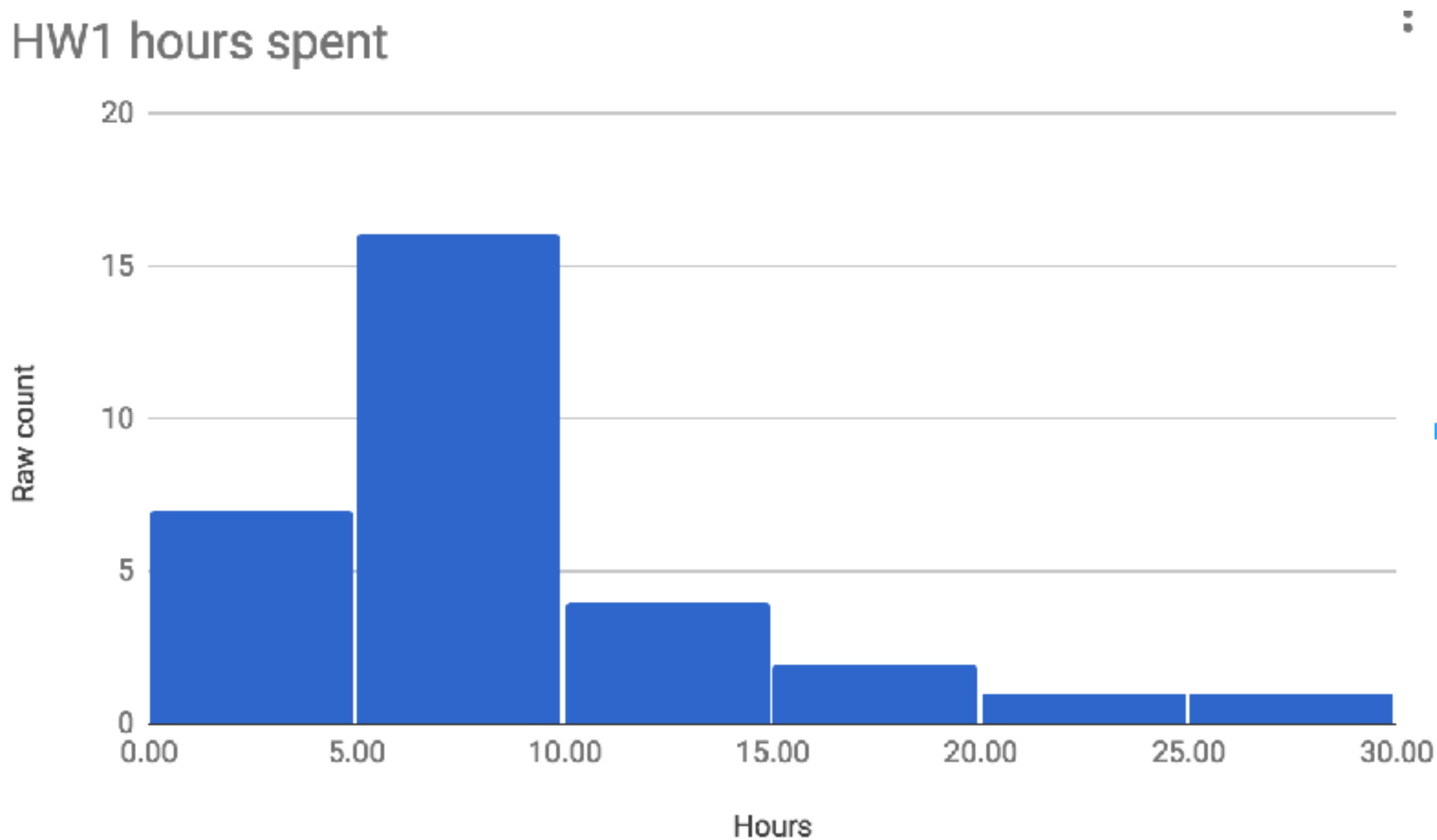
Carnegie Mellon University
Heinz College

95-865: Clustering wrap-up, topic modeling with Latent Dirichlet Allocation (LDA)

George Chen

Questionnaire Results



For the most part: students seem to want more demos

Reminders

- Your mid-mini quiz is on Wednesday (coverage: up to today)
    - Bring a laptop with Anaconda Python 3.6 (which includes Jupyter) installed
    - Make sure you have enough laptop battery life, or sit near an outlet
    - Open notes (so you can look at homework and demos)
    - Open internet (e.g., you can search up API documentation; cite external sources if it's not official API documentation such as stackoverflow)
    - No collaboration
- My office hours this week: Tuesday 5pm-7pm, HBH 2216

How to Choose a Clustering Method?

In general: not easy!

Some questions to think about:

- What features to even cluster on?
- For your application, what distance/similarity makes sense?
- Do you care about cluster assignments for new points?
- After you run a clustering algorithm, look at what data points ended up in the same cluster and make visualizations (e.g., histogram of various feature values)
  - Can you interpret the clusters?
  - Compare the cluster centers: do any of the centers for different clusters appear too close?
- Can you come up with some heuristic score function to say how good a cluster assignment is?

Clustering Last Remarks

- It's possible that several clustering methods give similar results (*which is great!* — it means that there are some reasonably "stable" clusters in your data)
  - Example: *tons* of clustering methods can figure out from senate voting data who Democrats and Republicans are (of course, *without* knowing each senator's political party)
- Ultimately, *you* have to decide on which clustering method and number of clusters make sense for your data
  - Do not just blindly rely on numerical metrics (e.g., CH index)
  - Interpret the clustering results in the context of the application you are looking at

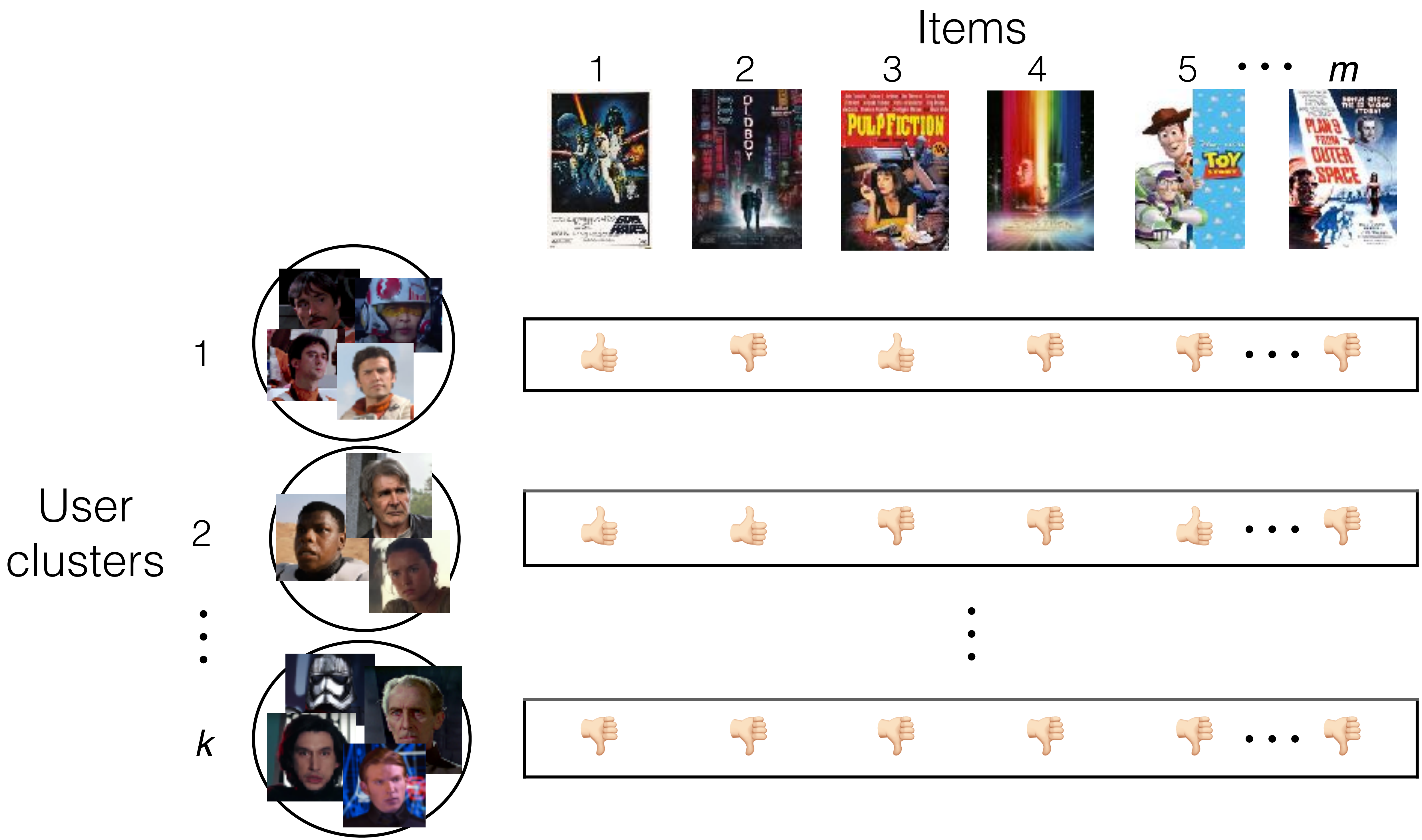
If you can set up a prediction task, then you can use the prediction task to guide the clustering

A Sketch of Interpreting Clusters

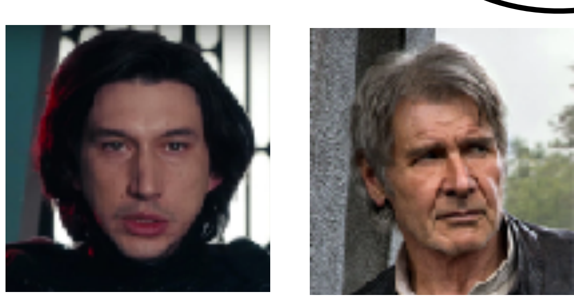
Demo

Is Clustering Structure Enough?

Is Clustering Structure Enough?

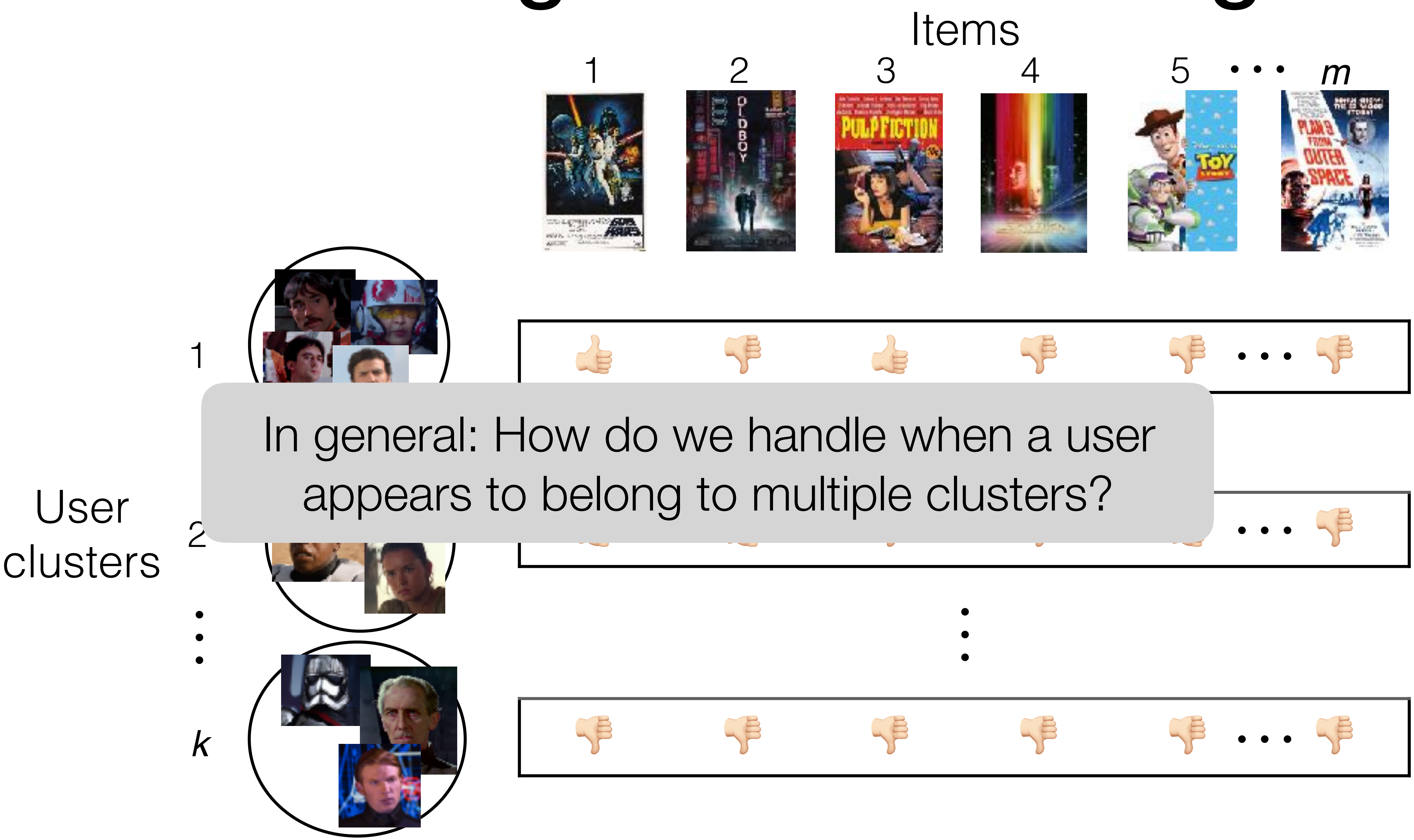


Is Clustering Structure Enough?

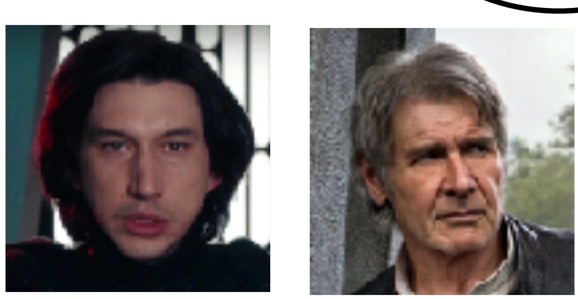


What if these two users shared a Netflix account (and used the same user profile)?

Is Clustering Structure Enough?



In general: How do we handle when a user appears to belong to multiple clusters?



What if these two users shared a Netflix account (and used the same user profile)?

Topic Modeling

Movie recommendation

Each user is part of multiple “clusters”/topics

Each cluster/topic consists of a bunch of movies
(example clusters: “sci-fi epics”, “cheesy rom-coms”)

Text

Each document is part of multiple topics

Each topic consists of a bunch of regularly co-occurring words
(example topics: “sports”, “medicine”, “movies”, “finance”)

Health care

Each patient’s health records explained by multiple “topics”

Each topic consists of co-occurring “events”
(example topics: “heart condition”, “severe pancreatitis”)

Topic Modeling

Movie recommendation

Each user is part of multiple “clusters”/topics

Each cluster/topic consists of a bunch of movies

(ex s”)

In all of these examples:

- Each data point (a feature vector) is part of multiple topics
- Each topic corresponds to specific feature values in the feature vector likely appearing

Each to words

(exar nce”)

Health care

Each patient’s health records explained by multiple “topics”

Each topic consists of co-occurring “events”

(example topics: “heart condition”, “severe pancreatitis”)

Latent Dirichlet Allocation (LDA)

- Easy to describe in terms of text (but works for not just text)
- Input: “document-word” matrix, and pre-specified # topics *k*

i-th row, *j*-th column: # times word *j* appears in doc *i*

- Output: what the *k* topics are (details on this shortly)

LDA Example

		Document	
		Alice's text	Bob's text
Topic	weather	0.1	0.5
	food	0.9	0.5

		Topic	
		weather	food
Word	cold	0.3	0.1
	hot	0.7	0.3
	apple	0.0	0.5
	pie	0.0	0.1

Each word in Alice's text is generated by:

1. Flip 2-sided coin for Alice
2. If weather: flip 4-sided coin for weather
   If food: flip 4-sided coin for food

LDA Example

		Document	
		Alice's text	Bob's text
Topic	weather	0.1	0.5
	food	0.9	0.5

		Topic	
		weather	food
Word	cold	0.3	0.1
	hot	0.7	0.3
	apple	0.0	0.5
	pie	0.0	0.1

Each word in Bob's text is generated by:

1. Flip 2-sided coin for Bob
2. If weather: flip 4-sided coin for weather
   If food: flip 4-sided coin for food

LDA Example

		Document	
		Alice's text	Bob's text
Topic	weather	0.1	0.5
	food	0.9	0.5

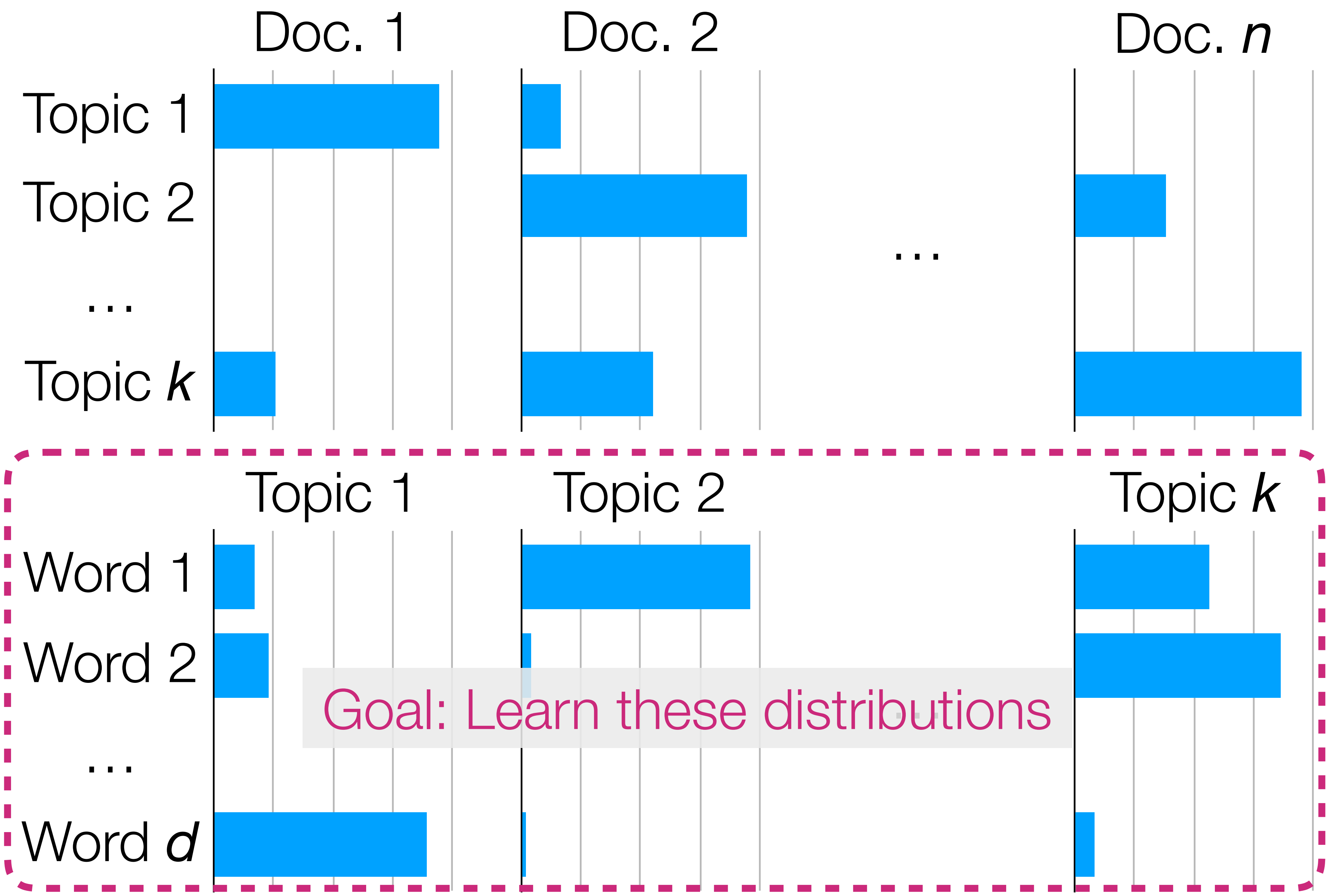
		Topic	
		weather	food
Word	cold	0.3	0.1
	hot	0.7	0.3
	apple	0.0	0.5
	pie	0.0	0.1

Each word in doc. *i* is generated by:

1. Flip 2-sided coin for doc. *i*
2. If weather: flip 4-sided coin for weather
   If food: flip 4-sided coin for food

"Learning the topics" means figuring out these 4-sided coin probabilities

LDA



LDA models each word in document *i* to be generated as:

- Randomly choose a topic *Z* (use topic distribution for doc *i*)
- Randomly choose a word (use word distribution for topic *Z*)

LDA

- Easy to describe in terms of text (but works for not just text)
- Input: “document-word” matrix, and pre-specified # topics *k*

$$
\begin{array}{cc|cccc|}
 & & & \textbf{Word} & & \\
 & & 1 & 2 & \ldots & d \\
\hline
 & 1 & & & & \\
 & 2 & & & & \\
\textbf{Document} & \vdots & & & & \\
 & n & & & & \\
\hline
\end{array}
$$

i-th row, *j*-th column: # times word *j* appears in doc *i*

- Output: the *k* topics’ distribution of words

LDA

Demo

How to Choose Number of Topics *k*?

Bayesian nonparametric variant of LDA:
Hierarchical Dirichlet Process (HDP)

(similar to how we went from GMM to DP-GMM)

Something like CH index is also possible:

For a specific topic, look at the *m* most probable words ("top words")

Coherence (within cluster/topic variability):

$$\sum_{\substack{\text{top words } v,w \\ \text{that are not the same}}} \log \frac{(\text{\# documents with at least one appearance of } v \text{ and } w) + \varepsilon}{\text{\# documents with at least one appearance of } w}$$

choose something small like 0.01

Inter-topic similarity (between cluster/topic variability):

Count # top words that do not appear in any of the other topics' *m* top words

(number of "unique words")

Can average each of these across the topics