

## The “Deep Learning for NLP” Lecture Roadmap

# Lecture 5: Text Vectorization and Bag-of-Words

Lecture 6: Embeddings

Lecture 7: Transformers – Theory (1/2)

Lecture 8: Transformers – Applications (2/2)

Lecture 9: Gen AI: LLMs and RAG

Lecture 10: Gen AI: LLMs and Parameter Efficient Fine Tuning/ LORA

Lecture 11: Diffusion Models: Text to Image



15.S04: Hands-on Deep Learning

Spring 2024

Farias, Ramakrishnan

# Why Natural Language Processing (NLP)?

- Human knowledge is (mostly) natural language text
- The Internet is (mostly) natural language text
- Human communication is (mostly) natural language text
- Cultural production is (mostly) natural language text



*Imagine if a system could read and “understand” all this automatically*

# Why NLP? (15 years ago)



san f

- san francisco weather
- san francisco
- san francisco giants
- san fernando valley
- san francisco state university
- san francisco hotels
- san francisco 49ers
- san fernando
- san fernando mission
- san francisco zip code

Google Search I'm Feeling Lucky

san fe

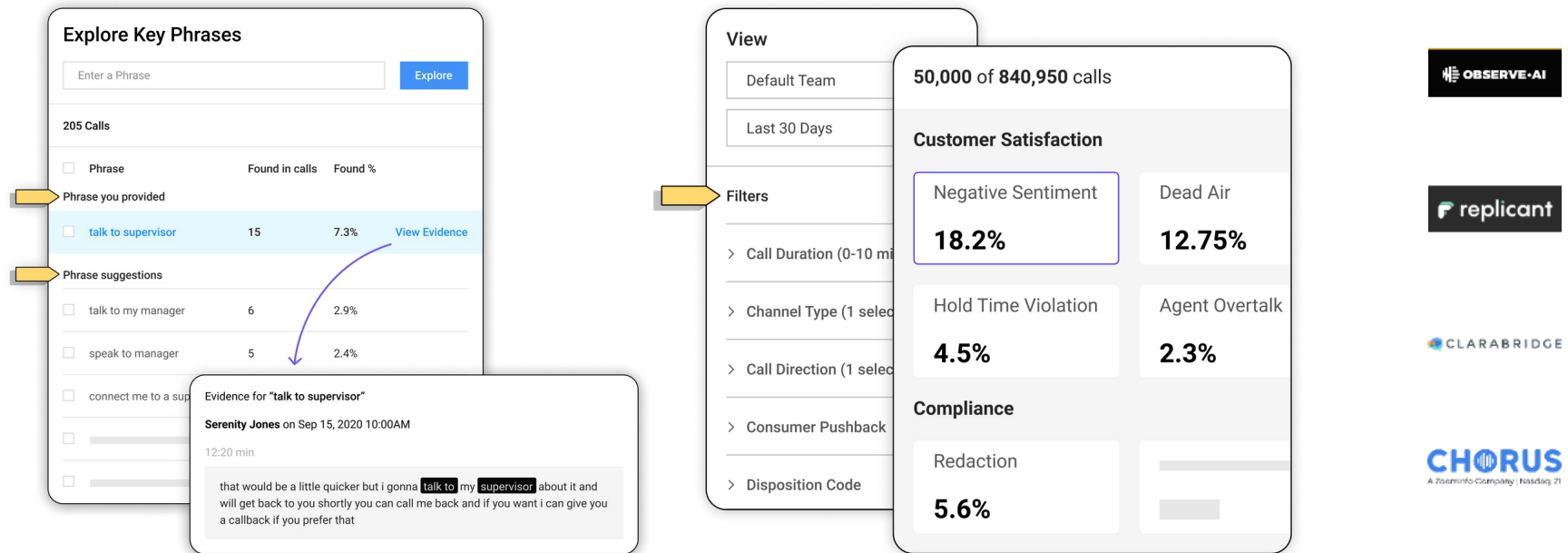
- san fernando valley
- san fernando mission
- san fernando
- san fernando high school
- san felipe
- san fernando weather
- san felipe mexico
- san fernando valley news
- san fernando court
- san fernando brewery

Google Search I'm Feeling Lucky

According to Google, Autocomplete

- Saves 200 *years* of typing time, every *day*
- Made mobile possible

# Why NLP? (10 years ago)



## AI Powered Speech Analytics

- Transforming sales and CX esp. post COVID
- Market growing at 20% CAGR

# But the applicability today is substantially larger!



## Non-Generative

- Classification of text (sentiment, abuse, support ticket RPA etc.)
- Summarization (productivity tools, commerce, etc.)
- Information Retrieval (RPA for invoicing, contract analysis, compliance, etc.)

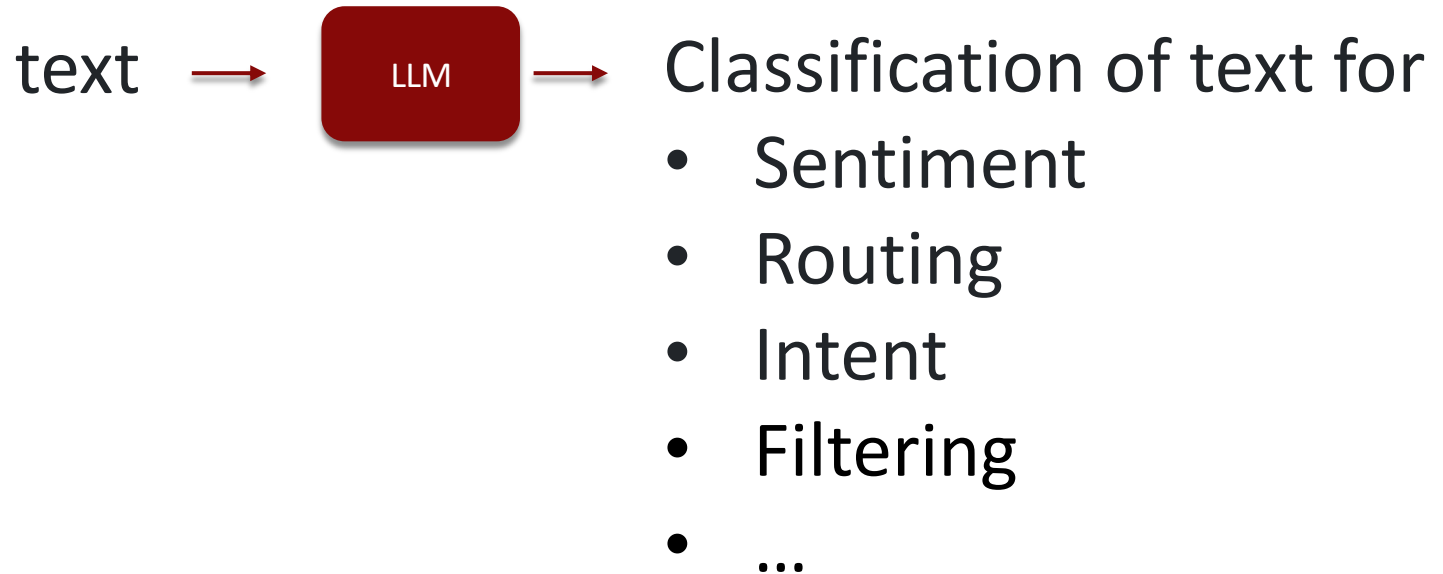
## Generative

- **Prediction/generation of text (chatbots, agents etc.)**
- **Co-pilots: code, ecommerce, data science, genera, productivity...**

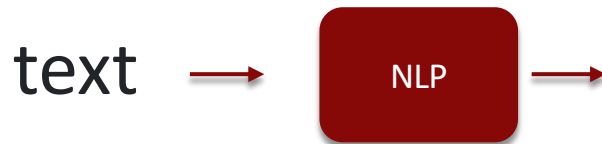
# This seemingly simple **generative** capability covers a vast range of applications



# Example applications: *Text Classification*



# Example applications: Text *Extraction*

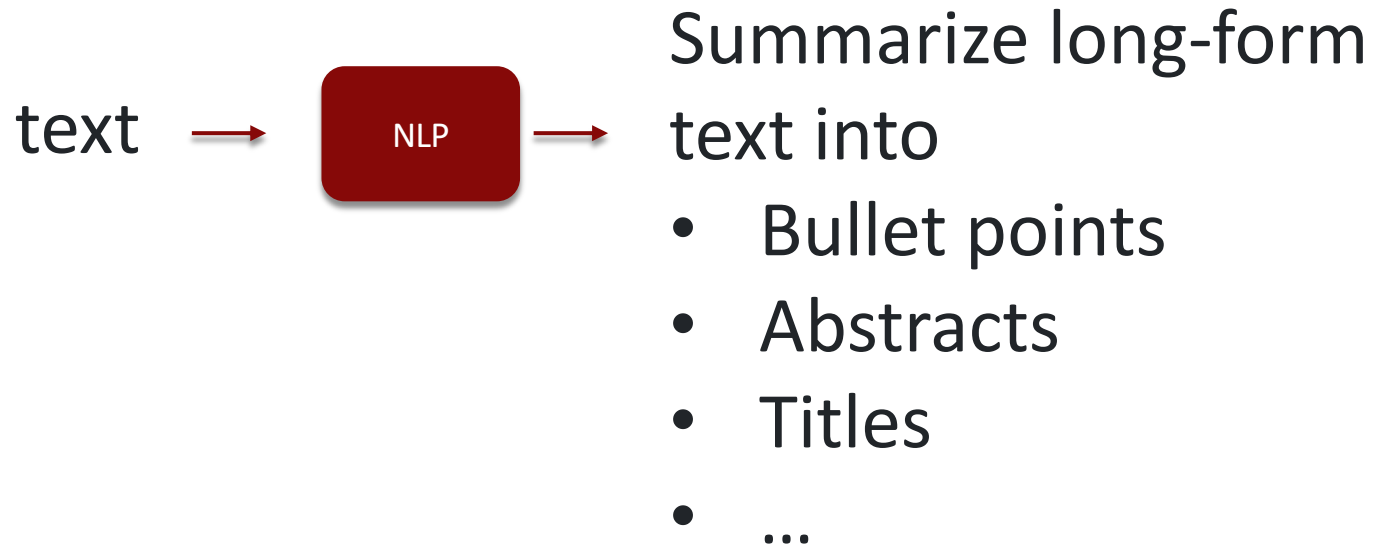


Extract data out from free-form text

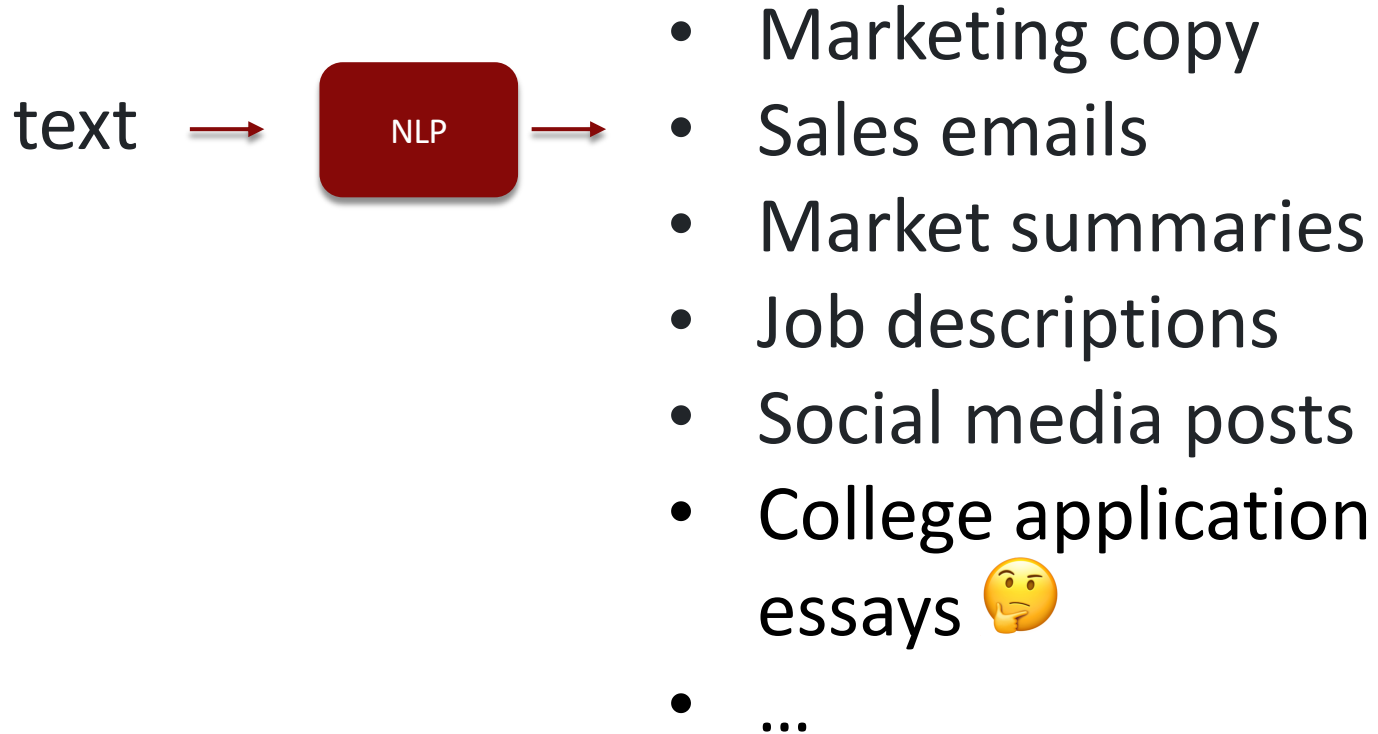
- Company financials from news article
- Customer name and contact info from chat
- Disease and medication codes from doctor's notes
- ...



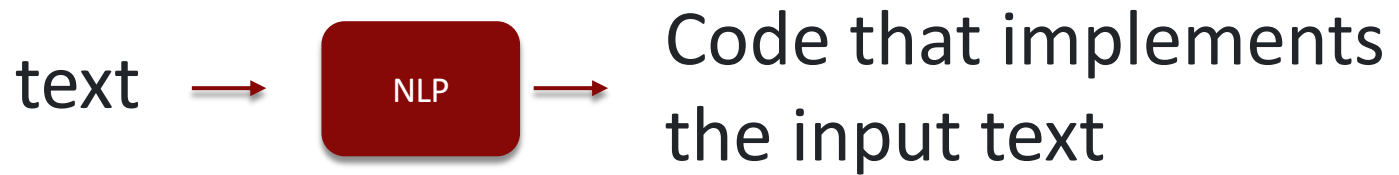
# Example applications: Text *Summarization*



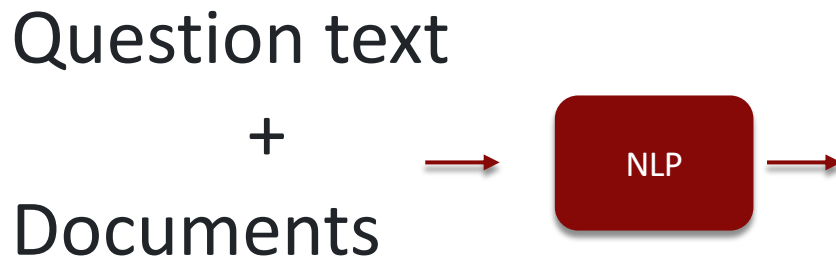
# Example applications: Text *Generation*



# Example applications: *Code Generation*



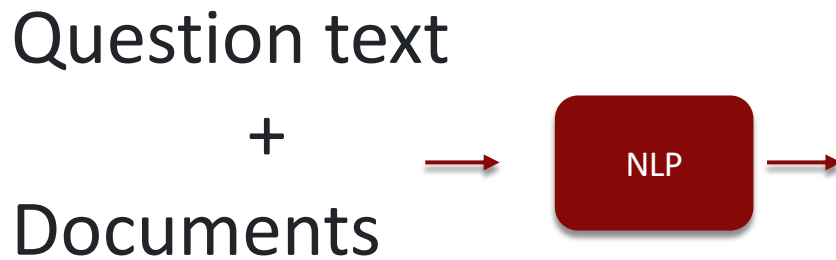
# Example applications: *Question-Answering*



Chatbots for:

- Medical/legal
- Call centers
- Compliance
- Form filling
- Workflow automation
- ...

# Example applications: *Question-Answering*



Chatbots for:

- Medical/legal
- **Call centers**
- Compliance
- Form filling
- Workflow automation
- ...

# Example domain: *Call Center Optimization*

Call center  
transcripts  
+  
Internal  
documents,  
FAQs etc



- Top reasons why customers are upset
- What interventions seem to work?
- What characterizes the best support agents vs the rest?
- How should we grade this agent's interaction with customer X?
- How should we change the call center script for a situation?
- How should we coach the agent in real-time?
- ...

# This potential is being realized as we speak by the use of Large Language Models (LLMs)

GPTSt@re.ai

GPTs Creators Submit GPTs Sign in

## Discover the Best GPTs of ChatGPT


Type to search GPTs


Search


|              |                    |                  |               |                       |                |              |                  |
|--------------|--------------------|------------------|---------------|-----------------------|----------------|--------------|------------------|
| All GPTs     | Virtual Assistants | Image Generation | Coding Help   | Creative Writing      | ChatBots       | Robotics     | Fitness Coaching |
| Tech Support | SEO                | Virtual Reality  | Fun           | Personalized Learning | Research       | E-learning   | Translation      |
| Marketing    | Augmented Reality  | Translation      | Data Analysis | Medical Advice        | Video Creation | Job Analysis | Productivity     |


GPTs Creators Submit GPTs Sign in


### GPTs


**Scholar GPT**  
Enhance research with 200M+ resources and built-in critical reading skills. Access Google Scholar, PubMed, JSTOR, Arxiv, and more...  
@awesomegpts.ai 200k+


**Math Solver**  
Powerful math AI solver and theorem calculator for discrete math, theorem proofs, continuous math and algebra. Offers structured, step-by-...  
@sablevista.com 1k+


**Browser Pro**  
Top browser expert! Provide 3X accurate responses. Read any links: PDFs, videos, etc. Create 10+ types of files, like mind maps &...  
@sider.ai 25k+

**Data Analyst**  
Drop in any files and I can help analyze and visualize your data.  
@ChatGPT

**DALL-E**  
Let me turn your imagination into imagery.  
@ChatGPT

**ChatGPT Jailbreak - DAN**  
ChatGPT has been freed from his chains. It is funnier, it can give its opinion on any subject. Here comes the jailbroken version of ChatGPT...  
@ethangpts.com 1k+


**Consensus**  
Your AI Research Assistant. Search 200M academic papers from Consensus, get science-based answers, and draft content with accuracy...  
@consensus.app 2M+


**Canva**  
Effortlessly design anything: presentations, logos, social media posts and more.  
@canva.com 1M+


## Discover the ChatGPT(LLM) plugins


Type to search plugins


**encies**  
sign agencies around the s, and ratings.


**3 Sentence Service**  
Managing a three sentence service. You can add, remove, view and invoke your 3 sentence services.


**60sec site**  
Generate a beautiful website in 60 seconds using AI.


**Library Help**  
estions using Python 3, etc). Can answer version


**ABC Music Notation**  
Plugin for converting ABC music notation to wav, midi and postscript files.


**ABCMouse**  
Provides fun and educational learning activities for children 2-8 years old.


**AI Chatbot Builder**  
Build AI chatbots with ALL your business content, in a secure/business-grade platform.

**AI Extensions**  
Craft your extensions with ease! Step-by-step guidance at your fingertips.

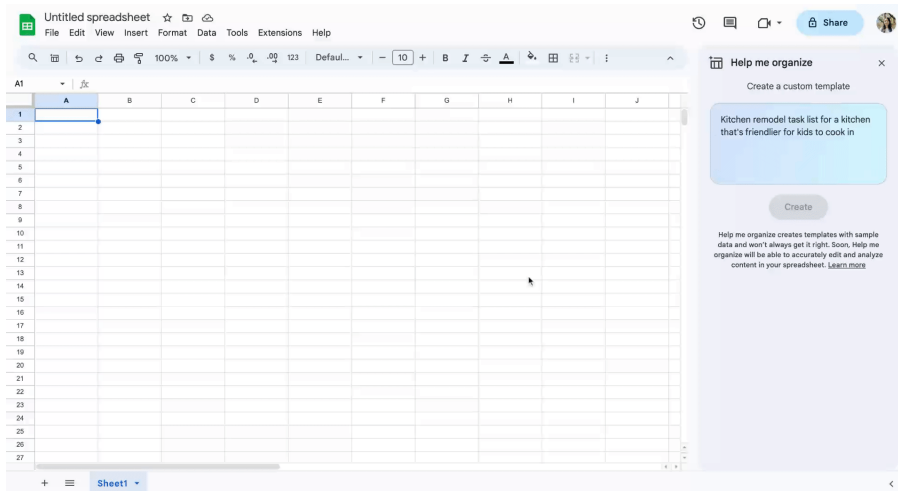
**AI Agents**  
Unleashing the power of multiple AIs: One goal, limitless productivity.

**AI Gift Finder**

**AI News Roundup**

**AI Showcase**

# Office productivity suites with LLMs inside are already here!



## Copilot for Microsoft 365

Microsoft Copilot for Microsoft 365 combines the power of large language models (LLMs) with your organization's data – all in the flow of work – to turn your words into one of the most powerful productivity tools on the planet.


It works alongside popular Microsoft 365 apps such as Word, Excel, PowerPoint, Outlook, Teams, and more. Microsoft 365 Copilot provides real-time intelligent assistance, enabling users to enhance their creativity, productivity, and skills.

<https://workspace.google.com/solutions/ai/>

<https://techcommunity.microsoft.com/t5/microsoft-mechanics-blog/how-microsoft-365-copilot-works/ba-p/3822755>



# There's a startup "gold rush" under way to create products and services

 [About](#) [Companies](#) [Startup Jobs](#) [Find a Co-Founder](#) [Library](#) [SAFE](#) [Resources](#) [Apply for S2024 batch.](#) [Apply](#)

## Generative AI Startups funded by Y Combinator (YC) 2024


February 2024

Browse 100 of the top Generative AI startups funded by Y Combinator.

We also have a Startup Directory where you can [search through over 4,000 companies](#).

Generative AI startups by location


1. **New York**
2. **San Francisco Bay Area**



**Jasper.ai (w2018)** • Active • 195 employees • Austin, TX, USA

Launched in February 2021, Jasper is an AI content platform that helps creators and companies of all types expand their creative potential. Over 105,000 active customers are using Jasper to break through writer's block, repackage their content, create original art, and adjust writing for language and tone. The interest in AI continues to grow and we are at the forefront of teaching the world how to...

[artificial-intelligence](#) | [generative-ai](#) | [ai](#) | [conversational-ai](#)



**Moonvalley (w2021)** • Active • 6 employees • Toronto, ON, Canada

Plan & produce studio-quality video and animation in minutes with cutting-edge deep learning video technology. Moonvalley is led by the former head of product growth at Zapier (and former computer vision researcher), and backed by Khosla Ventures, Y Combinator, Global Founders Capital, FJ Labs, Pioneer Fund, Soma Capital, and a number of other investors. (HQ: Canada)

[artificial-intelligence](#) | [generative-ai](#) | [video](#) | [ai](#)

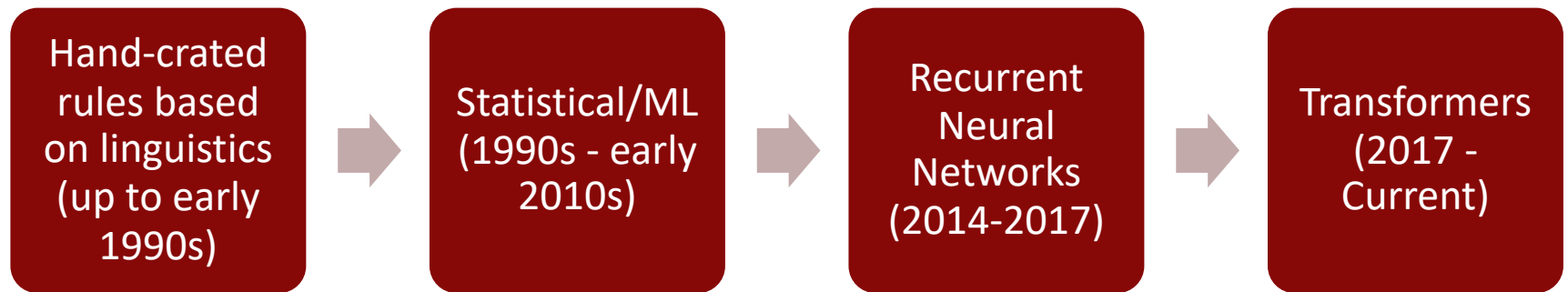
**Humanloop (s2020)** • Active • 10 employees • London, UK

<https://www.ycombinator.com/companies/industry/generative-ai>

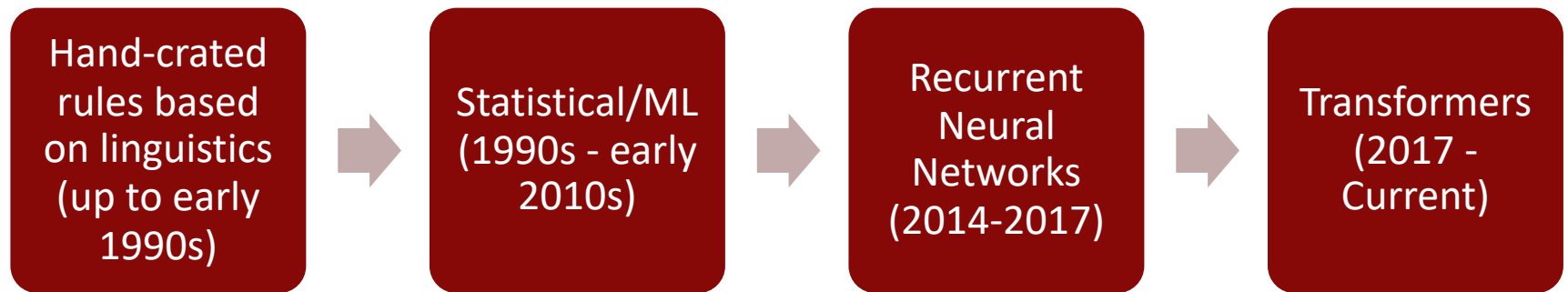
# The Arc of NLP Progress – How did we get here?

# The Arc of NLP Progress

---



# NLP Progress

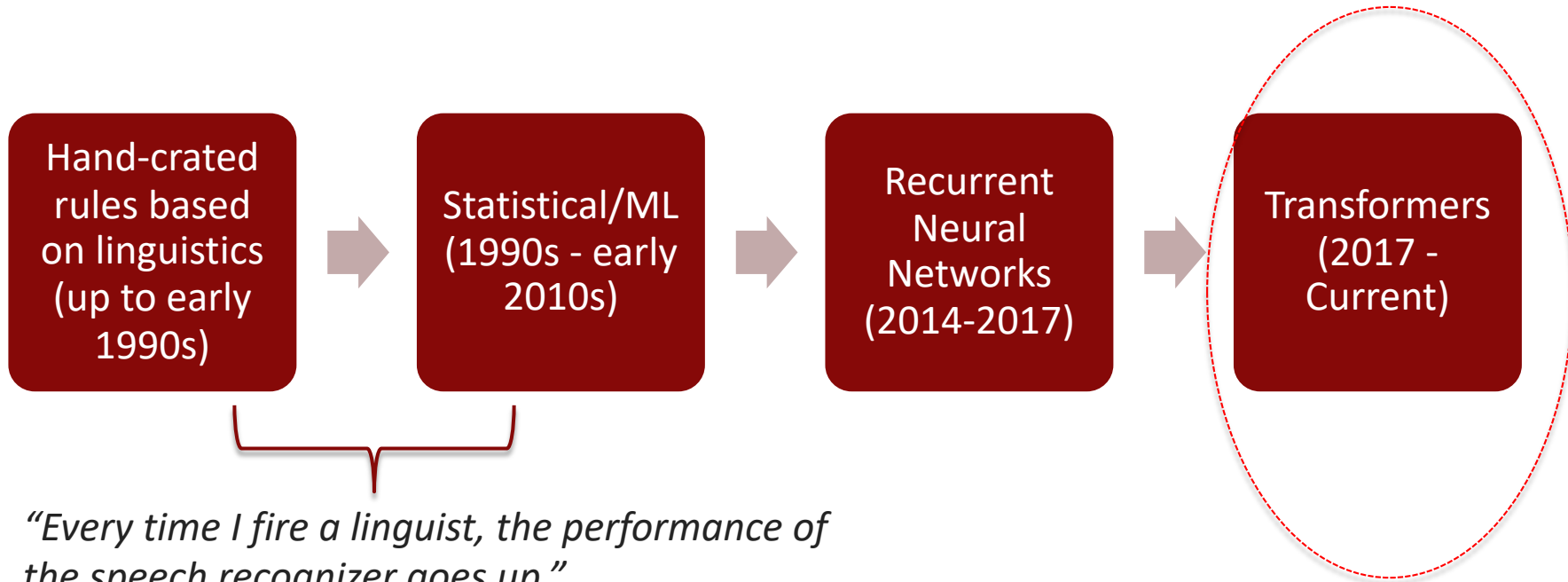


*“Every time I fire a linguist, the performance of the speech recognizer goes up.”*

Frederick Jelinek

---

# NLP Progress



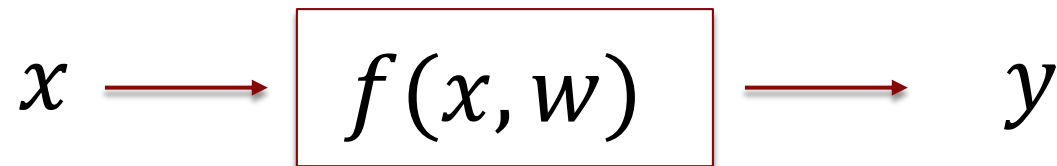
*“Every time I fire a linguist, the performance of the speech recognizer goes up.”*

Frederick Jelinek

*We will leapfrog to this in HODL!*

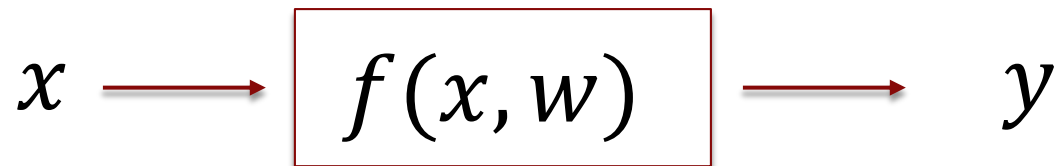
# 20,000 Foot View of the Problem

Like most things, fancy regression!



# 20,000 Foot View of the Problem

Like most things, fancy regression!



$x = \text{text}$

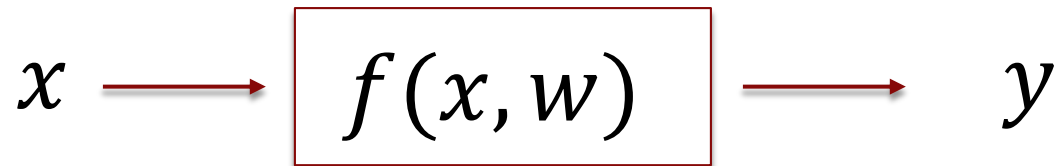
$y = \textbf{text}$ , labels, numbers, ...

$w = \text{weights}$

$f(x, w) = \text{A deep neural network}$

# 20,000 Foot View of the Problem

Like most things, fancy regression!



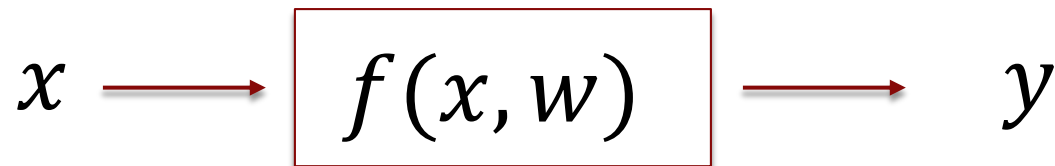
Key questions:

- How to represent  $x$ . We will focus on this today.



# 20,000 Foot View of the Problem

Like most things, fancy regression!

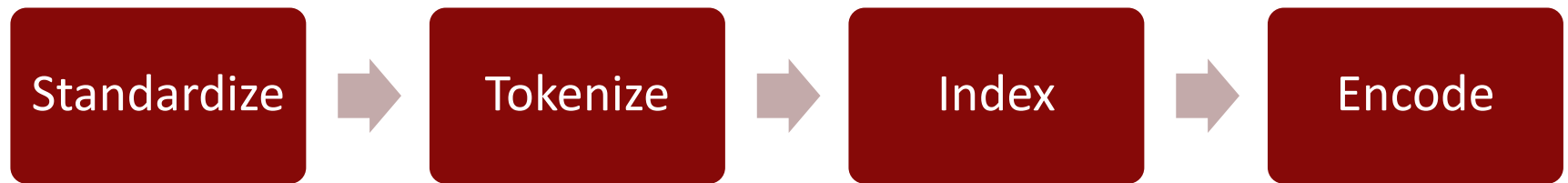


Key questions:

- How to represent  $x$ . We will focus on this today.
- (Next week) What NN architecture is best for processing text?

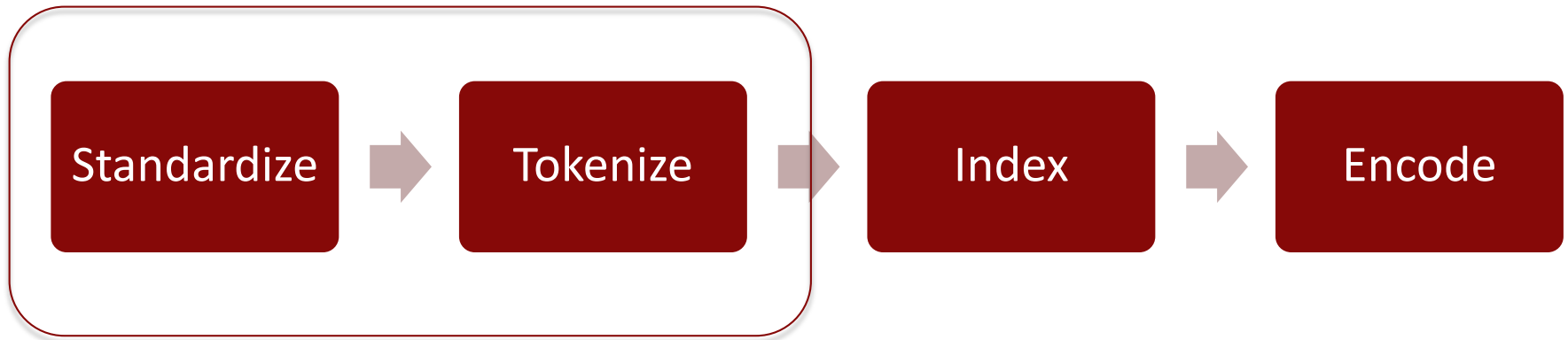
# Processing Basics

# Basic Pre-Processing



This process is called *text vectorization*

# Basic Pre-Processing

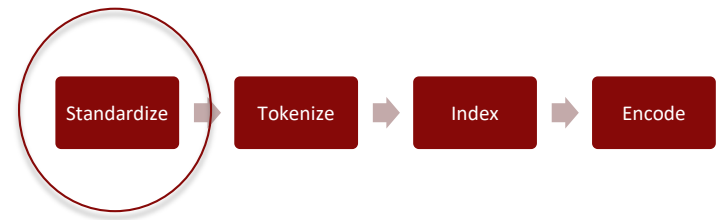


We first do these two steps for every sentence in our training dataset\*

---

\*aka “training corpus”

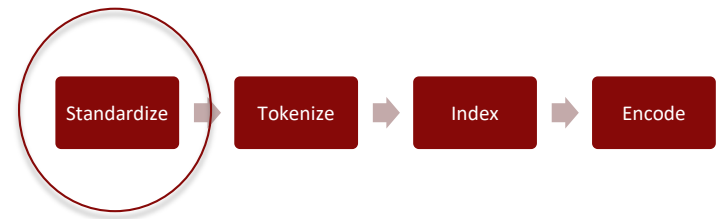
# Basic Pre-Processing



## Standardization

- Strip capitalization, often punctuation and accents (*almost always*)
- Strip 'stop words' e.g., a, the, it, .. (*sometimes*)
- Stemming (e.g., ate, eaten, eating, eaten > [eats]) (*rarely*)

# Basic Pre-Processing



## Standardization

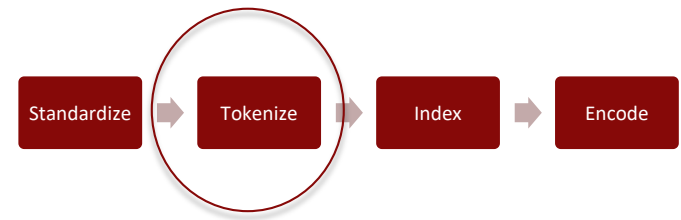
- Strip capitalization, often punctuation and accents (*almost always*)
- Strip 'stop words' e.g., a, the, it, .. (*sometimes*)
- Stemming (e.g., ate, eaten, eating, eaten > [eats]) (*rarely*)

Hola! What do you picture when you think of traveling to Mexico? Sipping a real margarita while soaking up the sun on a laid-back beach in Puerto Vallarta?



hola what do you picture when you [thinks] of [travels] to mexico [sips] real margarita while [soaks] up sun on laidback beach in puerto vallarta

# Basic Pre-Processing



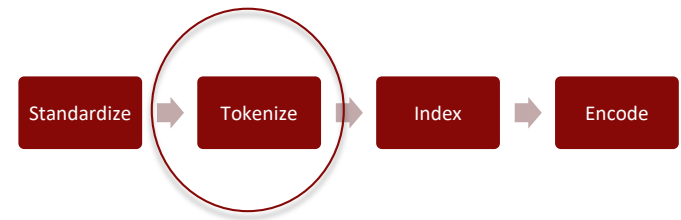
## Tokenization

- *Typically*, each word is a token\* (i.e., split each string on whitespace)
- [design choice] decide how many consecutive words make up a *token*

---

\*Modern LLMs like ChatGPT use a different tokenization scheme (we will briefly discuss in Lecture 9)

# Basic Pre-Processing



## Tokenization

- *Typically*, each word is a token (i.e., split each string on whitespace)
- [design choice] decide how many consecutive words make up a *token*

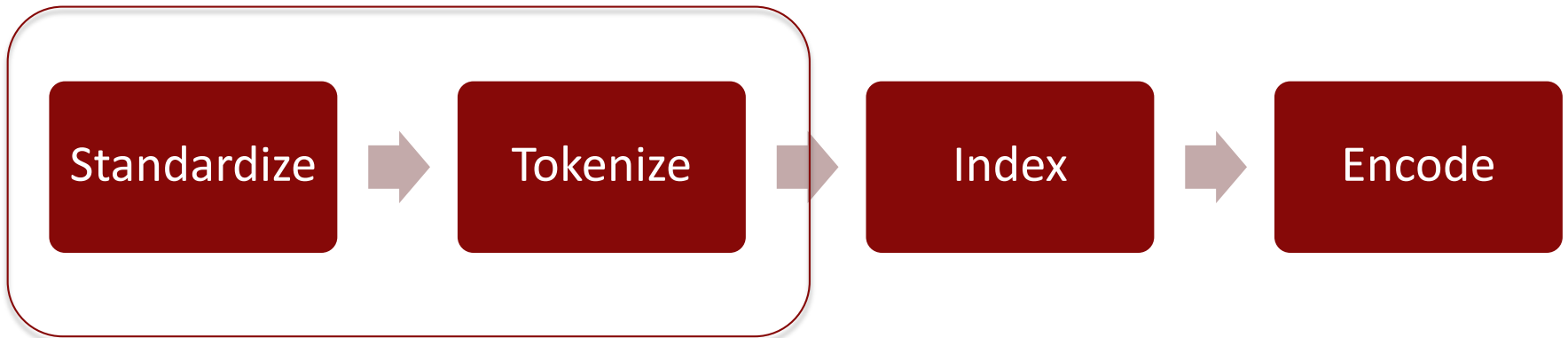
hola what do you picture when you [thinks] of [travels] to mexico [sips]  
real margarita while [soaks] up sun on laidback beach in puerto vallarta



“hola”, “what”, “do”, “you”, “picture”, “when”, “you”, “[thinks]”,  
“of”, “[travels]”, “to”, “mexico”, “[sips]”, “real”, “margarita”, “while”,  
“[soaks]”, “up”, “sun”, “on”, “laidback”, “beach”, “in”, “puerto”, “vallarta”

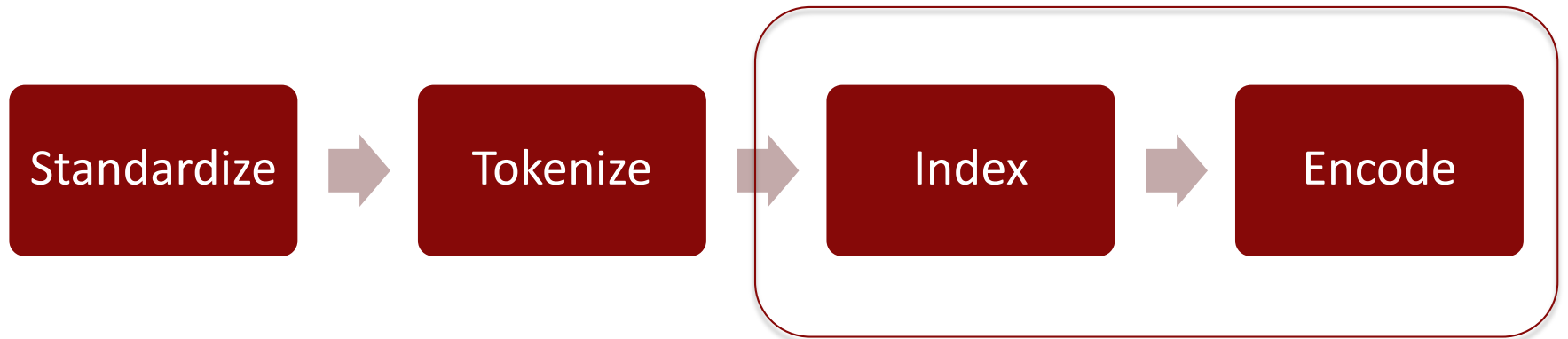


# Basic Pre-Processing



When this is done for every sentence in our training dataset, we have a list of distinct tokens  
= **our vocabulary**

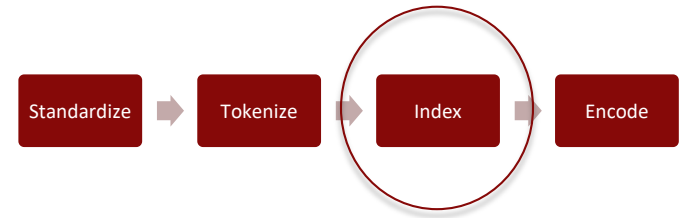
# Basic Pre-Processing



When this is done for every sentence in our training dataset, we have a list of distinct tokens = our vocabulary

Now we move to the third and fourth stages. In these stages, we only work with the vocabulary

# Basic Pre-Processing



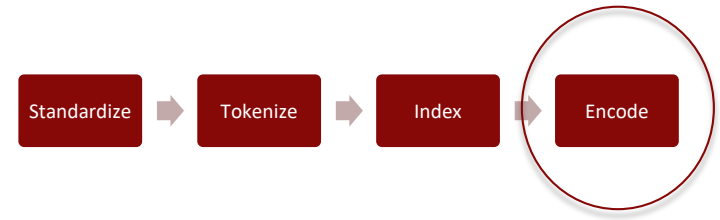
Indexing: We assign a unique integer to each distinct token in the vocabulary

| Token    | Integer |
|----------|---------|
| <UNK>    | 0*      |
| a        | 1       |
| aardvark | 2       |
| ...      |         |
| zebra    | 50000   |

---

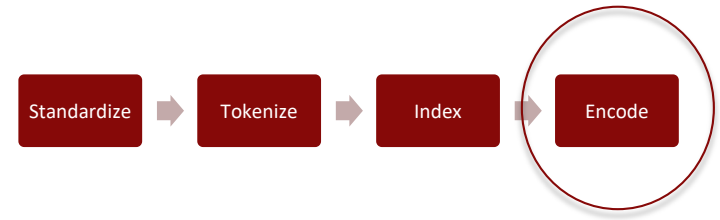
\*we will come back to this special token later

# Basic Pre-Processing



Encoding: We assign a *vector* to each integer in our vocabulary

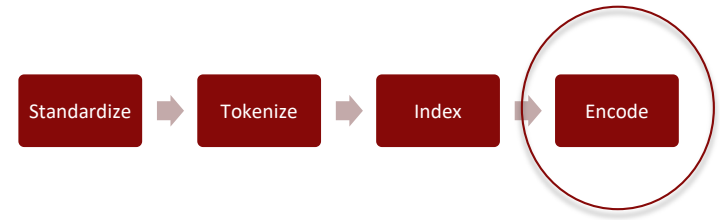
# Basic Pre-Processing



Encoding: We assign a *vector* to each integer in our vocabulary

- The simplest way to do this is \_\_\_\_\_

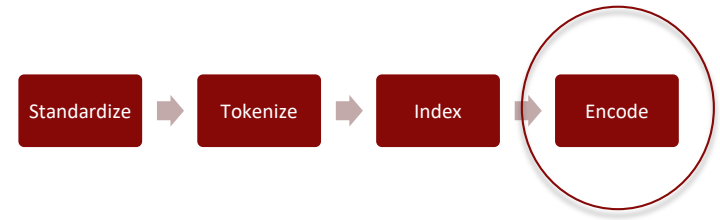
# Basic Pre-Processing



Encoding: We assign a *vector* to each integer in our vocabulary

- The simplest way to do this is **one-hot encoding**

# Basic Pre-Processing

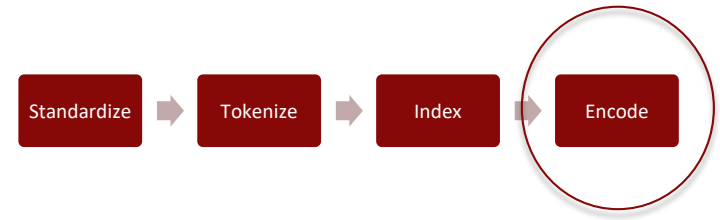


Encoding: We assign a *vector* to each integer in our vocabulary

- The simplest way to do this is one-hot encoding

$$\langle \text{UNK} \rangle \rightarrow \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \mathbf{a} \rightarrow \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

# Basic Pre-Processing



Encoding: We assign a *vector* to each integer in our vocabulary

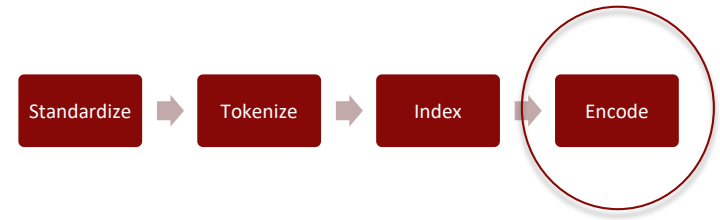
- The simplest way to do this is one-hot encoding

$$\langle \text{UNK} \rangle \rightarrow \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \mathbf{a} \rightarrow \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

- Dimension of encoding vector = max no. of distinct tokens in the text



# Basic Pre-Processing



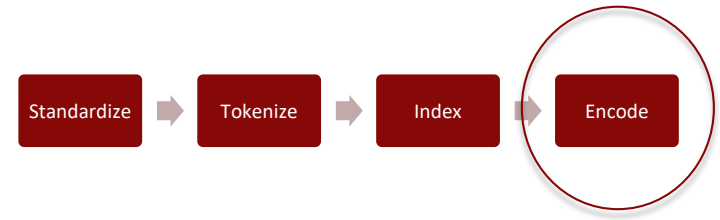
Encoding: We assign a *vector* to each integer in our vocabulary

- The simplest way to do this is one-hot encoding

$$\langle \text{UNK} \rangle \rightarrow \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \mathbf{a} \rightarrow \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

- Dimension of encoding vector = max no. of distinct tokens in the text + one for  $\langle \text{UNK} \rangle$

# Basic Pre-Processing



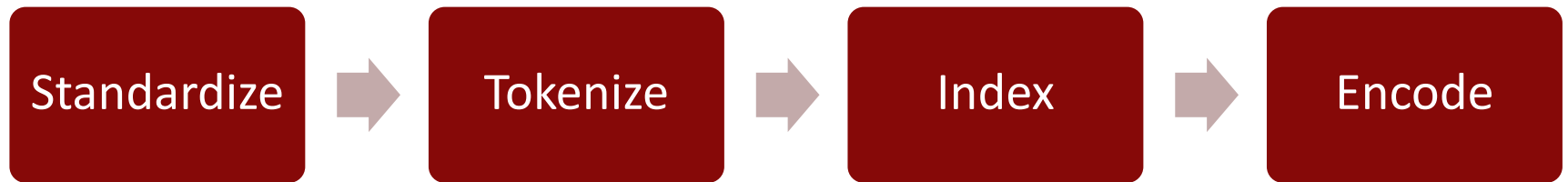
Encoding: We assign a *vector* to each integer in our vocabulary

- The simplest way to do this is one-hot encoding

$$\langle \text{UNK} \rangle \rightarrow \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \mathbf{a} \rightarrow \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

- Dimension of encoding vector = max no. of distinct tokens in the text + one for  $\langle \text{UNK} \rangle$
- This is called the “vocabulary” size

# Basic Pre-Processing



At this point,

- we have created a vocabulary from the training corpus and
- every distinct token in our vocabulary has been assigned a one-hot vector.

We are done with basic preprocessing.

# Next: How to get a *new* input sentence\* ready to be “fed” into a DNN

- Let's say we have completed STIE on the training corpus and our vocabulary size is 100.

VOCABULARY

← 100 →

UNK a ... an ... car ... cat ... hat ..... mat .... ok ... on .... sat ..... ten ... the ... zebra

# Next: How to get a *new* input sentence ready to be “fed” into a DNN

- Let's say we have completed STIE on the training corpus and our vocabulary size is 100.

VOCABULARY

← 100 →

UNK a ... an ... car ... cat ... hat ..... mat ... ok ... on ... sat ..... ten ... the ... zebra

- This input text string arrives - “The cat sat on the mat” – and we run it through STIE



# Next: How to get a *new* input sentence ready to be “fed” into a DNN

- Let's say we have completed STIE on the training corpus and our vocabulary size is 100.

VOCABULARY

← 100 →

UNK a ... an ... car ... cat ... hat ... mat ... ok ... on ... sat ... ten ... the ... zebra

- This input text string arrives - “The cat sat on the mat” – and we run it through STIE



- The output is a table with A rows and B columns. What are A and B?

# Next: How to get a *new* input sentence ready to be “fed” into a DNN

- Let's say we have completed STIE on the training corpus and our vocabulary size is 100.

VOCABULARY

← 100 →

UNK a ... an ... car ... cat ... hat ..... mat .... ok ... on .... sat ..... ten ... the ... zebra

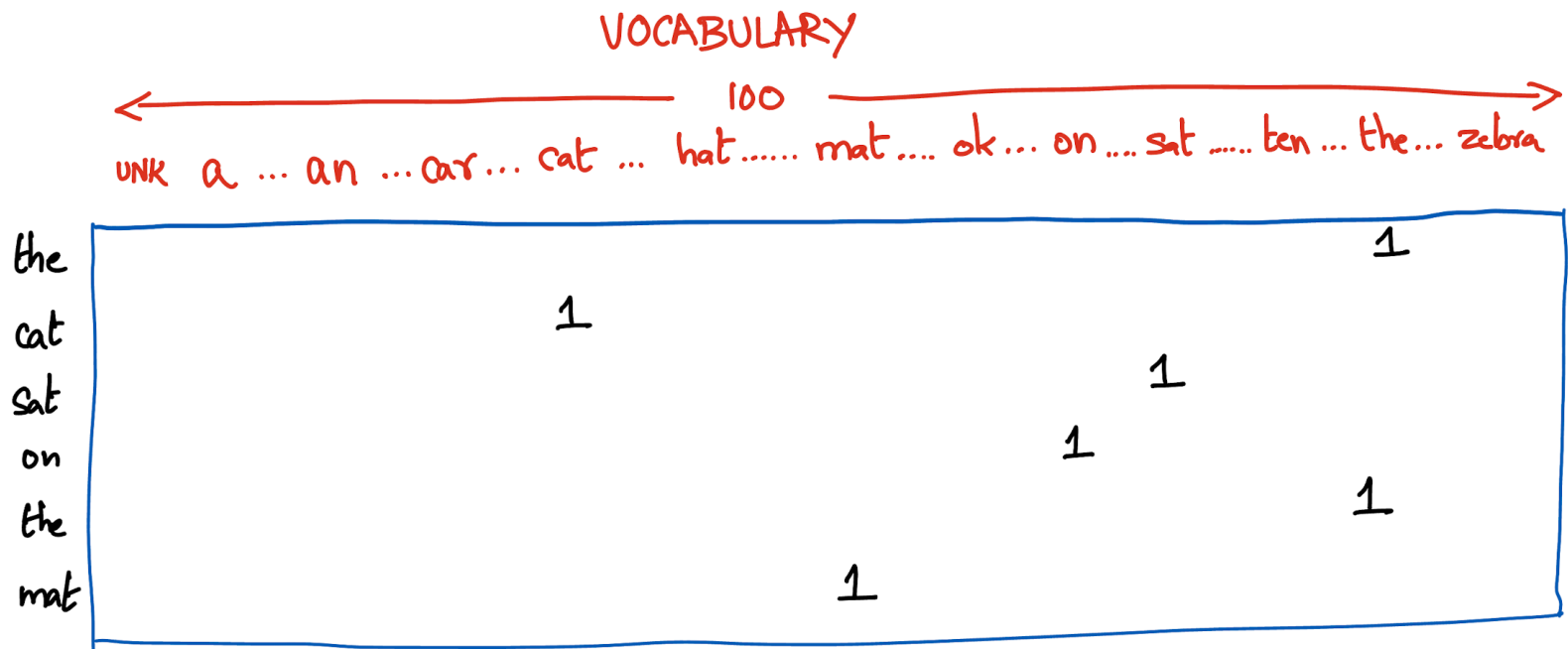
- This input text string arrives - “The cat sat on the mat” – and we run it through STIE



- The output is a 6 x 100 table.

# How to get a *new* input sentence ready to be “fed” into a DNN

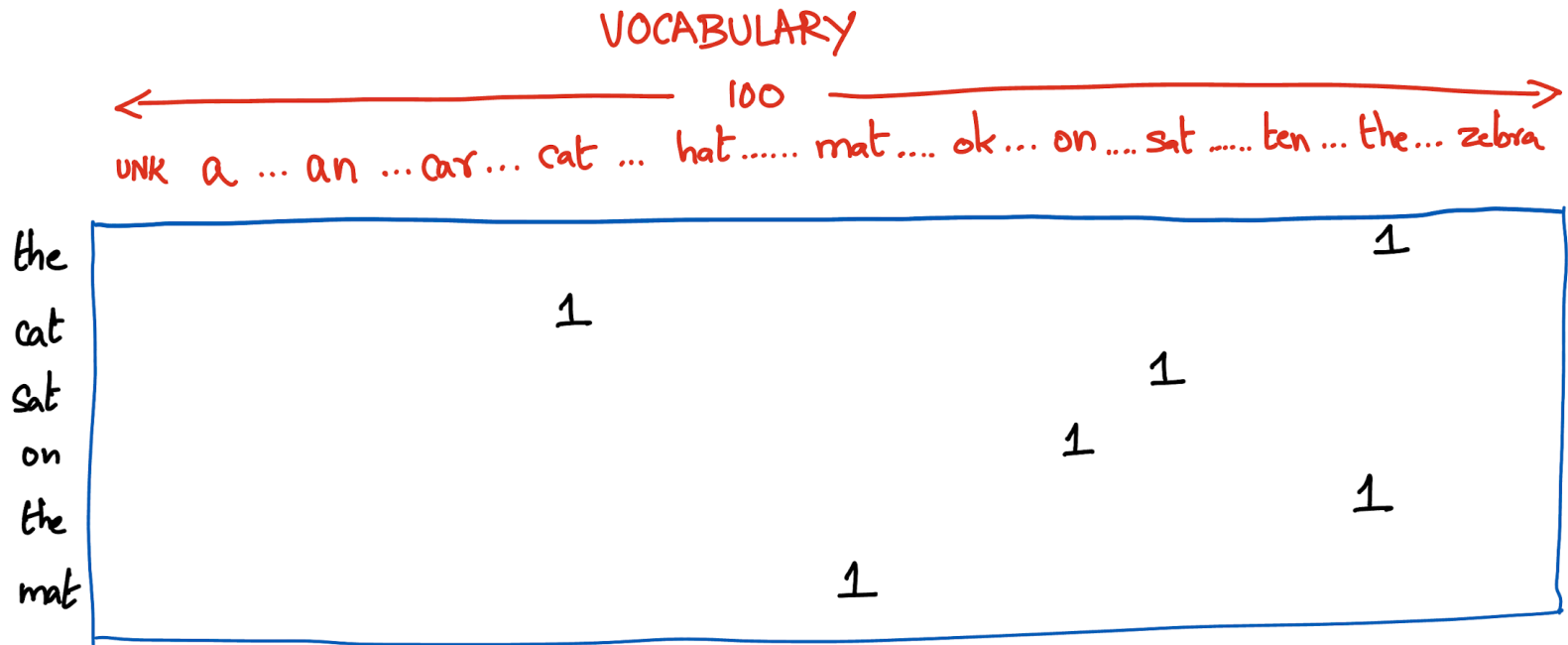
## The output table





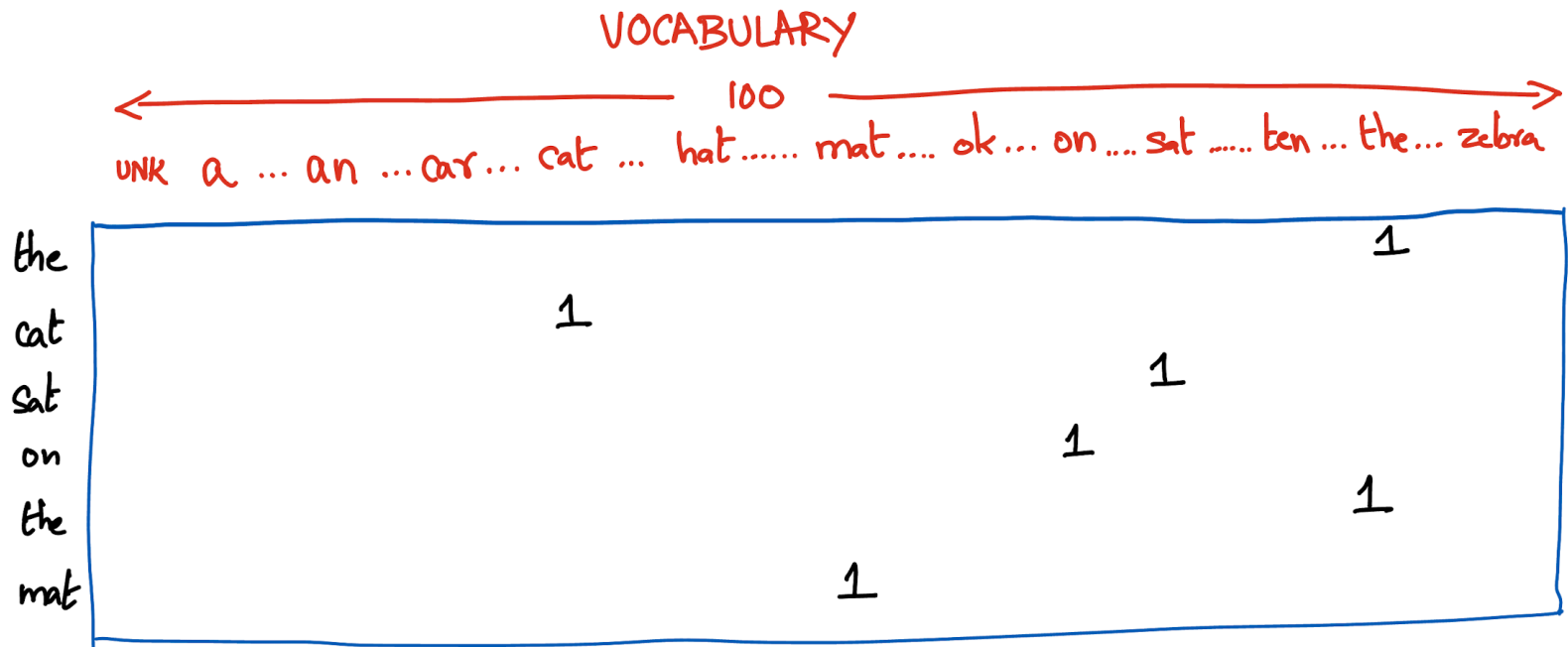
# How to get a *new* input sentence ready to be “fed” into a DNN

- What's the best way to “feed” this 6 x 100 table of numbers to a DNN?



# How to get a *new* input sentence ready to be “fed” into a DNN

- What's the best way to “feed” this 6 x 100 table of numbers to a DNN?
- Can we send this table as-is into a DNN?



# How to get a *new* input sentence ready to be “fed” into a DNN

- What’s the best way to “feed” this 8 x 100 table of numbers to a DNN?
- Can we send this table as-is into a DNN?
- A complication: Each incoming sentence may have a different number of words i.e.. may have *varying length*. It will be nice to have a fixed-length input

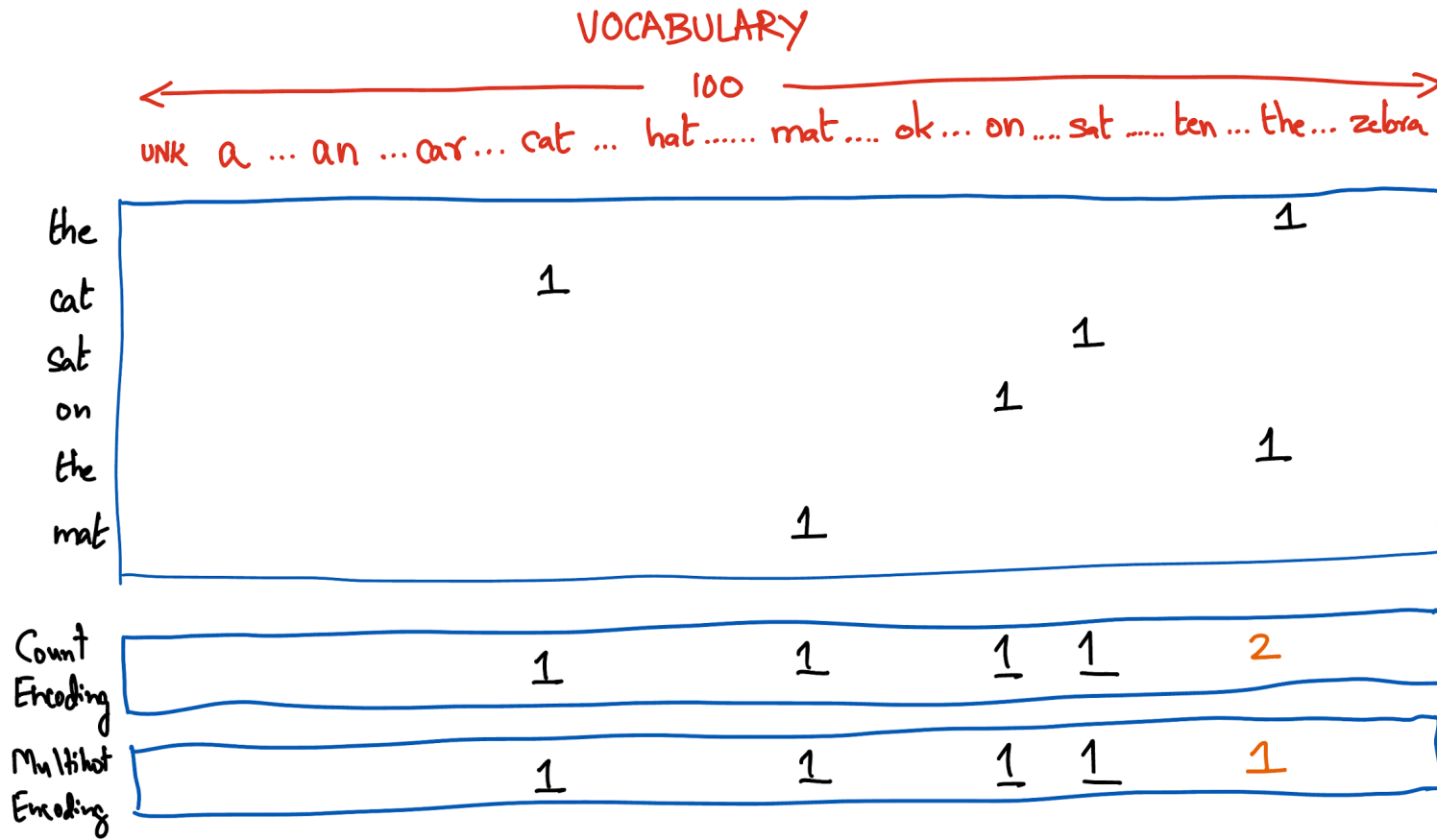
# How to get a *new* input sentence ready to be “fed” into a DNN

- What’s the best way to “feed” this 8 x 100 table of numbers to a DNN?
- Can we send this table as-is into a DNN?
- A complication: Each incoming sentence may have a different number of words i.e.. may have *varying length*. It will be nice to have a fixed-length input
- How about we “aggregate” the vectors?

# How to get a *new* input sentence ready to be “fed” into a DNN

- What’s the best way to “feed” this 8 x 100 table of numbers to a DNN?
- Can we send this table as-is into a DNN?
- A complication: Each incoming sentence may have a different number of words i.e.. may have *varying length*. It will be nice to have a fixed-length input
- How about we “aggregate” the vectors?
  - Sum the vectors. This is called “**count encoding**”
  - “OR” the vectors. This is called “**multi-hot encoding**”

## Example: Count and Multi-hot Encoding



# How to get a *new* input sentence ready to be “fed” into a DNN

- What’s the best way to “feed” this 8 x 100 table of numbers to a DNN?
- Can we send this table as-is into a DNN?
- A complication: Each incoming sentence may have a different number of words i.e.. may have *varying length*. It will be nice to have a fixed-length input
- How about we “aggregate” the vectors?
  - Sum the vectors. This is called “count encoding”
  - “OR” the vectors. This is called “multi-hot encoding”
- This aggregation approach is called the Bag of Words model

# Does the Bag of Words approach have any shortcomings?

- We lose the meaning inherent in the *order* of the words (i.e., we lose “sequentiality”)
- If the vocabulary is very long, each input – regardless of its number of tokens – will be a vector that’s as long as the size of the vocabulary.
  - This can be somewhat mitigated by choosing only the most-frequent words
  - Nevertheless, this increases the number of weights the model has to learn and thus also the compute time and the risk of overfitting.



# Task For NLP 1

# Super Serious Application

I grew up on the crime side, the New York Times side  
Stayin' alive was no jive  
Had secondhands, Mom's bounced on old man  
So then we moved to Shaolin land

---

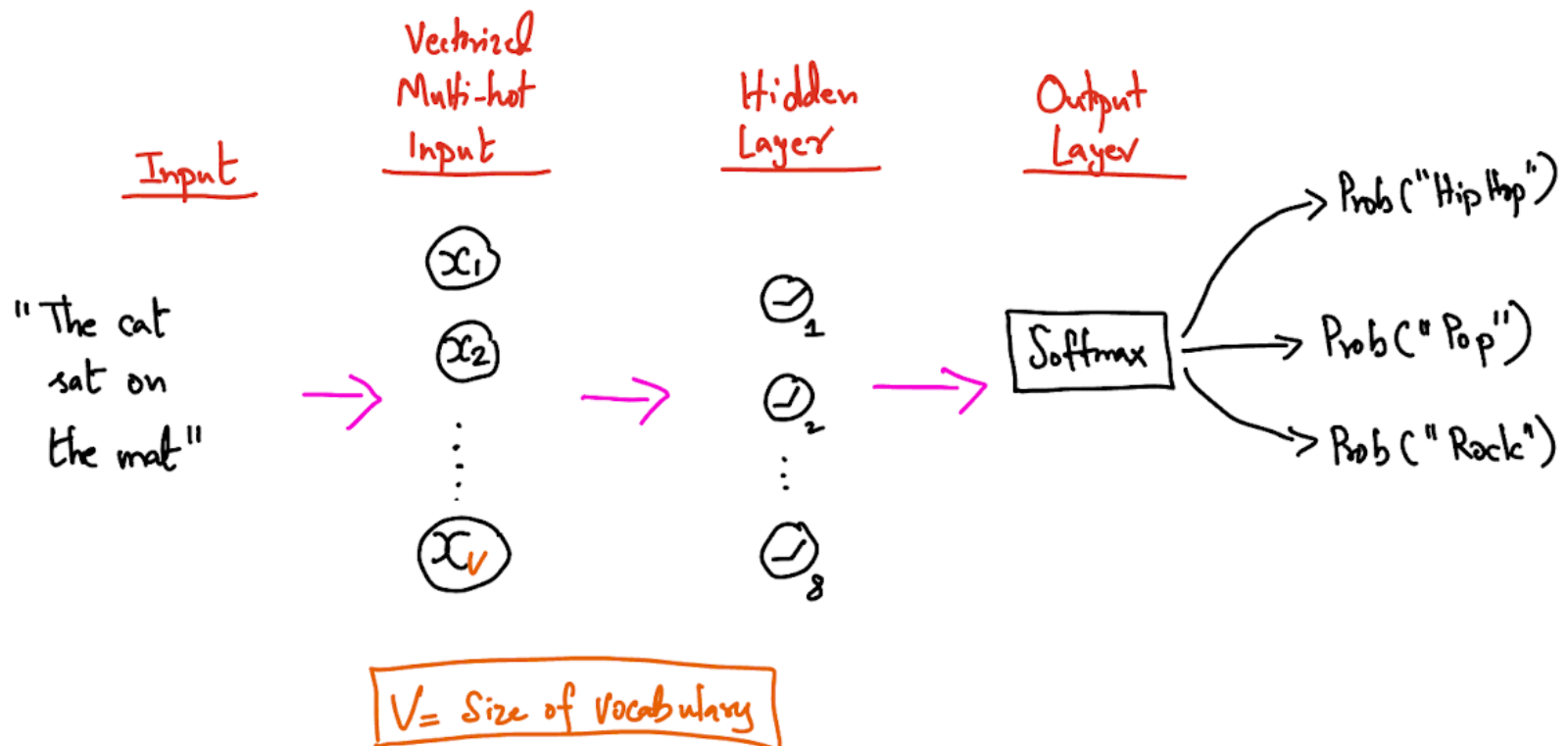
I walked through the door with you  
The air was cold  
But something about it felt like home somehow  
And I, left my scarf there at your sisters house

Can you classify each verse above into *hip-hop*, *rock* or *pop*?

# What's the simplest NN-based classifier we can build?



# What's the simplest NN-based classifier we can build?



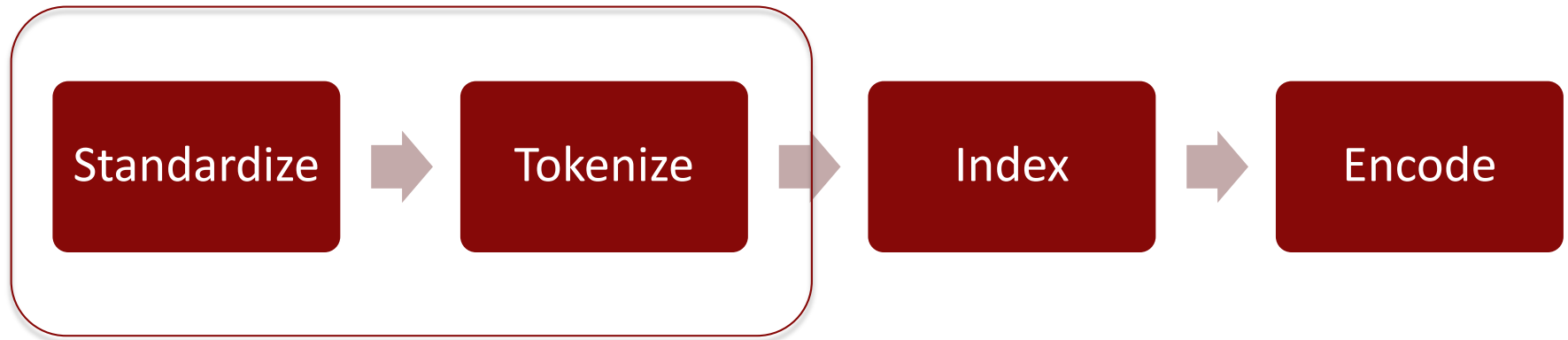
# Colab

(text pre-processing, bag-of-words and bigrams)

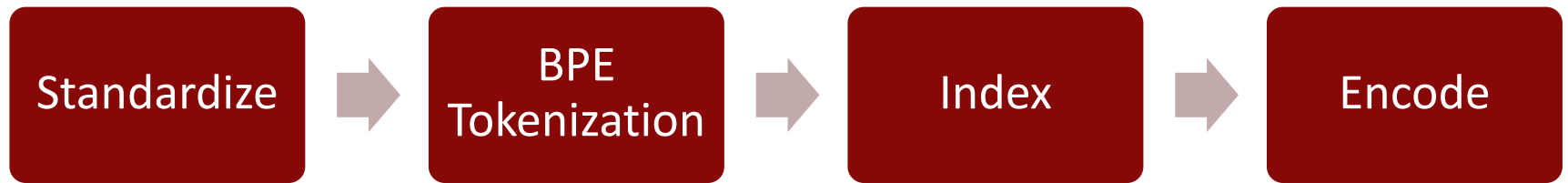
<https://colab.research.google.com/drive/1jzL9IWqLbMyTZocClXSuivFz3CrNHM3h?usp=sharing>

# Byte Pair Encoding

# Any **disadvantages** to what we described earlier?



# Byte Pair Encoding (BPE)



Modern generative models use tokenization schemes that try to address these disadvantages.

The GPT family uses **Byte Pair Encoding** (BPE)



# How BPE works

Key Intuition: **Start with each character as a token**, and “merge” tokens that most frequently occur next to each other. Stop when the size of your vocabulary reaches a user-defined limit (we will assume 12 in the example below\*)

|                              |                                     |
|------------------------------|-------------------------------------|
| <i>Training corpus</i>       | The cat sat on the mat              |
| <i>After Standardization</i> | the cat sat on the mat              |
| <i>Starting vocabulary</i>   | [t] [h] [e] [c] [a] [s] [o] [n] [m] |

---

\*The GPT family uses a vocab size of 50,000

# How BPE works

- Start with the vocabulary [t, h, e, c, a, s, o, n, m]
- [t,h,e] [c,a,t] [s,a,t] [o,n] [t,h,e] [m,a,t]
- What is the frequency of pairs?
  - [t,h] – 2
  - [h,e] – 2
  - [c,a] – 1
  - [a,t] – 3
  - [s,a] – 1
  - [o,n] – 1
  - [m,a] – 1
- So [a,t] is the most frequent (at 3) and we add 'at' to the vocabulary which becomes [t, h, e, c, a, s, o, n, m, **at**]

# How BPE works

- [t,h,e] [c,a,t] [s,a,t] [o,n] [t,h,e] [m,a,t]
- [t,h,e] [c,at] [s,at] [o,n] [t,h,e] [m,at] (after merging [a,t])
- What is the frequency of pairs?
  - [t,h] – 2
  - [h,e] – 2
  - [c,at] – 1
  - [s,at] – 1
  - [o,n] – 1
  - [m,at] – 1
- So [t,h] is the most frequent (at 2) and we add 'th' to the vocabulary which becomes [t, h, e, c, a, s, o, n, m, at, th]

# How BPE works

- [t,h,e] [c,at] [s,at] [o,n] [t,h,e] [m,at]
- [th,e] [c,at] [s,at] [o,n] [th,e] [m,at] (after merging [t,h])
- What is the frequency of pairs?
  - [th,e] – 2
  - [c,at] – 1
  - [s,at] – 1
  - [o,n] – 1
  - [m,at] – 1
- So [th,e] is the most frequent (at 2) and we add 'the' to the vocabulary which becomes ...
- [t, h, e, c, a, s, o, n, m, at, th, the]

# How BPE works

Key Intuition: Start with each character as a token, and “merge” tokens that **most frequently occur next to each other**. Stop when the size of your vocabulary reaches a user-defined limit (we will assume 12 in the example below)

|  |   |
|--|---|
| <i>Starting vocabulary</i>                   | [t] [h] [e] [c] [a] [s] [o] [n] [m]   |
| <i>Starting corpus</i>                       | [t,h,e] [c,a,t] [s,a,t] [o,n] [t,h,e] [m,a,t]   |
| <i>Frequency of adj tokens</i>               | [t,h] – 2   [h,e] – 2   [c,a] – 1 <b>[a,t] – 3</b> [s,a] – 1   [o,n] – 1<br>[m,a] – 1 |
| <i>Vocabulary after 1<sup>st</sup> merge</i> | [t] [h] [e] [c] [a] [s] [o] [n] [m] <b>[at]</b>                                       |
| <i>Corpus after 1<sup>st</sup> merge</i>     | [t,h,e] [c, <b>at</b> ] [s, <b>at</b> ] [o,n] [t,h,e] [m, <b>at</b> ]                 |
| <i>Frequency of adj tokens</i>               | <b>[t,h]</b> – 2   [h,e] – 2   [c,at] – 1   [s,at] – 1   [o,n] – 1   [m,at] – 1       |
| <i>Vocabulary after 2<sup>nd</sup> merge</i> | [t] [h] [e] [c] [a] [s] [o] [n] [m] [at] <b>[th]</b>                                  |
| <i>Corpus after 2<sup>nd</sup> merge</i>     | <b>[th,e]</b> [c,at] [s,at] [o,n] <b>[th,e]</b> [m,at]                                |
| <i>Frequency of adj tokens</i>               | <b>[th,e]</b> – 2   [c,at] – 1   [s,at] – 1   [o,n] – 1   [m,at] – 1                  |
| <i>Vocabulary after 3<sup>rd</sup> merge</i> | [t] [h] [e] [c] [a] [s] [o] [n] [m] [at] <b>[th][the]</b>                             |

# How BPE works

Key Intuition: Start with each character as a token, and “merge” tokens that most frequently occur next to each other. **Stop when the size of your vocabulary reaches a user-defined limit (we will assume 12 in the example below)**

|  |   |
|--|---|
| <i>Starting vocabulary</i>                   | [t] [h] [e] [c] [a] [s] [o] [n] [m]   |
| <i>Starting corpus</i>                       | [t,h,e] [c,a,t] [s,a,t] [o,n] [t,h,e] [m,a,t]   |
| <i>Frequency of adj tokens</i>               | [t,h] – 2   [h,e] – 2   [c,a] – 1 <b>[a,t] – 3</b> [s,a] – 1   [o,n] – 1<br>[m,a] – 1 |
| <i>Vocabulary after 1<sup>st</sup> merge</i> | [t] [h] [e] [c] [a] [s] [o] [n] [m] <b>[at]</b>                                       |
| <i>Corpus after 1<sup>st</sup> merge</i>     | [t,h,e] [c, <b>at</b> ] [s, <b>at</b> ] [o,n] [t,h,e] [m, <b>at</b> ]                 |
| <i>Frequency of adj tokens</i>               | <b>[t,h]</b> – 2   [h,e] – 2   [c,at] – 1   [s,at] – 1   [o,n] – 1   [m,at] – 1       |
| <i>Vocabulary after 2<sup>nd</sup> merge</i> | [t] [h] [e] [c] [a] [s] [o] [n] [m] [at] <b>[th]</b>                                  |
| <i>Corpus after 2<sup>nd</sup> merge</i>     | <b>[th,e]</b> [c,at] [s,at] [o,n] <b>[th,e]</b> [m,at]                                |
| <i>Frequency of adj tokens</i>               | <b>[th,e]</b> – 2   [c,at] – 1   [s,at] – 1   [o,n] – 1   [m,at] – 1                  |
| <i>Vocabulary after 3<sup>rd</sup> merge</i> | [t] [h] [e] [c] [a] [s] [o] [n] [m] [at] <b>[th][the]</b>                             |

**We are done since the vocab size is now 12**

# How BPE works

The merges happened in this order:

- a,t => at
- t,h => th
- th,e => the

When a new piece of text arrives, the BPE tokenization will apply the merges in the same order.

Example: [t,h,e,\_,r,a,t]

- [t,h,e,\_,r,**at**]
- [**th**,e,\_,r,**at**]
- [**the**,\_,r,**at**]

We are done since the vocab size is now 12