SMALL LANGUAGE MODELS

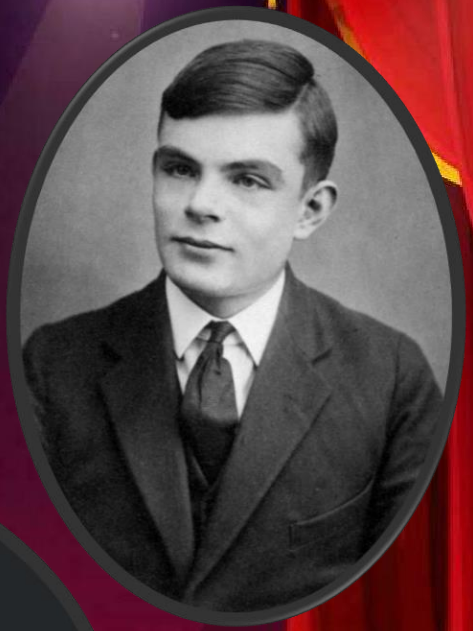
Lecture 7

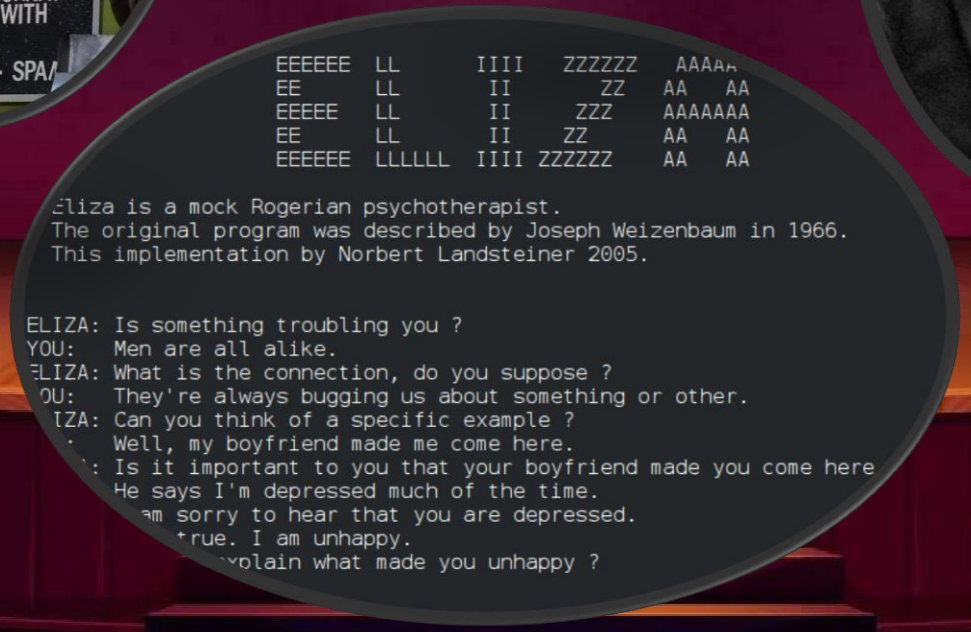
MAL2, Spring 2025

SMALL LANGUAGE MODELS

- Setting the stage
- Char-RNNs
- Implementing Char-RNNs
- Text classification
- Generating music


MENU
EGG + SPAM
EGG BACON + SPAM
EGG BACON SAUSAGE + SPAM
SPAM BACON SAUSAGE + SPAM
SPAM EGG SPAM SPAM BACON + SPAM
SPAM SPAM SPAM EGG + SPAM
SPAM SPAM SPAM SPAM SPAM SPAM BAKED BEANS SPAM SPAM SPAM + SPAM
LOBSTER THERMIDOR AUX CREVETTES WITH MORNAY SAUCE GARNISHED WITH TRUFFLE PATE




EEEEEE LL IIII ZZZZZZ AAAAA
EE LL II ZZ AA AA
EEEEE LL II ZZZ AAAAAAA
EE LL II ZZ AA AA
EEEEEE LLLLLL IIII ZZZZZZ AA AA
Eliza is a mock Rogerian psychotherapist.
The original program was described by Joseph Weizenbaum in 1966.
This implementation by Norbert Landsteiner 2005.
ELIZA: Is something troubling you ?
YOU: Men are all alike.
ELIZA: What is the connection, do you suppose ?
YOU: They're always bugging us about something or other.
IZA: Can you think of a specific example ?
Well, my boyfriend made me come here.
: Is it important to you that your boyfriend made you come here
He says I'm depressed much of the time.
am sorry to hear that you are depressed.
true. I am unhappy.
xplain what made you unhappy ?

Can we build a machine that can master written and spoken language?

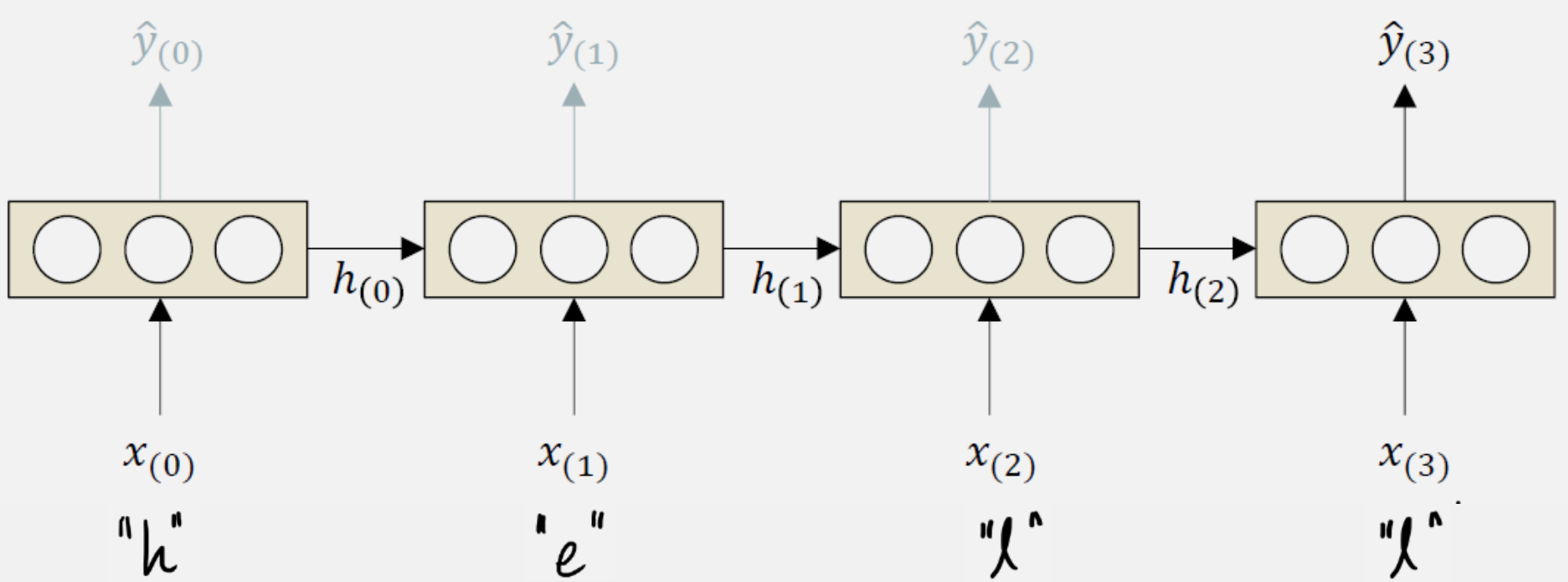
SMALL LANGUAGE MODELS

- Setting the stage
- Char-RNNs
- Implementing Char-RNNs
- Text classification
- Generating music

A char-RNN is a **standard classifier** where each character is a class

CHAR-RNNs

here:
seq2vec

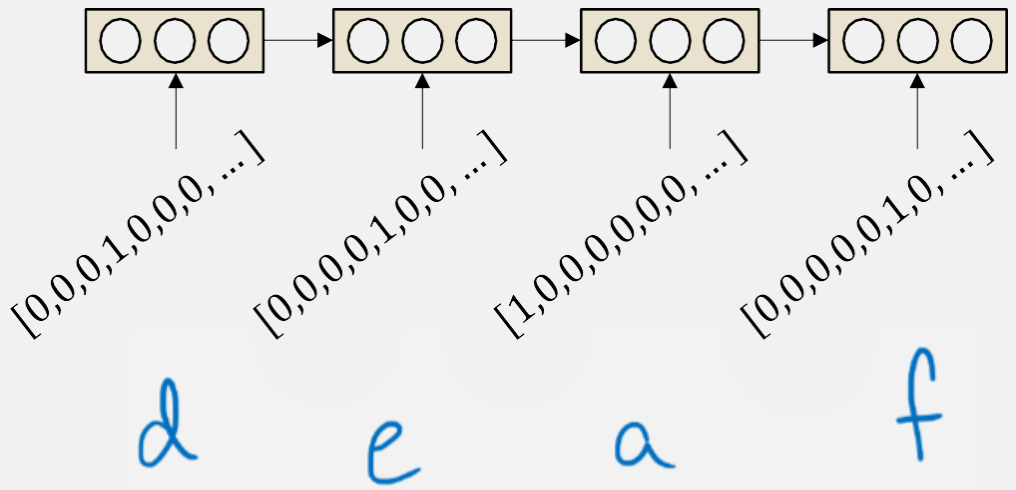


model.predict("hell")
> "o"

HOW DO WE REPRESENT LETTERS?

IDEA #1: One-hot encoding

"a" → [1,0,0,0,0,0,0, …]
"b" → [0,1,0,0,0,0,0, …]
"c" → [0,0,1,0,0,0,0, …]
"d" → [0,0,0,1,0,0,0, …]
"e" → [0,0,0,0,1,0,0, …]
"f" → [0,0,0,0,0,1,0, …]



This can work okay, but we can do something smarter!

HOW DO WE REPRESENT LETTERS?

IDEA #2: Embeddings

An embedding layer is a trainable layer that converts class labels into vectors

→since it's trained the vectors will hopefully meaningful

"Embedding space arithmetic"

king - man + woman ≈ queen

copenhagen - denmark + germany ≈ berlin

- A **dense vector** is like giving each city its **GPS coordinates**:
  - "Paris" = [48.8566, 2.3522]
  - "Berlin" = [52.5200, 13.4050]
  - "Rome" = [41.9028, 12.4964]

WRITING LIKE TOLKIEN I: CREATING A DATASET

```
X = [                                   y = [
    ”In a hole in the gro”                  ”u”
]                                       ]
```

”In a hole in the ground there lived a hobbit.”

WRITING LIKE TOLKIEN I: CREATING A DATASET

```
X = [                              y = [
    ”In a hole in the gro”,            ”u”,
    ”hole in the ground t”             ”h”

]                                  ]
```

”In a hole in the ground there lived a hobbit.”

WRITING LIKE TOLKIEN I: CREATING A DATASET

```
X = [                                   y = [
    ”In a hole in the gro”,                 ”u”,
    ”hole in the ground t”,                 ”h”,
    ”in the ground there ”                  ”l”


]                                       ]
```

”In a hole in the ground there lived a hobbit.”

WRITING LIKE TOLKIEN I: CREATING A DATASET

```
X = [                                   y = [
    ”In a hole in the gro”,                 ”u”,
    ”hole in the ground t”,                 ”h”,
    ”in the ground there ”,                 ”l”,
    ”e ground there lived”                  ” ”,

]                                       ]
```

”In a hole in the ground there lived a hobbit.”

WRITING LIKE TOLKIEN I: CREATING A DATASET

```
X = [                                y = [
    ”In a hole in the gro”,              ”u”,
    ”hole in the ground t”,              ”h”,
    ”in the ground there ”,              ”l”,
    ”e ground there lived”,              ” ”,
    ”und there lived a ho”               ”b”
]                                    ]
```

”In a hole in the ground there lived a hobbit.”

WRITING LIKE TOLKIEN II: TRANSFORM LETTERS TO CLASS LABELS

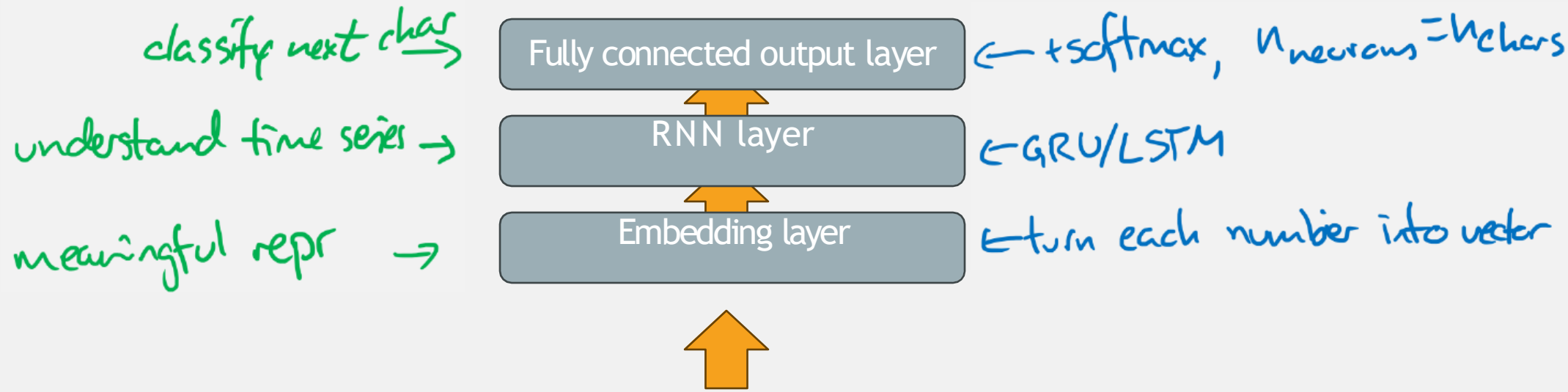
```
X = [[9, 14, 0, 1, 0, 8, 15, 12, 5, 0, 9, 14, 0, 20, 8, 5, 0, 7, 18, 15],
     [8, 15, 12, 5, 0, 9, 14, 0, 20, 8, 5, 0, 7, 18, 15, 21, 14, 4, 0, 20],
     [9, 14, 0, 20, 8, 5, 0, 7, 18, 15, 21, 14, 4, 0, 20, 8, 5, 18, 5, 0],
     [5, 0, 7, 18, 15, 21, 14, 4, 0, 20, 8, 5, 18, 5, 0, 12, 9, 22, 5, 4],
     [21, 14, 4, 0, 20, 8, 5, 18, 5, 0, 12, 9, 22, 5, 4, 0, 1, 0, 8, 15]]
```

```
y = [21,
     8,
     12,
     0,
     2]
```

```
X = [
    ”In a hole in the gro”,
    ”hole in the ground t”,
    ”in the ground there ”,
    ”e ground there lived”,
    ”und there lived a ho”
]
```

```
y = [
    ”u”,
    ”h”,
    ”l”,
    ” ”,
    ”b”
]
```

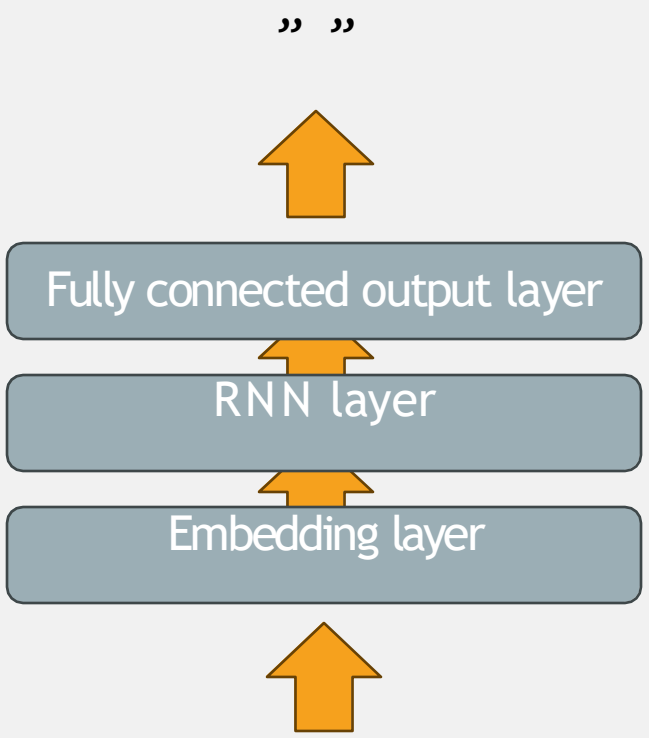
WRITING LIKE TOLKIEN III: CHAR-RNN



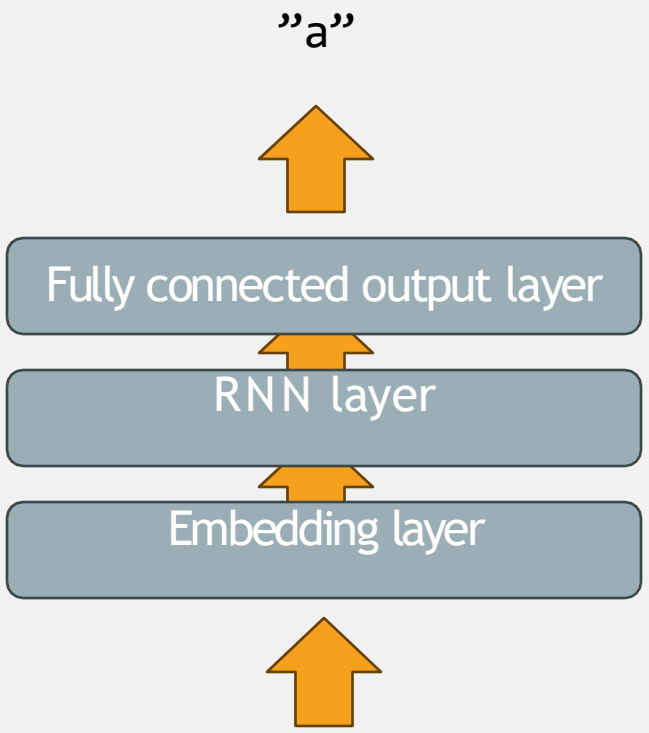

```
X = [[9, 14, 0, 1, 0, 8, 15, 12, 5, 0, 9, 14, 0, 20, 8, 5, 0, 7, 18, 15],
     [8, 15, 12, 5, 0, 9, 14, 0, 20, 8, 5, 0, 7, 18, 15, 21, 14, 4, 0, 20],
     [9, 14, 0, 20, 8, 5, 0, 7, 18, 15, 21, 14, 4, 0, 20, 8, 5, 18, 5, 0],
     [5, 0, 7, 18, 15, 21, 14, 4, 0, 20, 8, 5, 18, 5, 0, 12, 9, 22, 5, 4],
     [21, 14, 4, 0, 20, 8, 5, 18, 5, 0, 12, 9, 22, 5, 4, 0, 1, 0, 8, 15]]

y = [21,
     8,
     12,
     0,
     2]
```

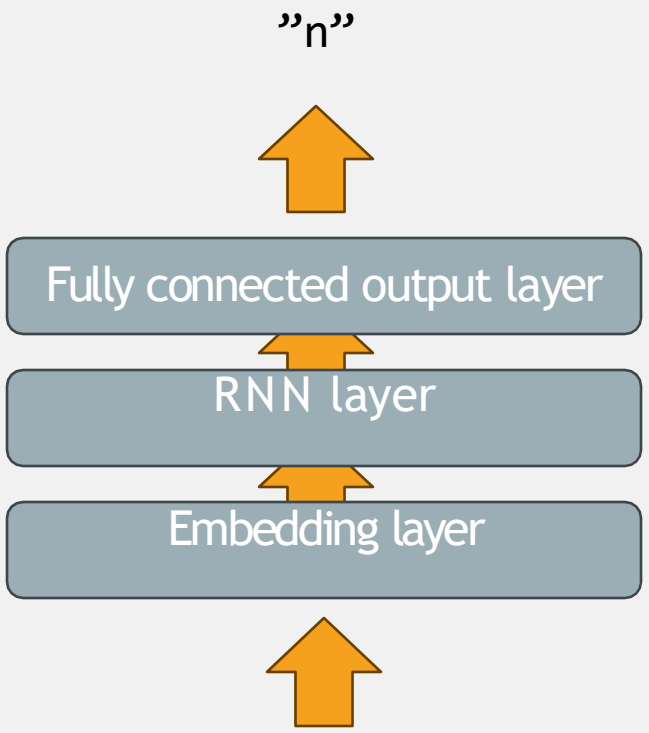
WRITING LIKE TOLKIEN IV: GENERATING TEXT



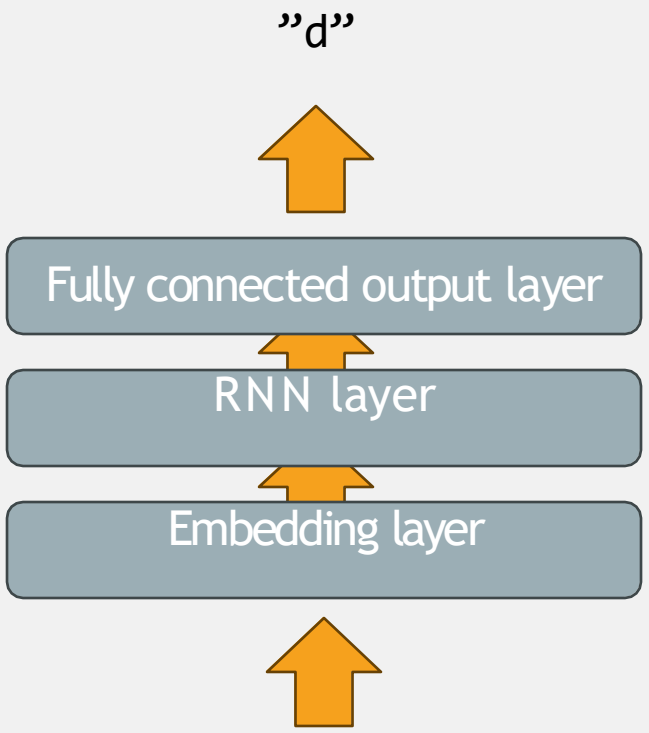
WRITING LIKE TOLKIEN IV: GENERATING TEXT



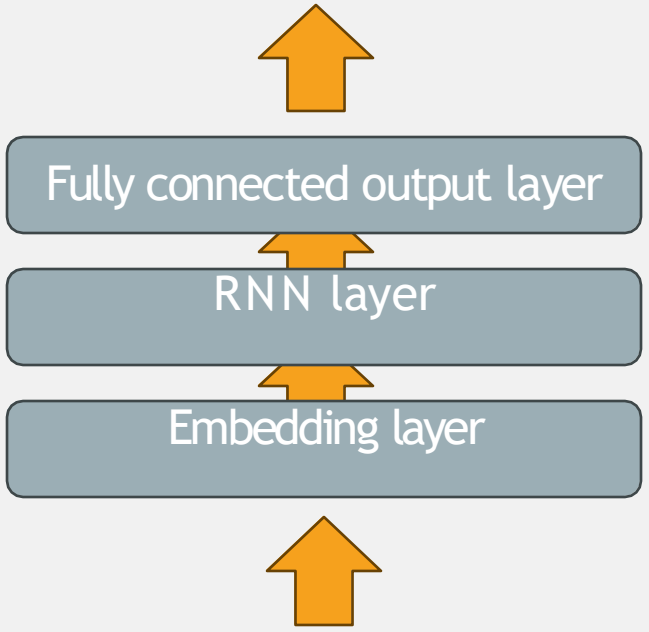
WRITING LIKE TOLKIEN IV: GENERATING TEXT



WRITING LIKE TOLKIEN IV: GENERATING TEXT



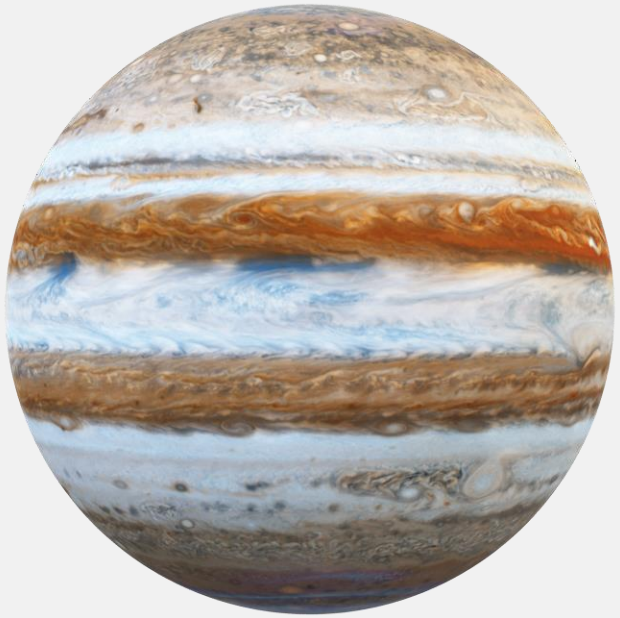
WRITING LIKE TOLKIEN IV: GENERATING TEXT



SMALL LANGUAGE MODELS

- Setting the stage
- Char-RNNs
- Implementing Char-RNNs
- Text classification
- Generating music

LET'S DO IT



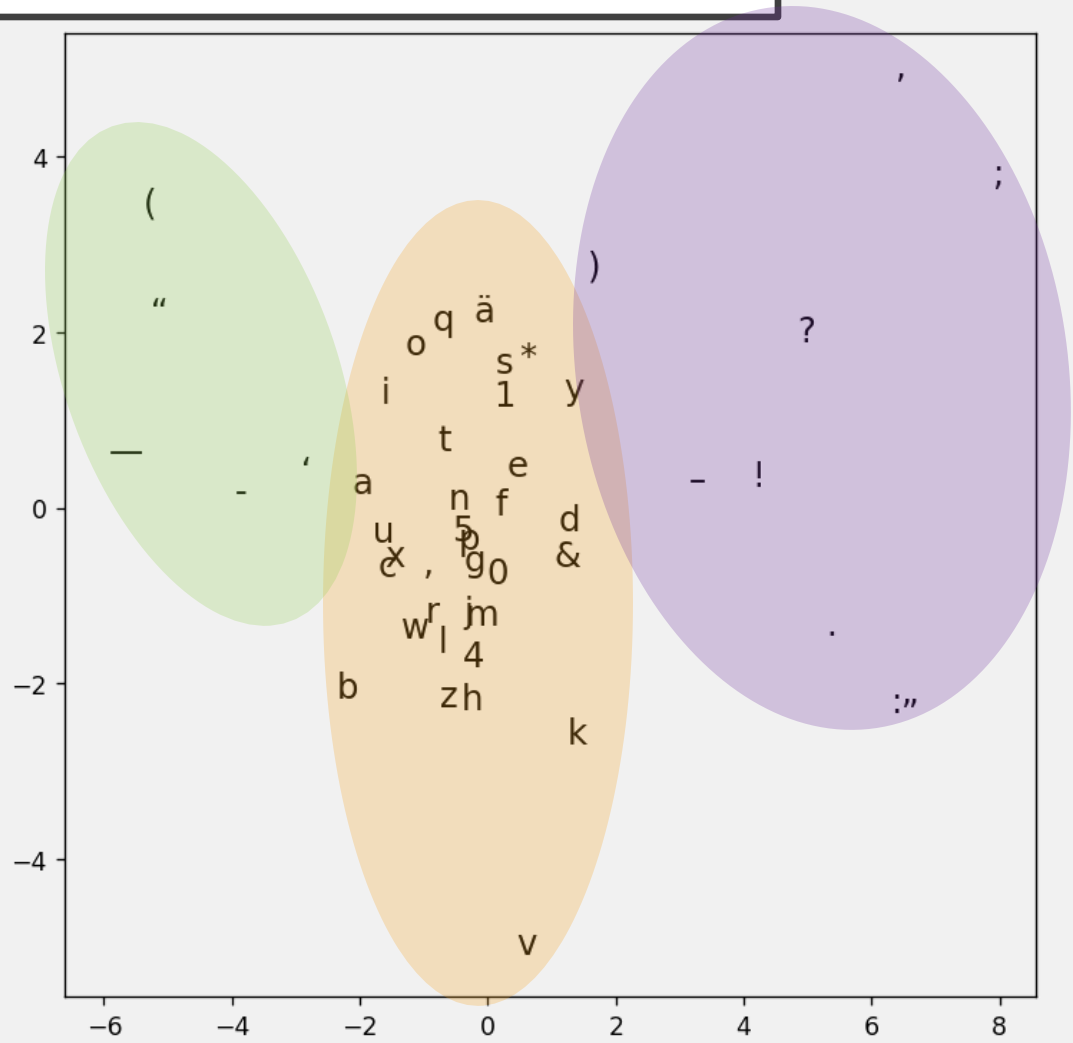
TWO THINGS TO THINK ABOUT

Do the embeddings represent anything meaningful?

What happens if we try to generate long text strings?

TWO THINGS TO THINK ABOUT

Do the embeddings represent anything meaningful?



TWO THINGS TO THINK ABOUT

What happens if we try to generate long text strings?

```
”gollum has some of the dwarves
and the dwarves and the dwarves
and the dwarves and the dwarves
and the dwarves and the dwarves
and the dwarves and the dwarves
and the dwarves and the dwarves
and the dwarves and the dwarves
and the dwarves and the dwarves
and the dwarves …”
```

If we always pick the most probable character, we usually end up repeating ourselves!

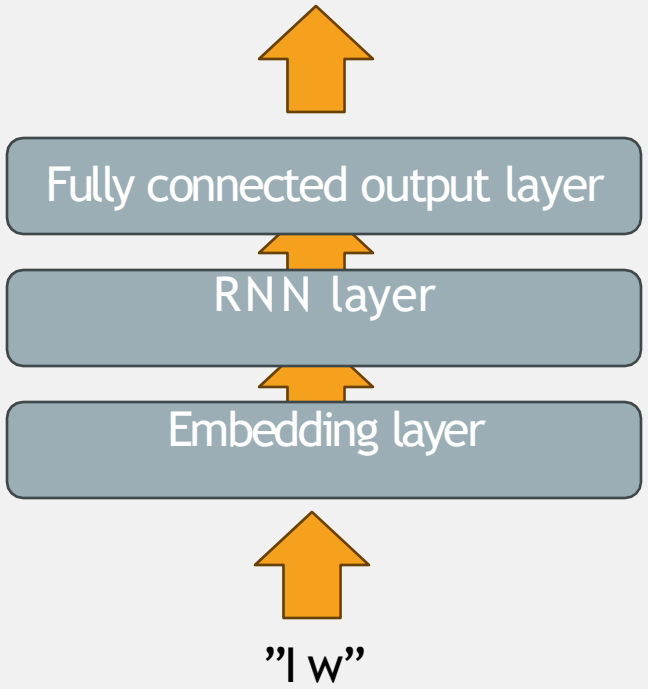
GENERATING LONG SENTENCES

"I want" "I went" "I wish"

a	b	c	d	e	f	g	h	i	…
0.31	0.01	0.03	0.02	0.12	0.01	0.01	0.02	0.25	…

Instead, sample from the probability distribution to get "a" 31 % of the time

but we can get more control



SOFTMAX AND TEMPERATURE

softmax $P_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$

temperate-adjusted softmax $P_i' = \frac{e^{z_i/T}}{\sum_j e^{z_i/T}}$

Low T $\Rightarrow$ large $\frac{z_i}{T}$ $\Rightarrow$ very large $e^{z_i/T}$ $\rightarrow$ differences exponentially exaggerated

a	b	c	d	e	f	g	h	i	...
1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...

High T $\Rightarrow$ small $\frac{z_i}{T}$ $\Rightarrow$ very small $e^{z_i/T}$ $\Rightarrow$ differences disappear

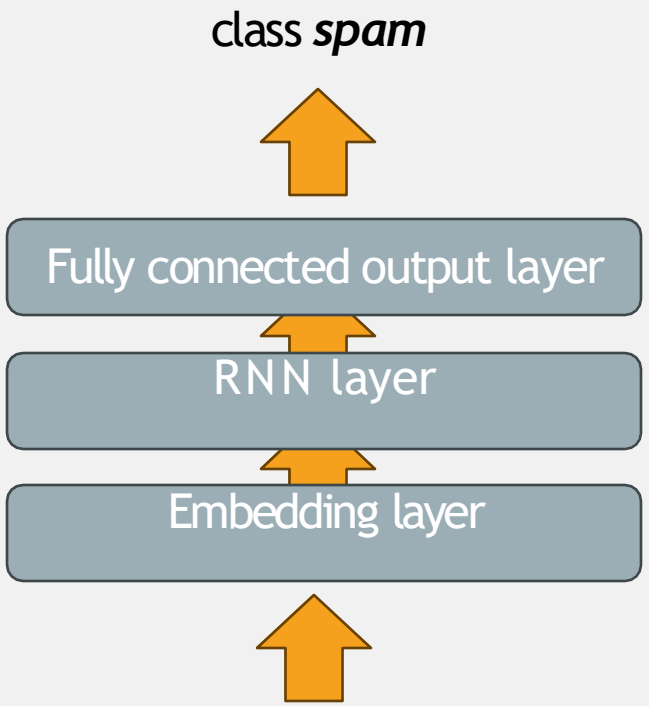
a	b	c	d	e	f	g	h	i	...
0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	...

SMALL LANGUAGE MODELS

- Setting the stage
- Char-RNNs
- Implementing Char-RNNs
- Text classification
- Generating music

TEXT CLASSIFICATION

Generating text can be fun, but is hardly the only use of language models



SMALL LANGUAGE MODELS

- Setting the stage
- Char-RNNs
- Implementing Char-RNNs
- Text classification
- Generating music

GENERATING MUSIC

G-G-D-D-E-E-D
C-C-B-B-A-A-G
D-D-C-C-B-B-A
D-D-C-C-B-B-A
G-G-D-D-E-E-D
C-C-B-B-A-A-G

```
e ----------------------------------------------
B ------------0------
G -------0-------0----
D -0-0-2-----0-----0--
A ----------------------------------------------
E ----------------3-----------3
```

Use a char-RNN to predict the next note

Great idea for a final project!