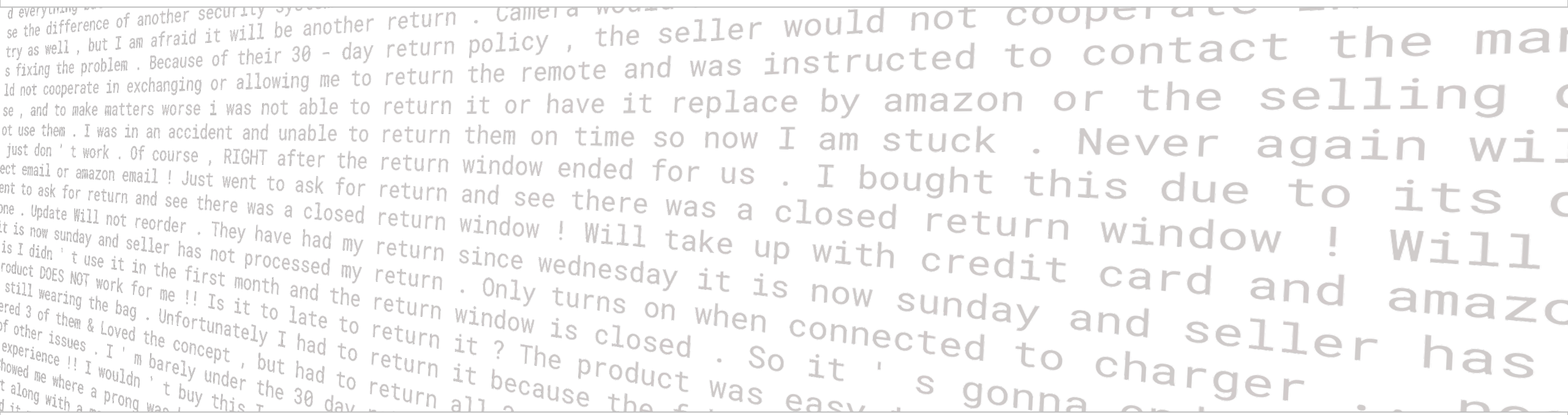
Computational Text Analysis:
Building your hybrid intelligence skills



Senthil Chandrasegaran
Jiwon Jung

What is Text Data?

Email, text messages, tweets, blogs

Product reviews, online forums

News articles, commentaries, speeches

Research papers, technical reports

Medical doctors' notes, patient forums

Books, manuals, and magazines

Computer programs

Why would you care about this stuff?

Why Computational Approaches?

Scale: make sense of large data

Detect **patterns** in the text data

Reveal **sentiments** and **emotions**

Compare document collections

Correlate text with other data

Identify **themes** and connections

Learning Objectives

By the end of today's workshop, you will be able to

- Programmatically open and pre-process structured and unstructured text files
- Apply different approaches to 'clean' and organize text data
- Use different approaches to identify salient information from large bodies of text
- Apply dictionary and other computational approaches to analyze text at scale

Managing Expectations

Today's workshop…

- …is not about learning Python
- …is not about efficient algorithmic techniques
- …is not about machine learning
  (though we may look into some basic techniques at the end)

Workshop Outline

1. Introduction

Why should designers care about text data?
Why use computational approaches?

2. Getting started with Python

Setting up your Jupyter Notebook
Basic (relevant) Python refresher
Reading a text file: lots of reviews
Reading structured data: reviews with metadata

3. Words, words, words

Tokenization
Counting words: more is more
Not all words are equal: stop words and the rest
Morphology and syntax: why do they matter?

4. More than words

Concordance: understanding context
Dictionary approaches: LIWC & Empath

5. Semantics

Document-Term Matrix
Topic Modeling

6. Embeddings (discussion)

Vector Spaces
Topic Modeling using transformer models

Workshop Outline

1. Introduction

Why should designers care about text data?
Why use computational approaches?

2. Getting started with Python

Setting up your Jupyter Notebook
Basic (relevant) Python refresher
Reading a text file: lots of reviews
Reading structured data: reviews with metadata

3. Words, words, words

Tokenization
Counting words: more is more
Not all words are equal: stop words and the rest
Morphology and syntax: why do they matter?

4. More than words

Concordance: understanding context
Dictionary approaches: LIWC & Empath

5. Semantics

Document-Term Matrix
Topic Modeling

6. Embeddings (discussion)

Vector Spaces
Topic Modeling using transformer models

A Case Study…

(over to Jiwon for her presentation)

Workshop Outline

1. Introduction

Why should designers care about text data?
Why use computational approaches?

2. Getting started with Python

Setting up your Jupyter Notebook
Basic (relevant) Python refresher
Reading a text file: lots of reviews
Reading structured data: reviews with metadata

3. Words, words, words

Tokenization
Counting words: more is more
Not all words are equal: stop words and the rest
Morphology and syntax: why do they matter?

4. More than words

Concordance: understanding context
Dictionary approaches: LIWC & Empath

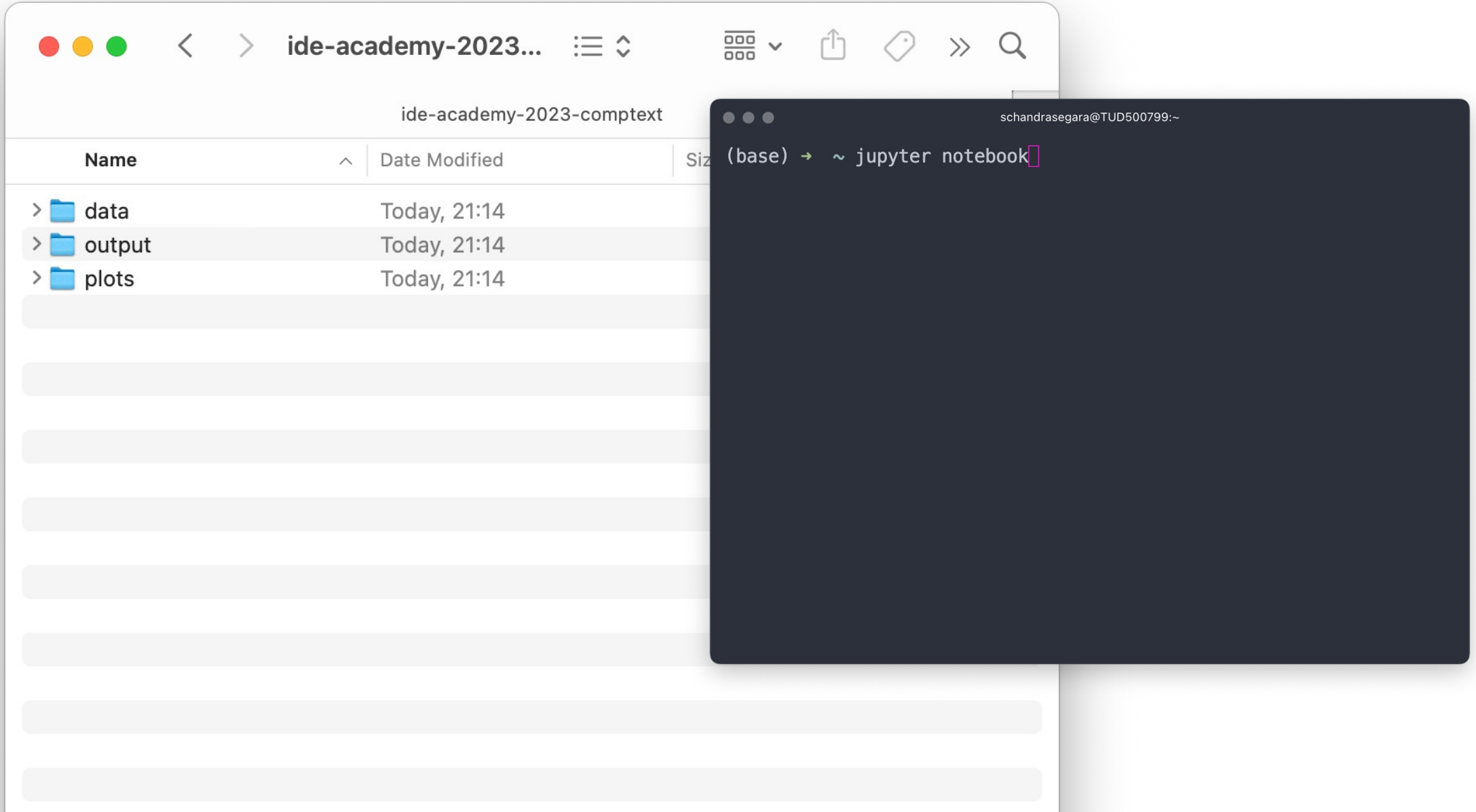
5. Semantics

Document-Term Matrix
Topic Modeling

6. Embeddings (discussion)

Vector Spaces
Topic Modeling using transformer models

Important: Launch Jupyter from the correct folder.



Work through the notebook with me

1. Python Refresher

Feel free to skip down to the part of the notebook that you think you need a refresher for. Topics covered are:

1. Using the Jupyter Notebook
2. Basic calculations
3. Variables and assignments
4. Datatypes (the kind you are likely to use in the workshop)
5. Conditional statements and loops

1. Using Jupyter Notebook

The jupyter notebook is one of several kinds of 'notebook interfaces' developed with the idea that programs should follow the logic and thought process of the scientist/analyst/developer, and not the requirements demanded by the programming language. A notebook is made of a series of 'cells': each cell is computed and the results of the last computation is displayed below the cell. For instance:

In [1]:
```
3 + 5
```

Out[1]: 8

You can also use the `print()` command to display specific outputs.

In [2]:
```
print(3 + 5)
```

8

Note the absence of an **`Out[ ]`** marker under the print command. This is because the command itself does not return a value to be displayed by the notebook cell. However, executing the command prints a value that is displayed below. This is a subtle difference, but something to keep in mind.

A cell can also be a space for you to write down your thoughts, explanations, decisions, and plans---any documentation that will help you return to this work at a later date and pick up where you left off, or disseminate this work to others. You can choose `code` (executable cell) or `markdown` (explanatory cell) on the toolbar at the top to choose what kind of cell you want to use.

There are several other ways of using the cells but these are the basic and most commonly used options.

2. Basic Calculations

Python's syntax is fairly human-readable, so let's dive into it with examples. First, you have the basic arithmetic calculations: addition, subtraction,

Workshop Outline

1. Introduction

Why should designers care about text data?
Why use computational approaches?

2. Getting started with Python

Setting up your Jupyter Notebook
Basic (relevant) Python refresher

3. Words, words, words

Reading a text file: lots of reviews
Tokenization & counting words
Not all words are equal: stop words
Morphology and syntax: why do they matter?
Reading structured data: reviews with metadata

4. More than words

Concordance: understanding context
Dictionary-based approaches

5. Semantics

Document-Term Matrix
Topic Modeling

6. Embeddings (discussion)

Vector Spaces
Topic Modeling using transformer models

Work through the notebook with me

2. Reading files

Load Review Text

Let's load a file where each line contains a review of one kind of product. You can choose the product you like from the `data/text/` folder; I'm going to go with camera reviews.

In [1]:
```
with open('./data/text/reviews_camera.txt', 'r', encoding='utf-8') as fo:
    reviews = fo.read()
```

Since the file contains severa reviews, we want to make sure that each review is separate from the other, i.e., the reviews should be in the form of a `list` and **not** a single `string` that has combined all reviews.

In [2]:
```
type(reviews)
```

Out[2]:
```
str
```

Looks like we have indeed done what we should not have done, i.e., read the text as a single string. Let's try that again, but with a different command to read each line in the file separately.

In [3]:
```
with open('./data/text/reviews_camera.txt', 'r', encoding='utf-8') as fo:
    reviews = fo.readlines()

type(reviews)
```

Out[3]:
```
list
```

Excellent. Now we can do things like check how many reviews (i.e., lines in the file) have been loaded, and even print out the first few reviews.

In [4]:
```
print("Number of reviews loaded: ", len(reviews))
print("-----------------------------------")
print("First 3 reviews:")
for review in reviews[0:3]:
    print(review)
```

```
Number of reviews loaded:  2139
-----------------------------------
First 3 reviews:
It doesn't work after less than a month. The contacts on the connection are cheap. Don't waste your money.

Doesn't work and tried contacting via email based off of the card that came with the item. Email does not work.
```

Work through the notebook with me

3. Making use of Metadata

We often have more than just the text. What can we do with the metadata?

In [1]:
```
import pandas as pd
import seaborn as sns
```

In [2]:
```
reviews_df = pd.read_json('data/json/amazon_reviews.json', lines=True, encoding='utf-8') # to prevent error due
reviews_df.sample(3)
```

Out[2]:

	review_id	product_id	reviewer_id	stars	review_body	review_title	language	product_category
194528	en_0211326	product_en_0085402	reviewer_en_0422986	5	For the IT specialist of a certain age. Don't ...	Irresistible	en	apparel
43677	en_0413148	product_en_0517672	reviewer_en_0204911	2	I replaced the clamp with a screw and it works...	The clamp is garbage	en	sports
111689	en_0722797	product_en_0875316	reviewer_en_0749063	3	I like how I can finally play my own music in ...	I like how I can finally play my own music in ...	en	wireless

Get statistics in case your data is quantitative.

In [3]:
```
reviews_df.describe()
```

Out[3]:

	stars
count	200000.000000
mean	3.000000
std	1.414217
min	1.000000
25%	2.000000
50%	3.000000
75%	4.000000
max	5.000000

Choose one product category

Let's focus on reviews of one product category.

Workshop Outline

1. Introduction

Why should designers care about text data?
Why use computational approaches?

2. Getting started with Python

Setting up your Jupyter Notebook
Basic (relevant) Python refresher

3. Words, words, words

Reading a text file: lots of reviews
Tokenization & counting words
Not all words are equal: stop words
Morphology and syntax: why do they matter?
Reading structured data: reviews with metadata

4. More than words

Concordance: understanding context
Dictionary-based approaches

5. Semantics

Document-Term Matrix
Topic Modeling

6. Embeddings (discussion)

Vector Spaces
Topic Modeling using transformer models

Dictionary Approaches

Revealing psychological states and processes

> “The idea behind LIWC is that the words people use reflect their feelings and that by the simple process of counting these words we can gain insights into their emotional states. We assume that angry people would use anger-related words; sad people would use sadness words.”
>
> – *James Pennebaker, The Secret Life of Pronouns (2011)*

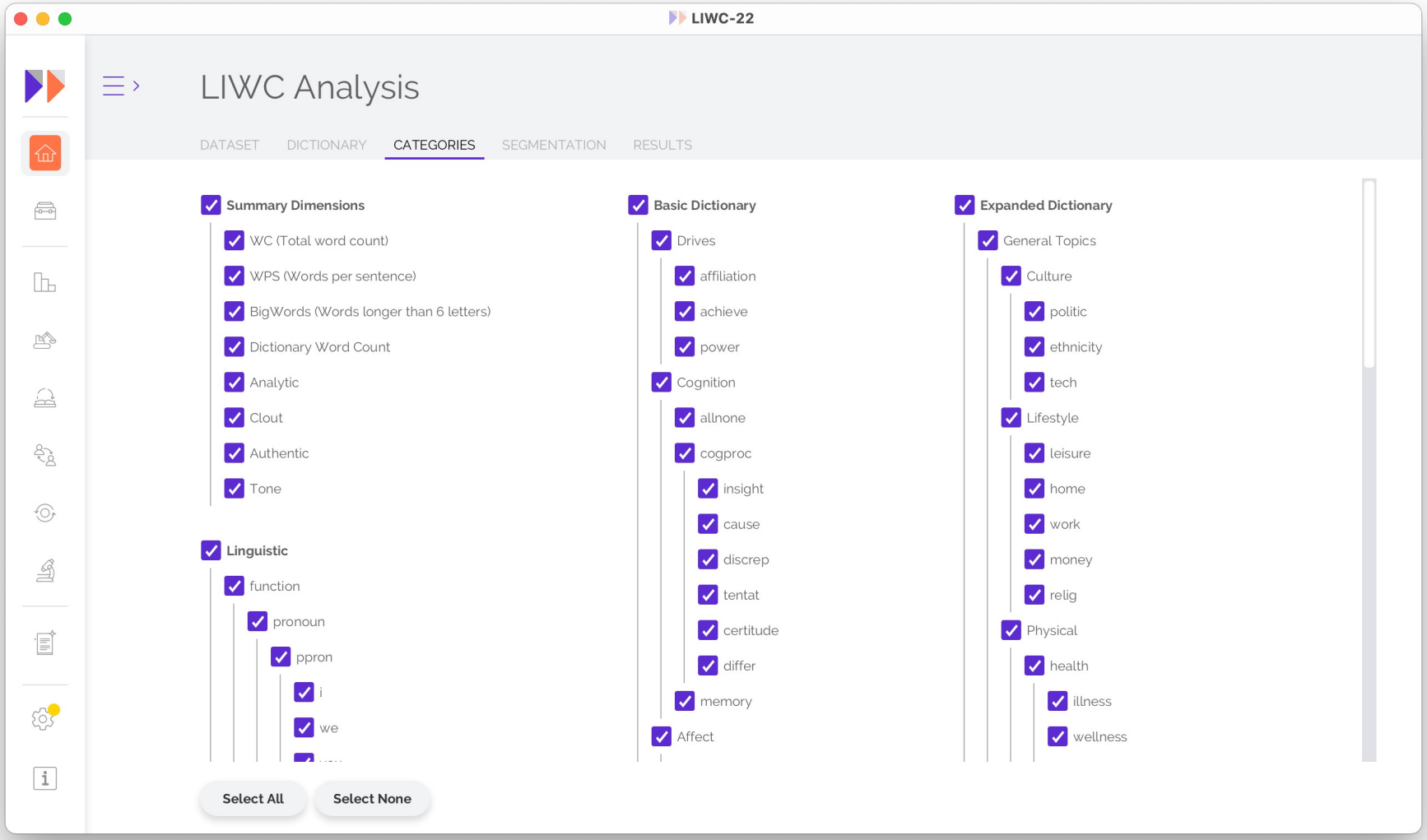
Dictionary Approaches

Revealing psychological states and processes

> "… people who were young, female, or both young and female are more likely to use discourse markers" *(e.g. like, I mean, you know)*
>
> "When having conversations with listeners, conscientious people use discourse markers … to imply their desire to share or rephrase opinions to recipients."
>
> – *Laserna et al. (2014)*

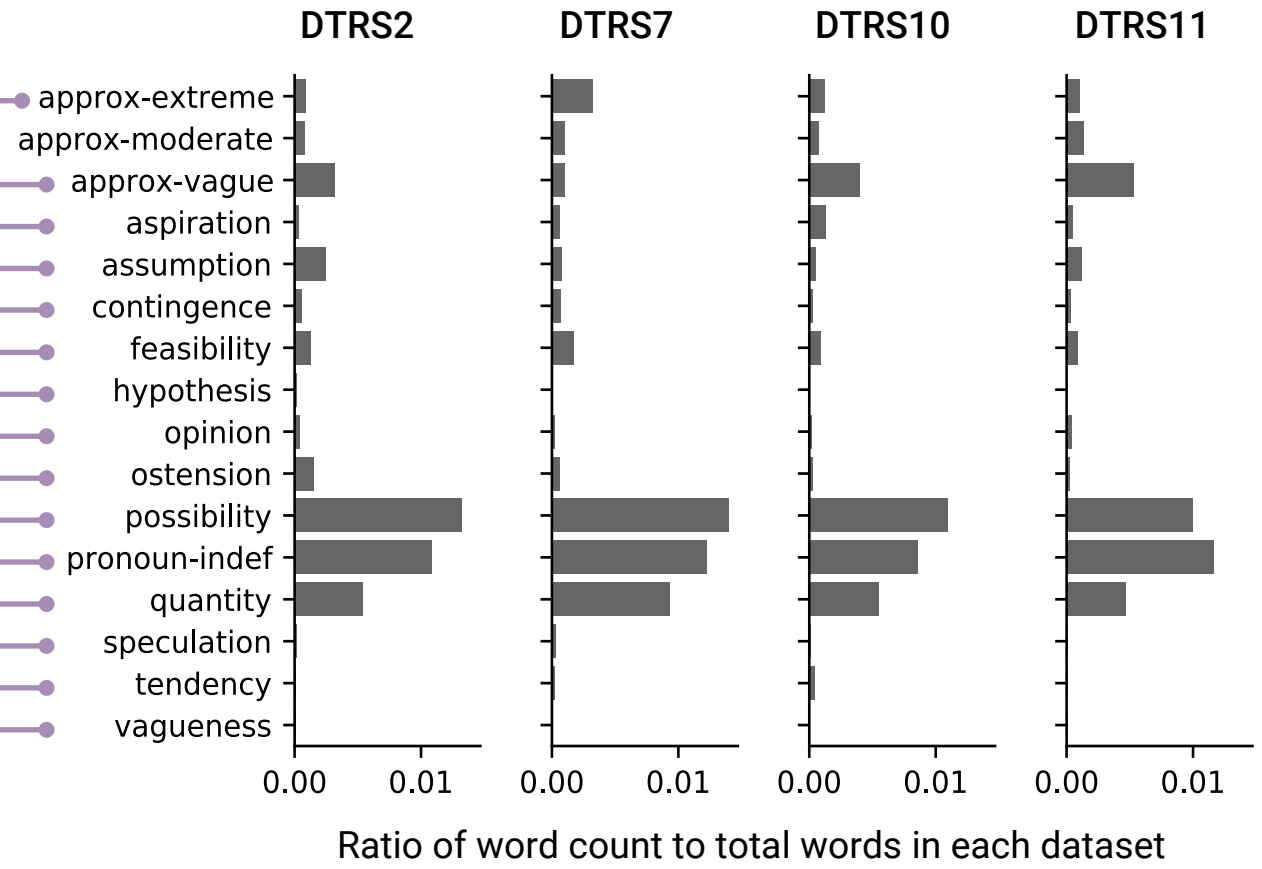
Laserna, C. M., Seih, Y. T., & Pennebaker, J. W. (2014). Um... who like says you know: Filler word use as a function of age, gender, and personality. Journal of Language and Social Psychology, 33(3), 328-338.

Dictionary Approaches



tentativeness

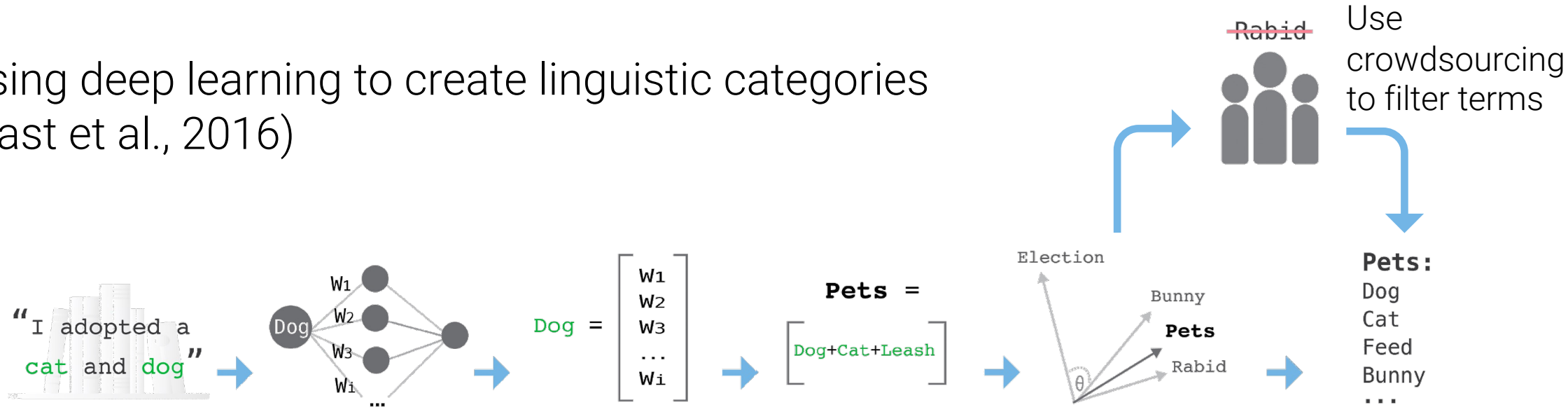
barely, hardly, almost, practically, virtually, fairly, pretty, quite	approx-extreme
approximately, partly, most, often, occasionally, sometimes, mainly, mostly, generally, typically, usually	approx-moderate
vaguely, kinda, kind of, sorta, somewhat	approx-vague
hope, hoped, hopefully, hoping, may, someday, sometime, potentially, theoretically, likely, unlikely	aspiration
assume, assumed, assumes, assuming, guess, guessing, suppose, supposed, supposing	assumption
contingent, depend, depending, depends, option, optional, optionally, options, random, randomly, variable, variables, varies, vary	contingence
chance, feasible, possibilities, possibility, possible, potential, try, trying	feasibility
assumption, assumptions, hypothesize, hypothetical, theory	hypothesis
bet, betting, doubting, hesitant, luck, lucky, opinion, question, questioning, questions, undecided	opinion
anyhow, apparently, appears, dunno, seem, seemed, seems, somehow, supposedly	ostension
if, maybe, might, perhaps, possibly, probably	possibility
any, anybody, anyone, anything, anytime, anywhere, or, somebody, somebody's, someone, someone's, something, something's, somewhere	pronoun-indef
alot, lot, lots, some, sort, sorts	quantity
doubt, wonder, wondered, wondering	speculation
appear, appearing, border, borderline, indirect, indirectly, lotta, overall, temporally, temporary, temporarily	tendency
blurred, confuse, confused, confusing, confusion, confusions, mystery, puzzle, puzzling, unclear, unknown, vague	vagueness



Lloyd, P., Akdag Salah, A., & Chandrasegaran, S. (2021).
How designers talk: Constructing and analysing a design thinking data corpus.
In Proceedings of ASME IDETC/CIE Conference.

Empath

Using deep learning to create linguistic categories (Fast et al., 2016)



Mine **600,000** stories (fiction)

Word prediction with neural network

Convert neural weights to **vectors**

User inputs seed words for **categories**

Find related terms based on **cosine similarity**

Final category **word list**

Fast, E., Chen, B., & Bernstein, M. S. (2016, May). Empath: Understanding topic signals in large-scale text. In Proceedings of the 2016 CHI conference on human factors in computing systems (pp. 4647-4657).

Work through the notebook with me

4. Exploring a dictionary-based approach with Empath

Empath (see Fast et al., 2016) is a tool for analysing a given corpus of text to identify the occurrence of certain pre-defined linguistic categories (similar to what is provided by LIWC), but also provides us with a way to create our own linguistic categories based on the behaviour we might want to examine.

Empath does not come pre-installed with standard python distributions so you would need to install it on your system using the following command in your terminal:

```
pip install empath
```

In [1]:
```
import pandas as pd
from empath import Empath
lexicon = Empath()
```

In [2]:
```
reviews_df = pd.read_json('data/json/amazon_reviews.json', lines=True, encoding='utf-8') # to prevent error due
reviews_df.sample(3)
```

Out[2]:

	review_id	product_id	reviewer_id	stars	review_body	review_title	language	product_category
10241	en_0807374	product_en_0604183	reviewer_en_0258752	1	I absolutely do not like these socks. The will...	One Star	en	apparel
73717	en_0862049	product_en_0886212	reviewer_en_0886594	2	It's personally too small for me. The clitoral...	Strong but small	en	drugstore
16739	en_0645633	product_en_0534417	reviewer_en_0487580	1	Didn't work as described and a waste of money	Didn't work	en	lawn_and_garden

In [3]:
```
camera_reviews_df = reviews_df[reviews_df['product_category'] == 'camera']
camera_reviews = list(camera_reviews_df['review_body'])
```

Get the list of categories from Empath

Empath has a set of predefined categories that you can print by using the following command.

In [4]:
```
lexicon.cats.keys()
```

Out[4]:
```
dict_keys(['help', 'office', 'dance', 'money', 'wedding', 'domestic_work', 'sleep', 'medical_emergency', 'cold', 'h
ate', 'cheerfulness', 'aggression', 'occupation', 'envy', 'anticipation', 'family', 'vacation', 'crime', 'attractiv
e', 'masculine', 'prison', 'health', 'pride', 'dispute', 'nervousness', 'government', 'weakness', 'horror', 'sweari
ng_terms', 'leisure', 'suffering', 'royalty', 'wealthy', 'tourism', 'furniture', 'school', 'magic', 'beach', 'journ
alism', 'morning', 'banking', 'social_media', 'exercise', 'night', 'kill', 'blue_collar_job', 'art', 'ridicule', 'p
lay', 'computer', 'college', 'optimism', 'stealing', 'real_estate', 'home', 'divine', 'sexual', 'fear', 'irritabili
ty', 'superhero', 'business', 'driving', 'pet', 'childish', 'cooking', 'exasperation', 'religion', 'hipster', 'inte
rnet', 'surprise', 'reading', 'worship', 'leader', 'independence', 'movement', 'body', 'noise', 'eating', 'medieval
```

Dictionary-Based Methods

What are the disadvantages?

- Polysemy
- Nuance
- Irony/sarcasm
- Negation
- … any others?

Workshop Outline

1. Introduction

Why should designers care about text data?
Why use computational approaches?

2. Getting started with Python

Setting up your Jupyter Notebook
Basic (relevant) Python refresher

3. Words, words, words

Reading a text file: lots of reviews
Tokenization & counting words
Not all words are equal: stop words
Morphology and syntax: why do they matter?
Reading structured data: reviews with metadata

4. More than words

Concordance: understanding context
Dictionary-based approaches

5. Semantics

Document-Term Matrix
Topic Modeling

6. Embeddings (discussion)

Vector Spaces
Topic Modeling using transformer models

Document-Term Matrix

Some terms are more “meaningful” than others

How relevant is the term “dog” in understanding what a document is about?

Total number of documents

$$Tf \cdot idf = Tf \cdot \log(\frac{N}{df(T)})$$

Number of times a term **T** appears in a document

Total number of documents containing term **T**

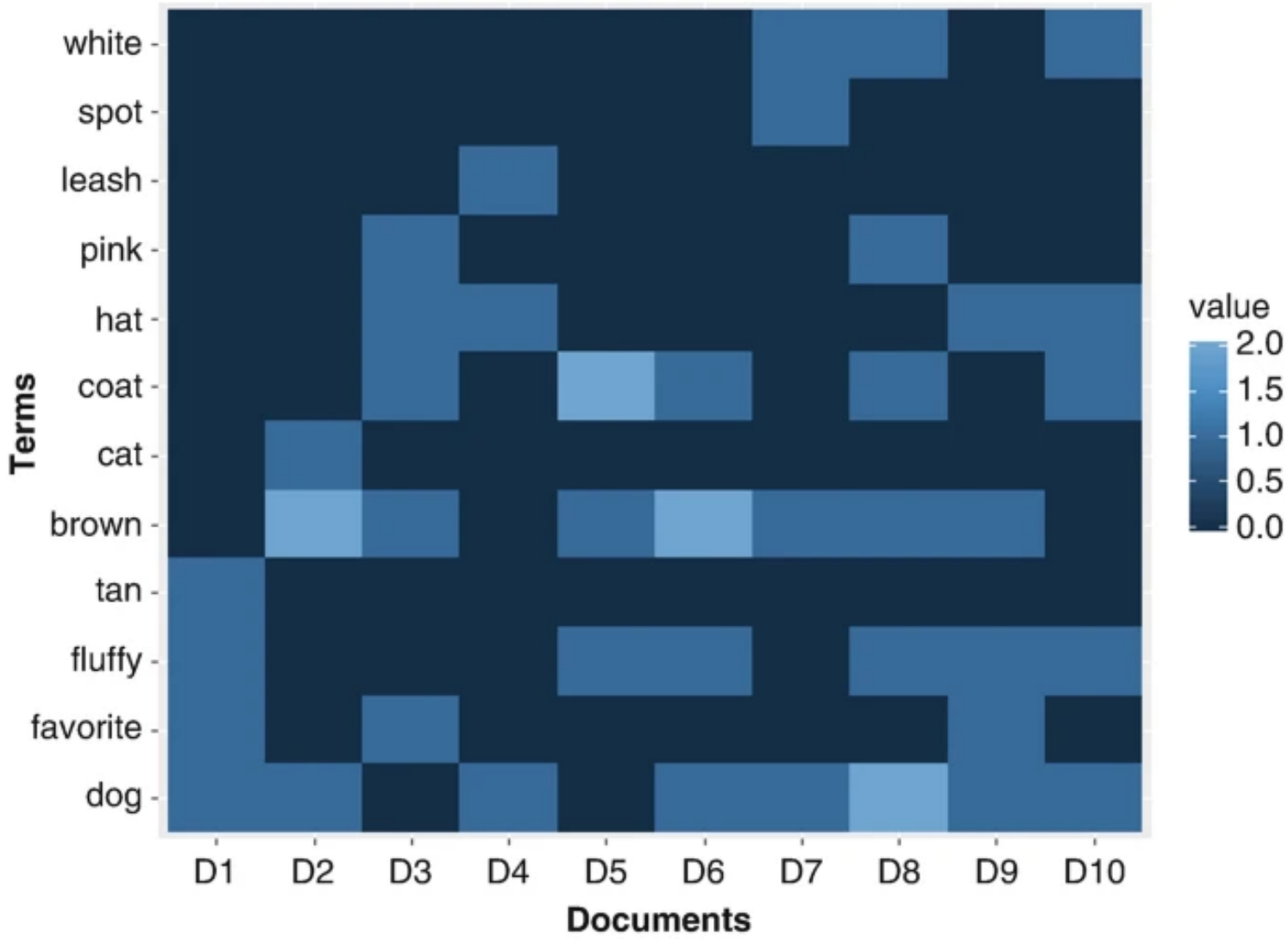


Image source: Anandarajan, M., Hill, C., Nolan, T., Anandarajan, M., Hill, C., & Nolan, T. (2019). Term-document representation. *Practical Text Analytics: Maximizing the Value of Text Data*, 61-73.

Topic Modeling: LDA

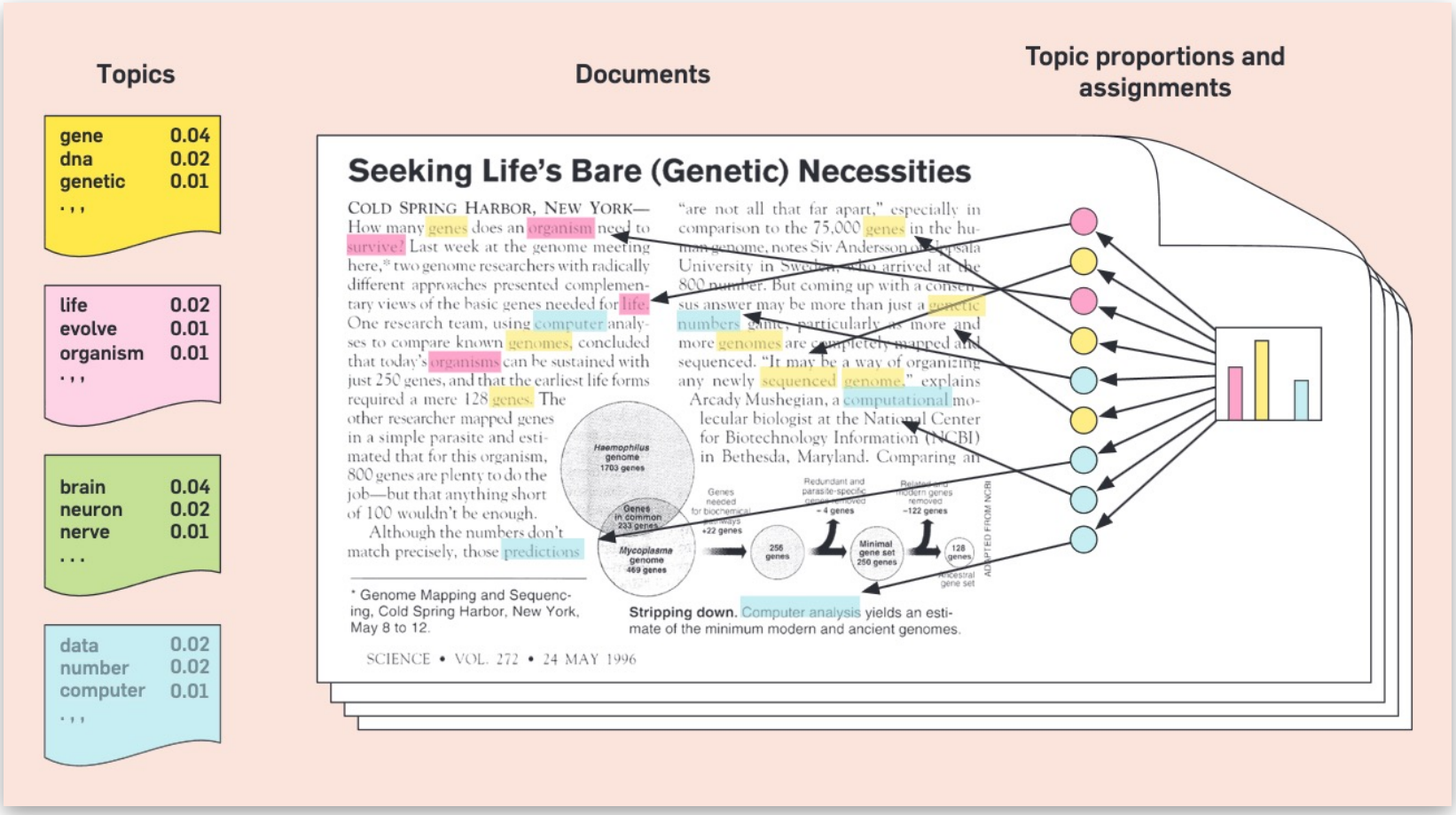
Assumptions

A number of “topics” exist for the collection of documents we are interested in.

A “topic” is a probability distribution over a vocabulary.

Each “document” is generated by a distribution over the topics (histogram) by:

- First, choose a topic randomly from the distribution.
- From the topic’s vocabulary, choose a word randomly.
- Repeat.



Blei, D. M. (2012). Probabilistic topic models. Communications of the ACM, 55(4), 77-84.

Work through the notebook with me

5. Topic Modeling with LDA

Use this notebook to practice topic modeling (see lecture slides for reference). First, the usual stuff.

In [1]:
```
import pandas as pd
reviews_df = pd.read_json('data/json/amazon_reviews.json', lines=True, encoding='utf-8') # to prevent error due
reviews_df.sample(3)
```

Out[1]:

	review_id	product_id	reviewer_id	stars	review_body	review_title	language	product_category
63015	en_0199099	product_en_0705066	reviewer_en_0806719	2	I hate to write bad reviews, but the fact this...	Hidden Thorn, potential dangerous on using the...	en	office_product
132688	en_0789121	product_en_0463343	reviewer_en_0155196	4	You can't see a few of the lighter colored words.	Cute sign	en	home
60394	en_0844842	product_en_0841159	reviewer_en_0242760	2	This product might be great, had it actually s...	Didn't stick	en	wireless

In [2]:
```
reviews = reviews_df['review_body'].tolist()
reviews[:3]
```

Out[2]:
```
["Arrived broken. Manufacturer defect. Two of the legs of the base were not completely formed, so there was no way
to insert the casters. I unpackaged the entire chair and hardware before noticing this. So, I'll spend twice the am
ount of time boxing up the whole useless thing and send it back with a 1-star review of part of a chair I never got
to sit in. I will go so far as to include a picture of what their injection molding and quality assurance process m
issed though. I will be hesitant to buy again. It makes me wonder if there aren't missing structures and supports t
hat don't impede the assembly process.",
 'the cabinet dot were all detached from backing... got me',
 "I received my first order of this product and it was broke so I ordered it again. The second one was broke in mor
e places than the first. I can't blame the shipping process as it's shrink wrapped and boxed."]
```

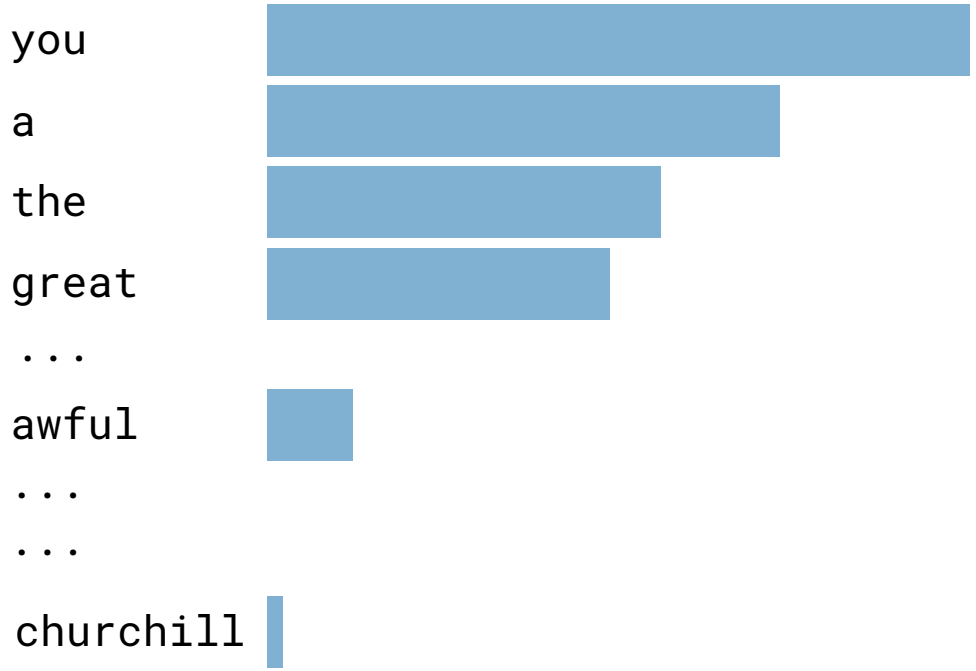
In [3]:
```
def remove_puncts(review_text, alphanumeric_only='True'):
    review_text = review_text.replace('-', ' ')
    clean_review_text = ''.join(e for e in review_text if e.isalnum() or e == ' ').lower()
    clean_review_text = ' '.join(clean_review_text.split())
    return clean_review_text
```

In [4]:
```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
# nltk.download('stopwords')
stop_words = set(stopwords.words('english'))

```

Embeddings

“The first rule of fight club is …”



Embeddings

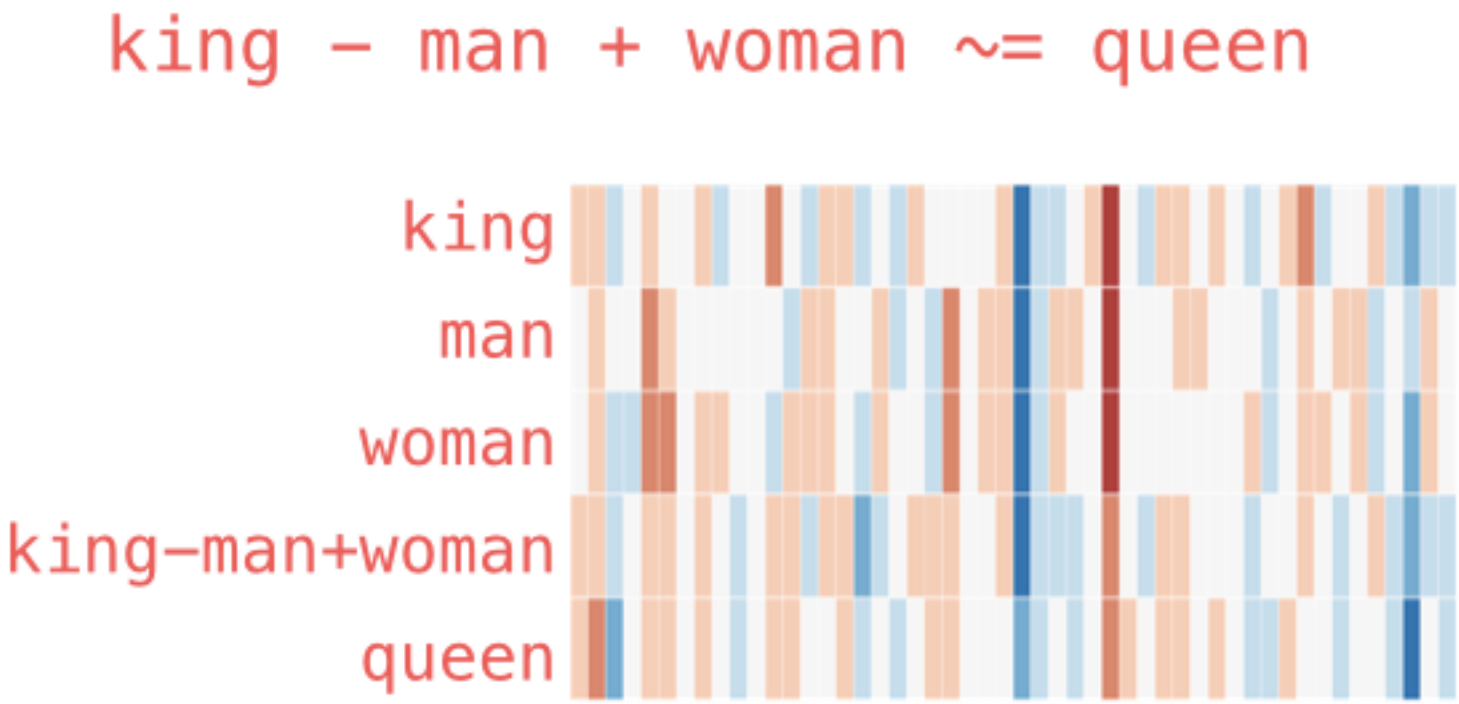


Image source: https://jalammar.github.io/illustrated-word2vec/

Embeddings

Encode nuances of language use

Encode a broader understanding of domains
(LDA is restricted to vocabulary of provided documents)

Allow for flexible approaches to thematic analysis (see BERTopic)

Workshop Takeaways

Text analysis at scale can provide (surprising) insights!

Decisions (remove stopwords? Lemmatize? etc.) not always straightforward: Document your rationale!

Always get an overview first, but then dive in to examine context!

Verify with close reading.

Keep exploring new approaches—we have only scratched the surface!

If you liked this workshop, you may enjoy…

ID5138 : Exploring Design Intelligence (EDI)

Offered in Q4. Enrollments close in early Q2!

2023/2024	Industrial Design Engineering	IO Electives
ID5138	Exploring Design Intelligence (EDI)	ECTS: 3

Course Coordinator	**Name** / **E-mail** Dr. R.S.K. Chandrasegaran / R.S.K.Chandrasegaran@tudelft.nl Dr. P.A. Lloyd / P.A.Lloyd@tudelft.nl
Contact Hours / Week x/x/x/x	
Education Period	4
Start Education	4
Exam Period	none
Course Language	English
Expected prior knowledge	Some experience with programming can be helpful but is not necessary.
Course Contents	Exploring Design Intelligence will provide a theoretical and practical foundation in the tools and methods with which to analyse design language and conversation. This provides the basis for using AI systems to model and generate design conversation and a route to producing AI design methods. The course will develop technical and research competencies alongside deeper thinking about the process of design.
Study Goals	The learning objectives of EDI are as follows: LO1: Explain the characteristics and features of design conversation. LO2: Characterise a design conversation by analysing it based on existing theories of designing LO3: Create an AI design method using Machine Learning-based text generation techniques.
Education Method	EDI will adopt a blended learning, flipped classroom model where lectures, videos, podcasts, and reading material will be used to inform the student of relevant theories and approaches. Students will explore the material and reinforce their understanding of concepts through studies and explorations that they will perform in a studio-based environment. Assignments will help students learn to use the tools and methods introduced in the course.
Literature and Study Materials	Podcasts, YouTube, TED, etc. ('found material'), Notebooks to structure computational activity, Miro. Sample References: