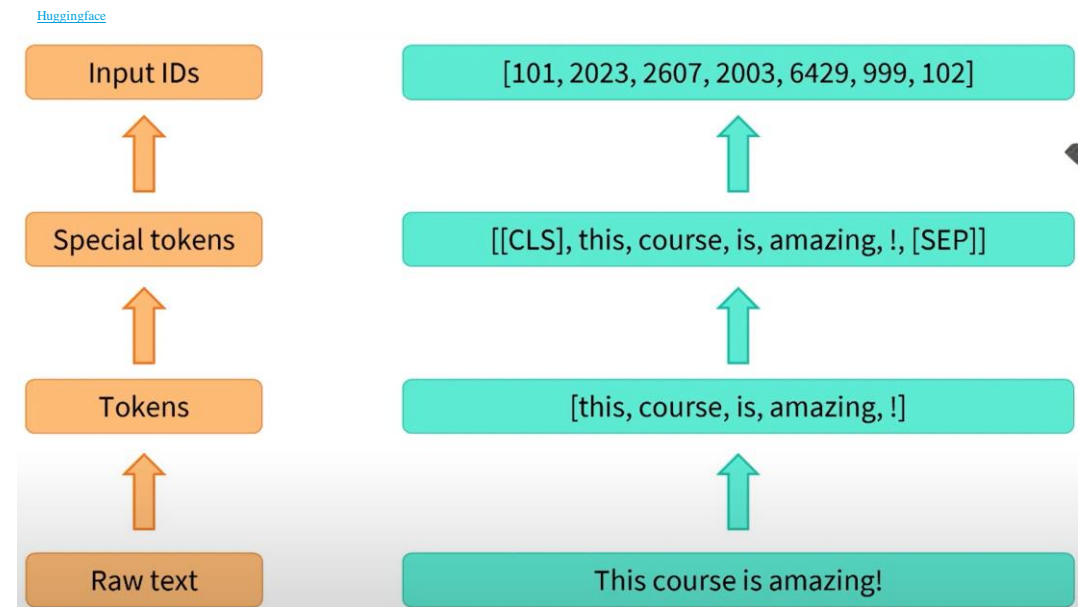


Can you see any issue with inputting words into an NN?

[Lena-voita](#)

Tokenization

- Computers only understand numbers.
- We need to convert the text into tokens (e.g., words).
- Can we use the Input IDs as representation?
- Is it a good representation?

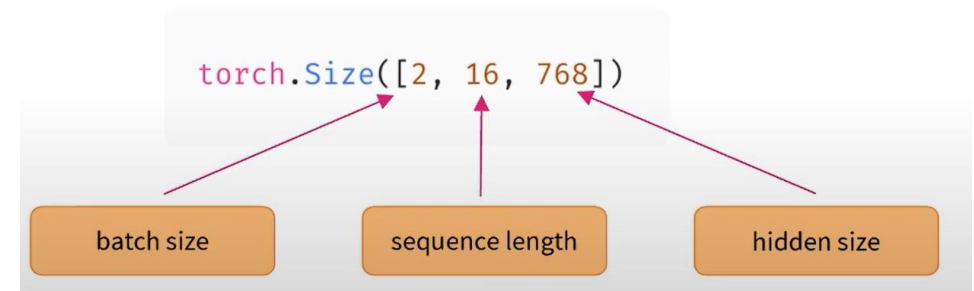


Syntax vs. Semantics

- Notice that:
 - Different tokens might have the same meaning (e.g., like, enjoy)
 - The same token might have different meanings (is *like* something, to *like* something)
- **So, while tokens represent syntax, we really care about meaning/semantics.**
- In many cases, you can only get the meaning through **context** (i.e., the token's place with respect to other tokens.)

Vectorization

- Once tokenized, we convert the tokens into vectors.
- So, if we have a dataset of sentences, we represent them as:
 - A batch of sentences (i.e., batches)
 - Each sentence is represented as a sequence of tokens (sequence length)
 - Each token is represented as a vector (hidden size)



Word Embeddings

- We ideally want words that have similar meanings to have smaller distances.
- [Demo](#)
- Examples:
 1. [Word2Vec \(Google\)](#)
 2. [GloVe \(Stanford\)](#)
 3. [Train your own!](#)

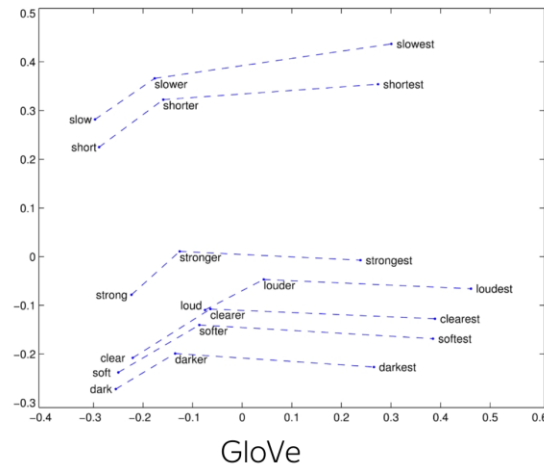
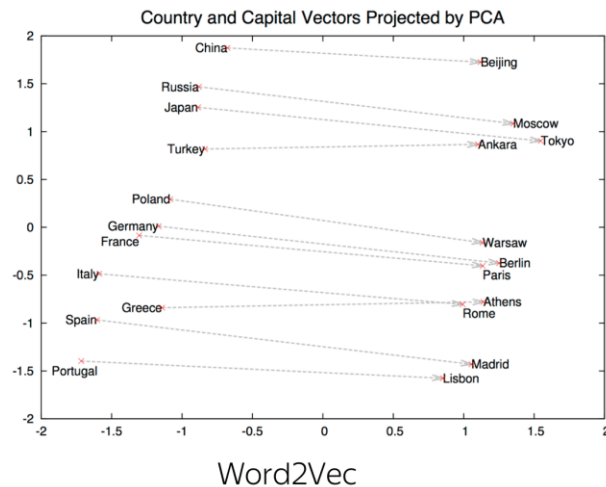
Word embeddings

Word embeddings can also be used find directionality in the corpus:

- [Demo 1 \(semantics\)](#)
- [Demo 2 \(vector view\)](#)
- [Demo 3 \(dimensionality\)](#)

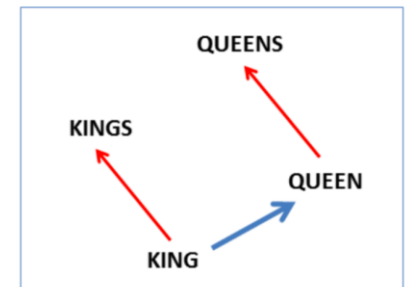
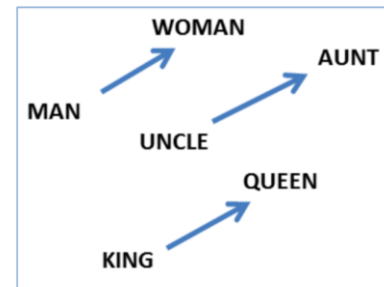
Word embeddings

Word embeddings can also be used find directionality in the corpus.



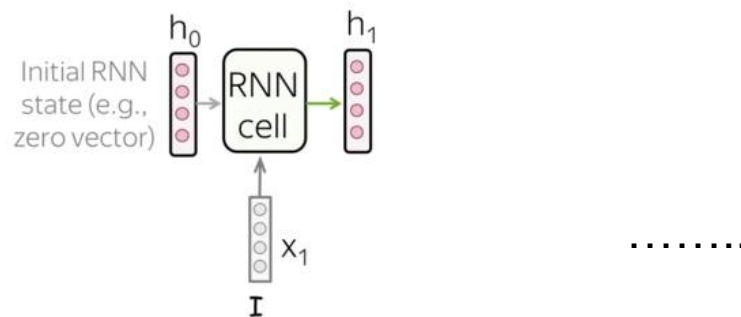
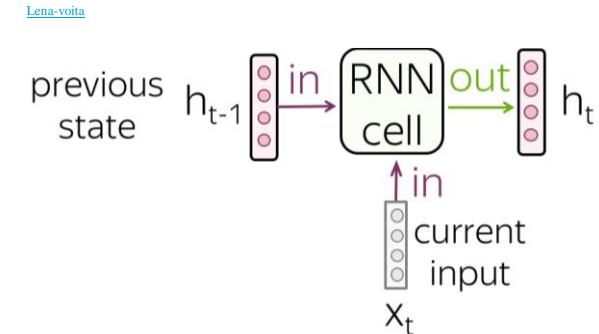
semantic: $v(\text{king}) - v(\text{man}) + v(\text{woman}) \approx v(\text{queen})$

syntactic: $v(\text{kings}) - v(\text{king}) + v(\text{queen}) \approx v(\text{queens})$

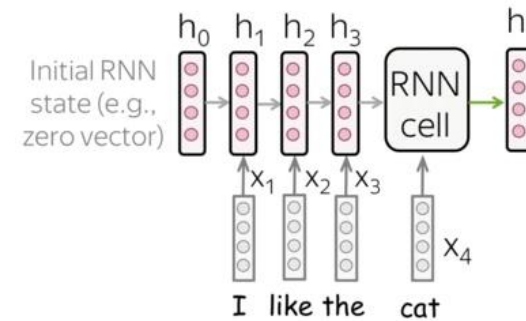


Recurrent Neural Nets (RNNs)

- Wouldn't it be nice if the model kept a "memory" of the previous state?
- The model could combine the embedding of the "memory" and current word → gives next word.



.....



Text: I like the cat on a mat <eos>

↑
we are here

not read yet

Text: I like the cat on a mat <eos>

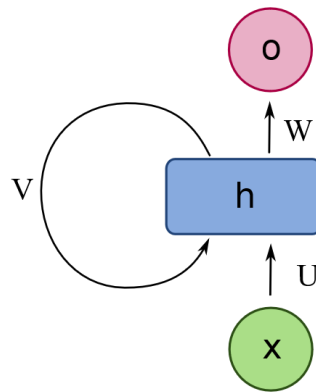
↑
we are here

not read yet

What is an “RNN cell”?

It is just a set of matrices:

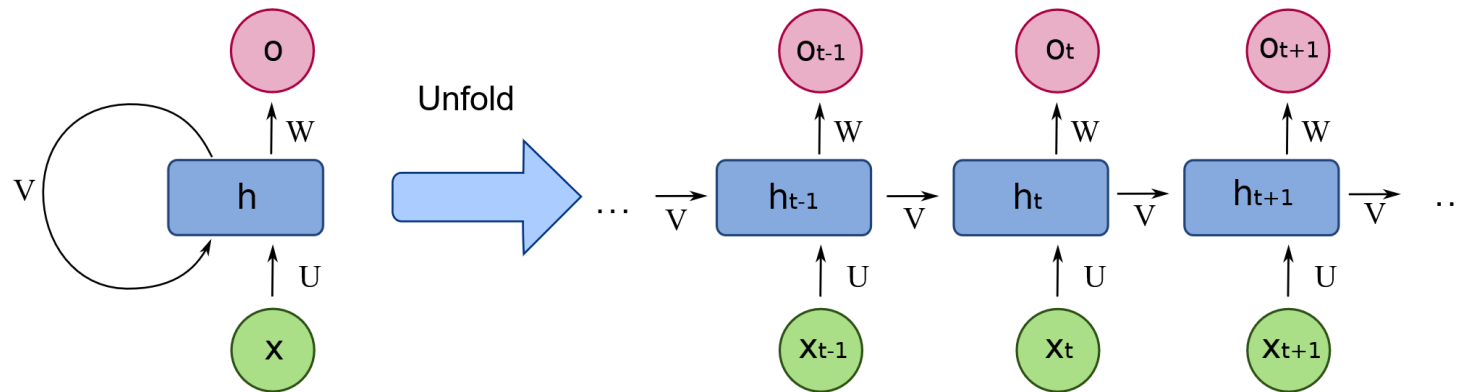
- U encodes current input into an embedding.
- V transforms the embedding representing the context up to this point.



What is an “RNN cell”?

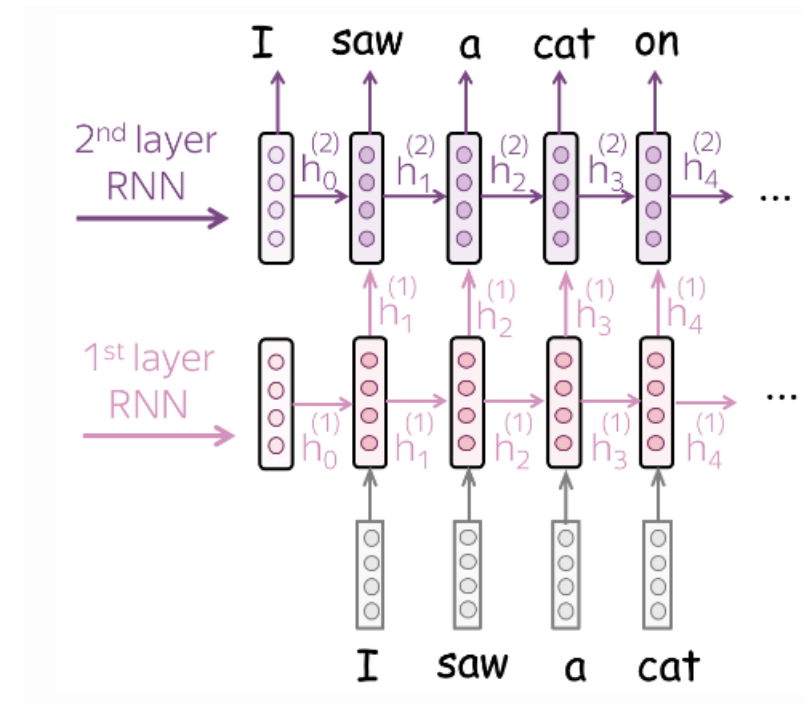
It is just a set of matrices:

- h is a new embedding representing a combination of the context embedding and the current word embedding.
- W decodes the new embedding back into a prediction.



Recurrent Neural Nets (RNNs)

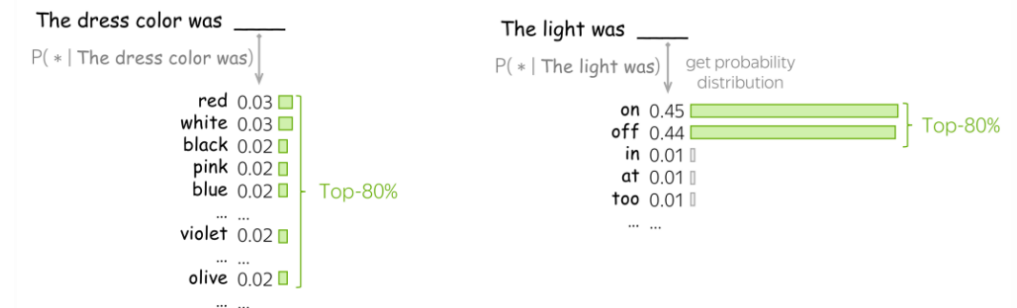
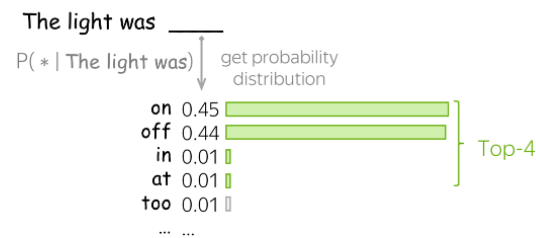
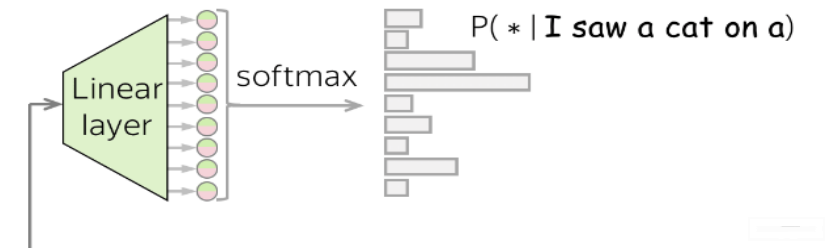
- We can add more layers and units per layer to increase complexity.
- [Demo](#)



[Lena-voita](#)

Sampling The Distribution

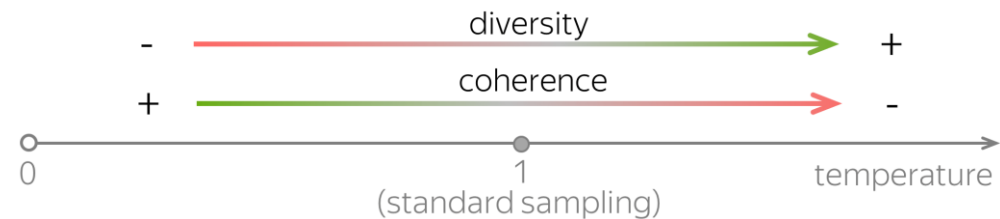
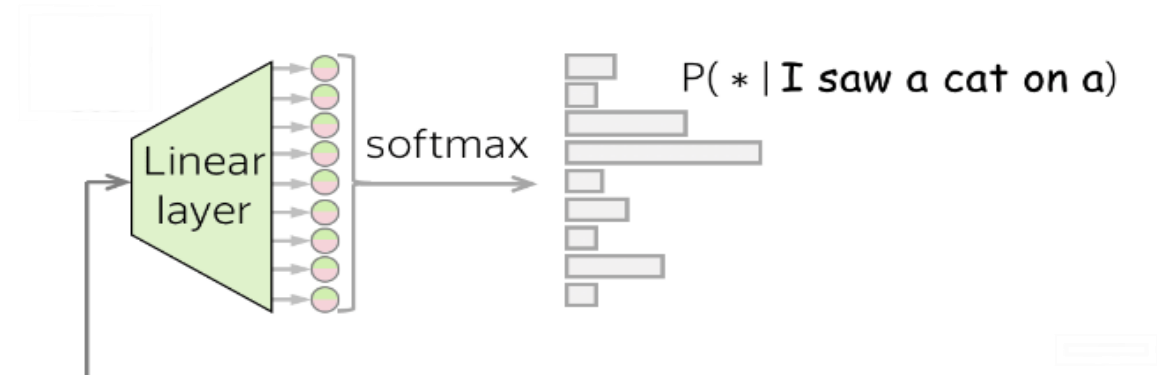
- Always take top probability?
 - That makes the model deterministic (no creativity).
- Alternative?
 - Sampling, Top-k, or top-p.



[Lena-voita](#)

Sampling The Distribution

- Some words have way higher probability than others.
- This can be manually tuned through **temperature**.
- [Demo](#)



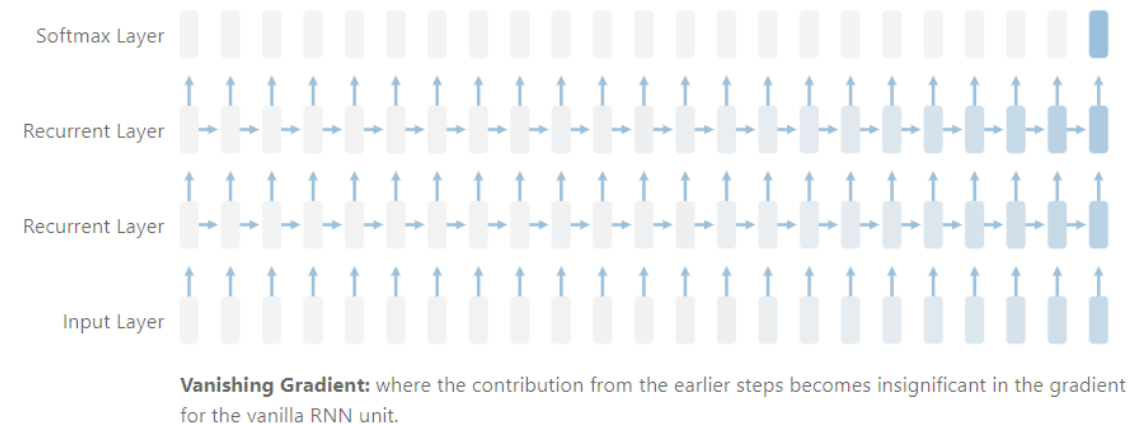
[Lena-voita](#)

Measuring The Metric

- What are we looking for?
 - A model that is not surprised by the new text it seen.
- We use perplexity.
 - Takes values between 1 and number of possible tokens.
 - Smaller is better.
 - [Demo](#)

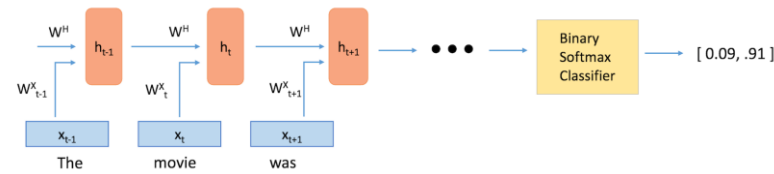
RNNs (issues)

- Gradient becomes insignificant for long contexts
 - The network forgets early words...
 - It is called the “vanishing gradient” or “memorization” problem.
 - RNNs have an issue memorizing long contexts.

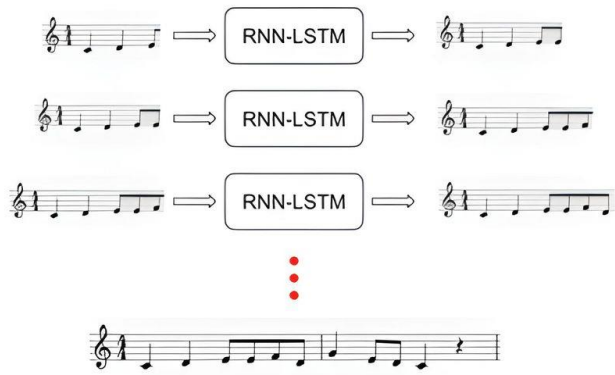


[distill.pub](#)

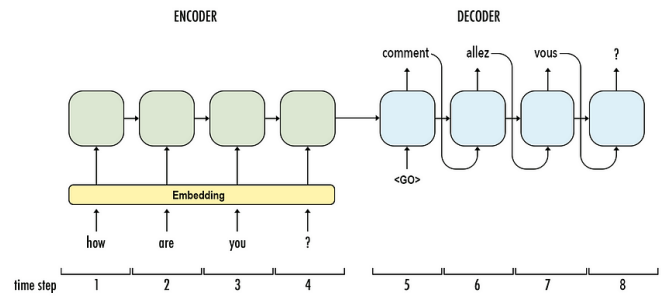
This Idea Also Applies Elsewhere...



[Adit Deshpande](#)



[aimc2023](#)



[Thomas Tracey](#)



Other ways to handle sequences...

- You can even use convolution!
- We will explore more sophisticated networks later...

