Carnegie Mellon University
Heinz College

94-775 Unstructured Data Analytics for Policy

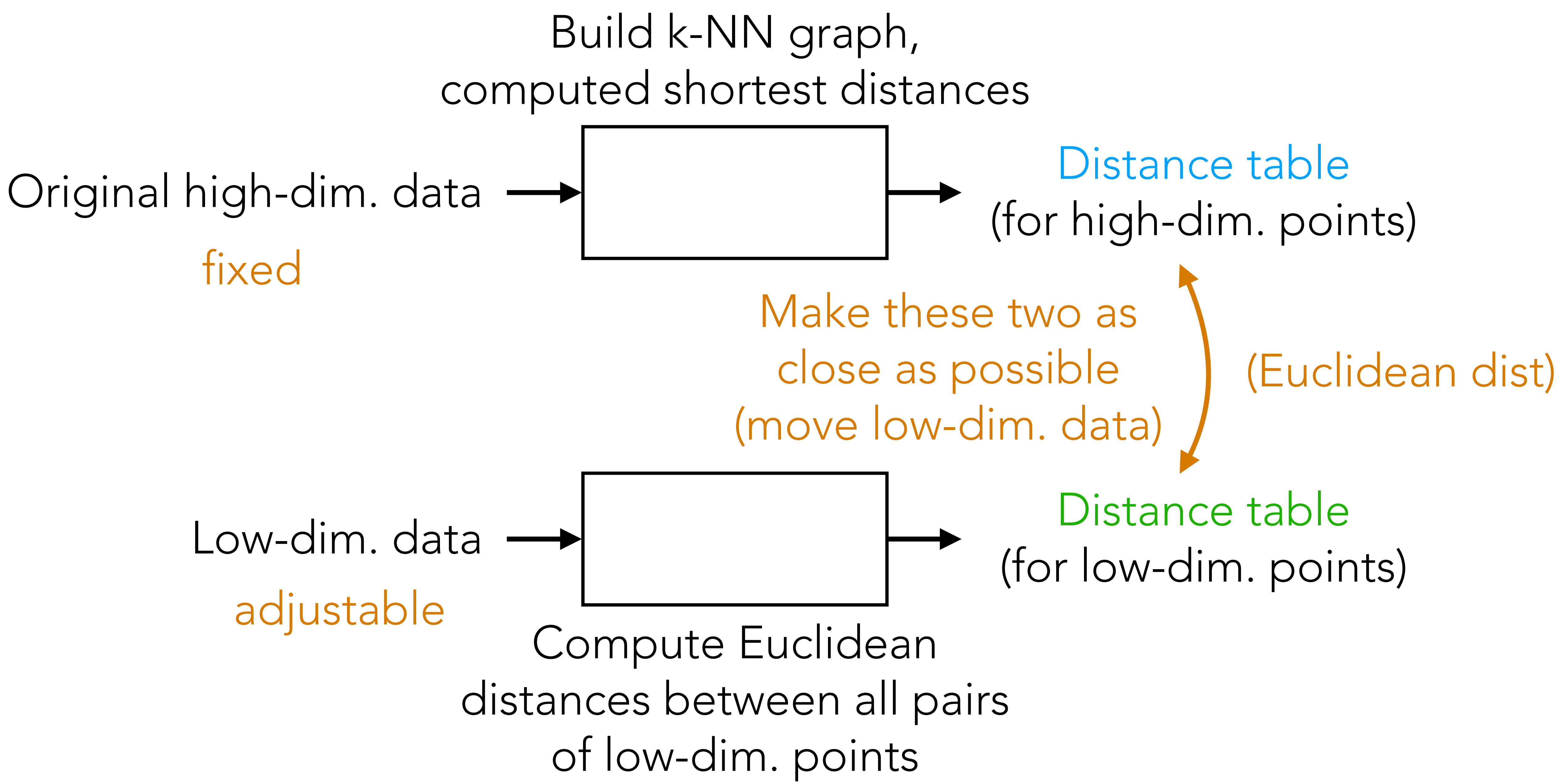
Lecture 6: Manifold learning (cont'd)

Slides by George H. Chen

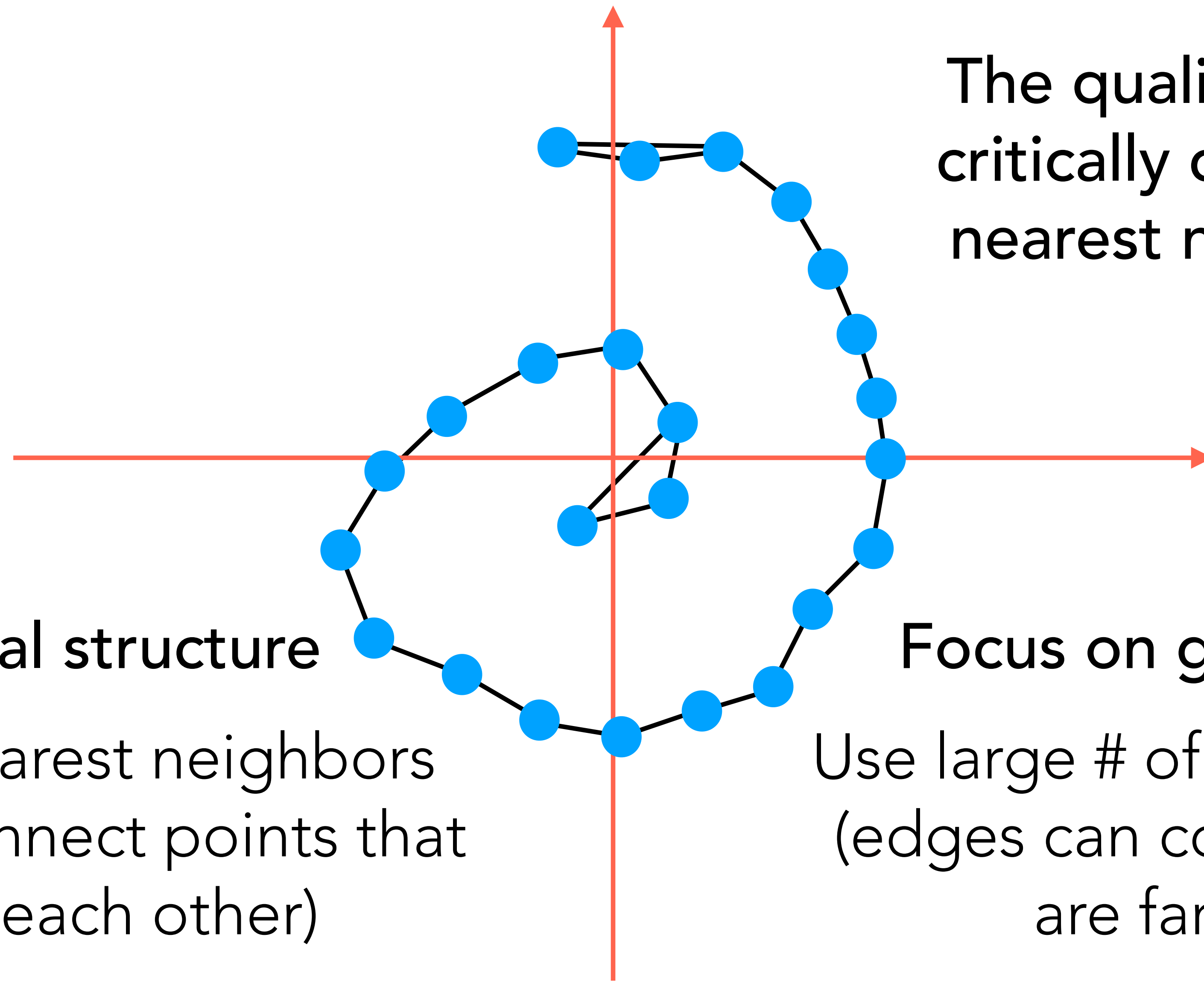
Administrivia

- HW1 is due tonight 11:59pm
- Your Quiz 1 is this Friday during your recitation slot (first 40 min) 😱
- There is an optional Quiz 1 review session on Wednesday 3/25 7pm-8pm over Zoom (run by your TA Shurui) that focuses on the Spring 2025 Quiz 1 but you're welcome to ask other questions too
  - We'll post solutions to all practice quizzes released around the time of the review session
- Reminder: final project proposals are due next Friday 4/3 at 11:59pm

(Flashback) Isomap



Some Observations on Isomap



The quality of the result critically depends on the nearest neighbor graph

Focus on local structure

Use small # of nearest neighbors (edges tend to connect points that are closer to each other)

There might not be enough edges

Focus on global structure

Use large # of nearest neighbors (edges can connect points that are farther apart)

Might connect points that shouldn't be connected

In general: try different parameters for nearest neighbor graph construction when using Isomap + visualize

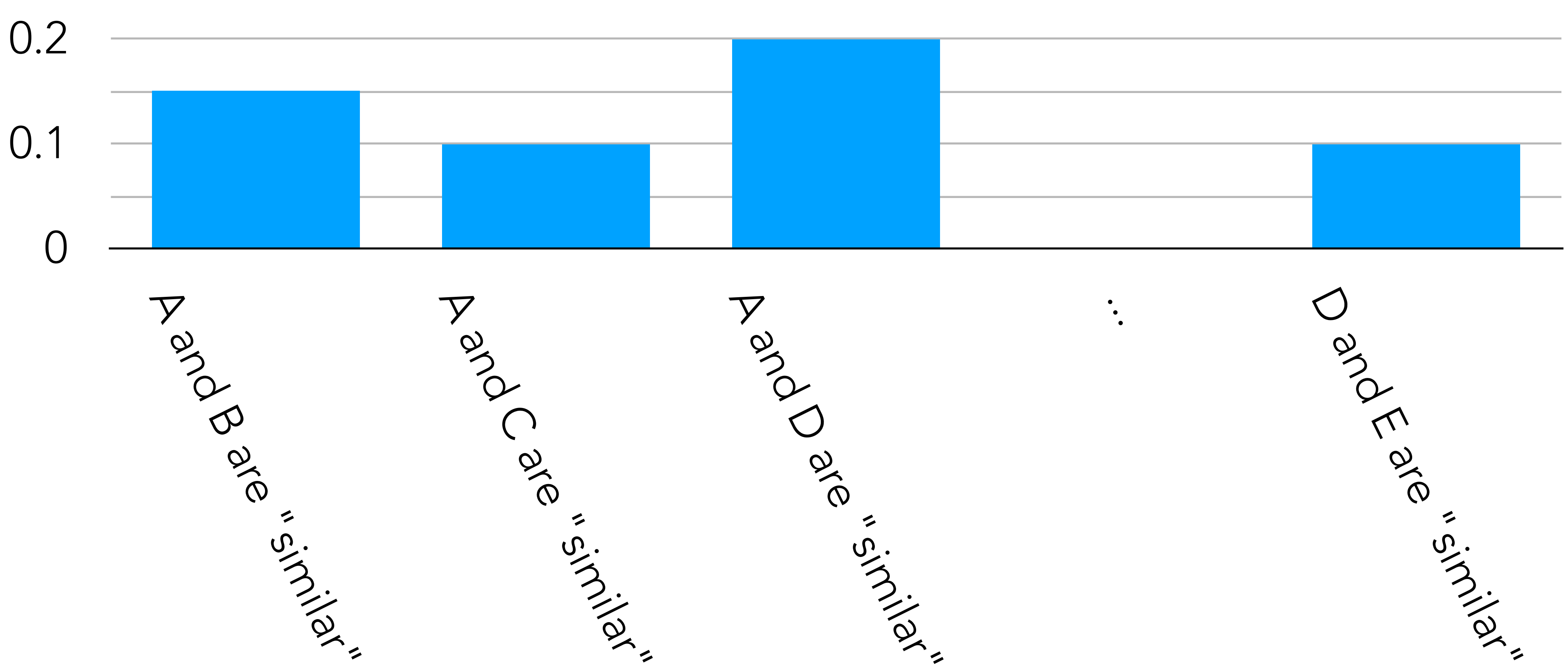
t-SNE
(t-distributed stochastic neighbor embedding)

High-level t-SNE Idea

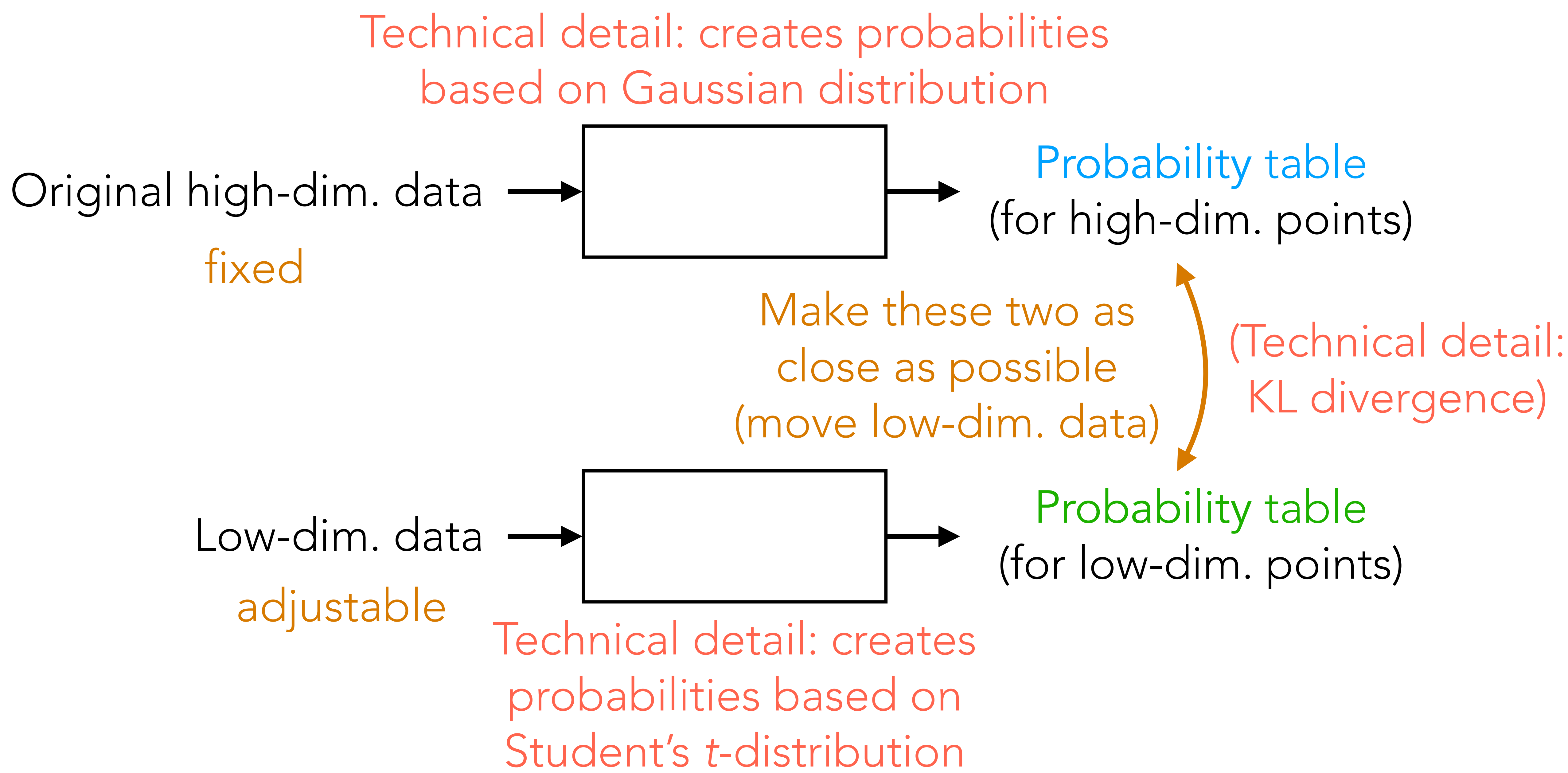
- Don't use deterministic definition of which points are neighbors

	A	B	C	D	E
A	0	5	8	13	16
B	5	0	5	10	13
C	8	5	0	5	8
D	13	10	5	0	5
E	16	13	8	5	0

- Use probabilistic notation instead



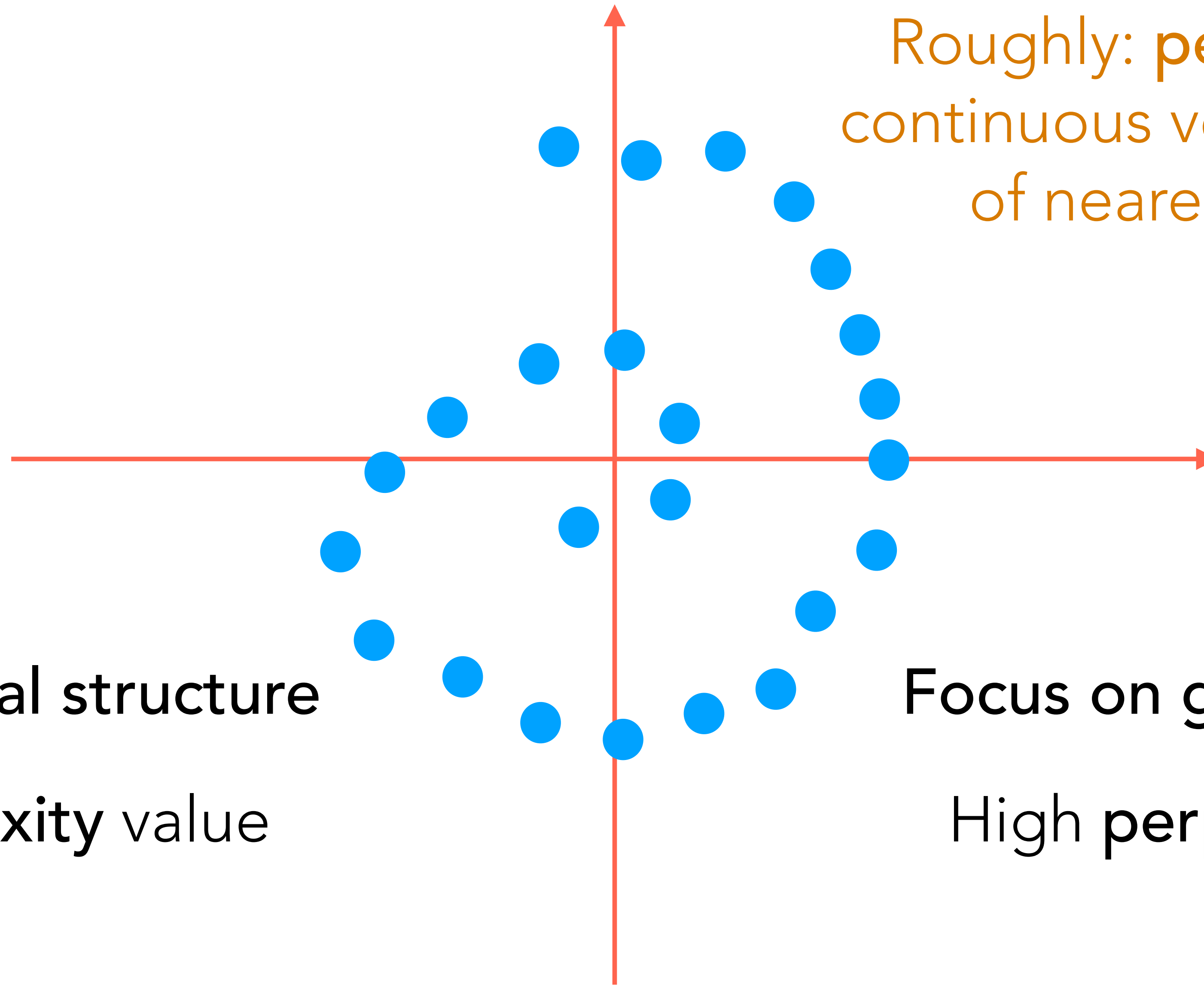
t-SNE



Technical details are in separate slides (posted on webpage)

t-SNE Parameters…

Roughly: **perplexity** is like a continuous version of “number of nearest neighbors”



Focus on local structure

Low **perplexity** value

Focus on global structure

High **perplexity** value

Also: play with # iterations, learning rate

how many times to try to improve guess of low-dim. representation

each time we try to improve low-dim. representation, how much we can change it

In practice, often people initialize with PCA

Manifold Learning with t-SNE

Demo

Interpreting t-SNE Plots

Required reading for this class:

"How to Use t-SNE Effectively" (Wattenberg et al 2016)

https://distill.pub/2016/misread-tsne/

Uniform Manifold Approximation and Projection (UMAP)

UMAP has recently become popular as an alternative to t-SNE *since it runs much faster than t-SNE* & often produces good visualizations too

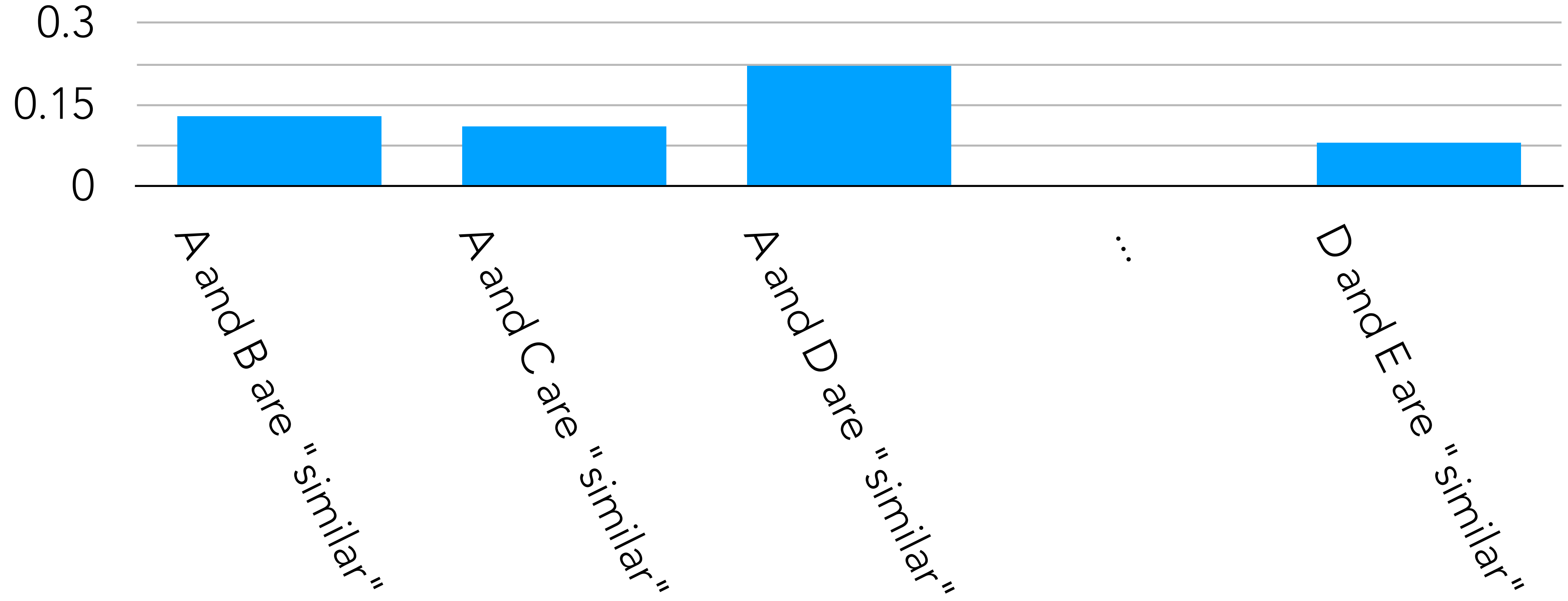
At a high-level, it's actually in many ways similar to t-SNE

For the purposes of this class, I want you to understand interpreting t-SNE plots in detail

High-level UMAP Idea

- Like t-SNE: define pairwise scores for how similar pairs of points are

- Use "membership strengths" instead (between 0 and 1, technically not probabilities since they do not in general sum to 1)



UMAP

Can be computed very quickly with the help of the k-NN graph, which speeds up model fitting

Technical detail: construct k-NN graph, use it to calculate membership strengths

Original high-dim. data

fixed

Membership strengths
(for high-dim. points)

Make these two as close as possible
(move low-dim. data)

(Technical detail: cross entropy)

Low-dim. data

adjustable

Membership strengths
(for low-dim. points)

Technical detail: creates membership strengths based on a *generalized* t-distribution

For technical details, see supplemental materials (not required for the class)

Manifold Learning with UMAP

Demo

Let’s look at images

(Flashback) Multiple Documents

Representing data as feature vectors is very general, not just for text!

Choose a common vocabulary to use across all documents

☀ ☁ ☂ ☃

Text doc #1 → [TF vector #1]

Text doc #2 → [TF vector #2]

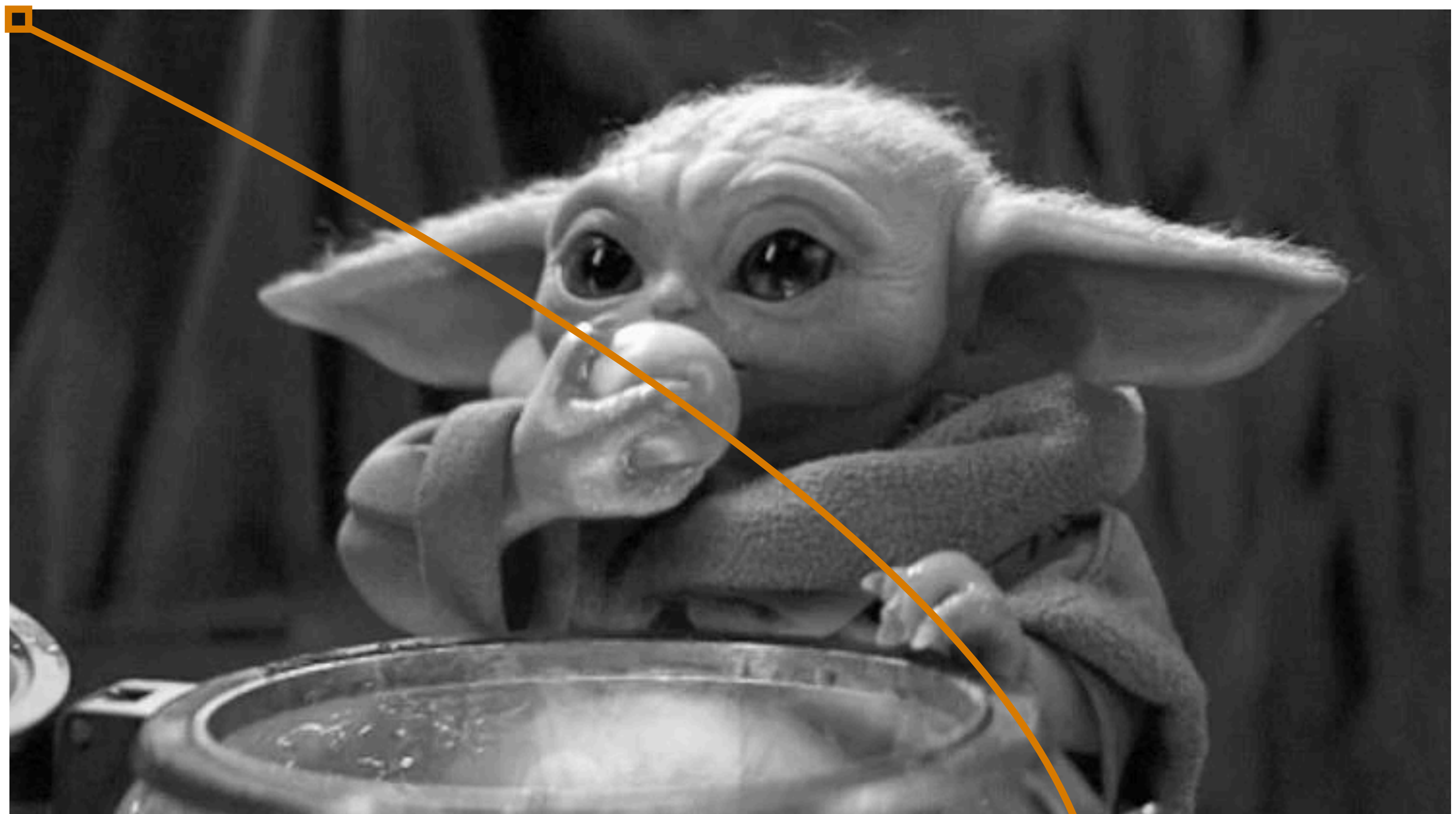
⋮

Text doc #*n* → [TF vector #*n*]

HW1 asks you to implement this idea to construct TF vectors for different text docs corresponding to job descriptions

Example: Representing an Image

Go row by row and look at pixel values



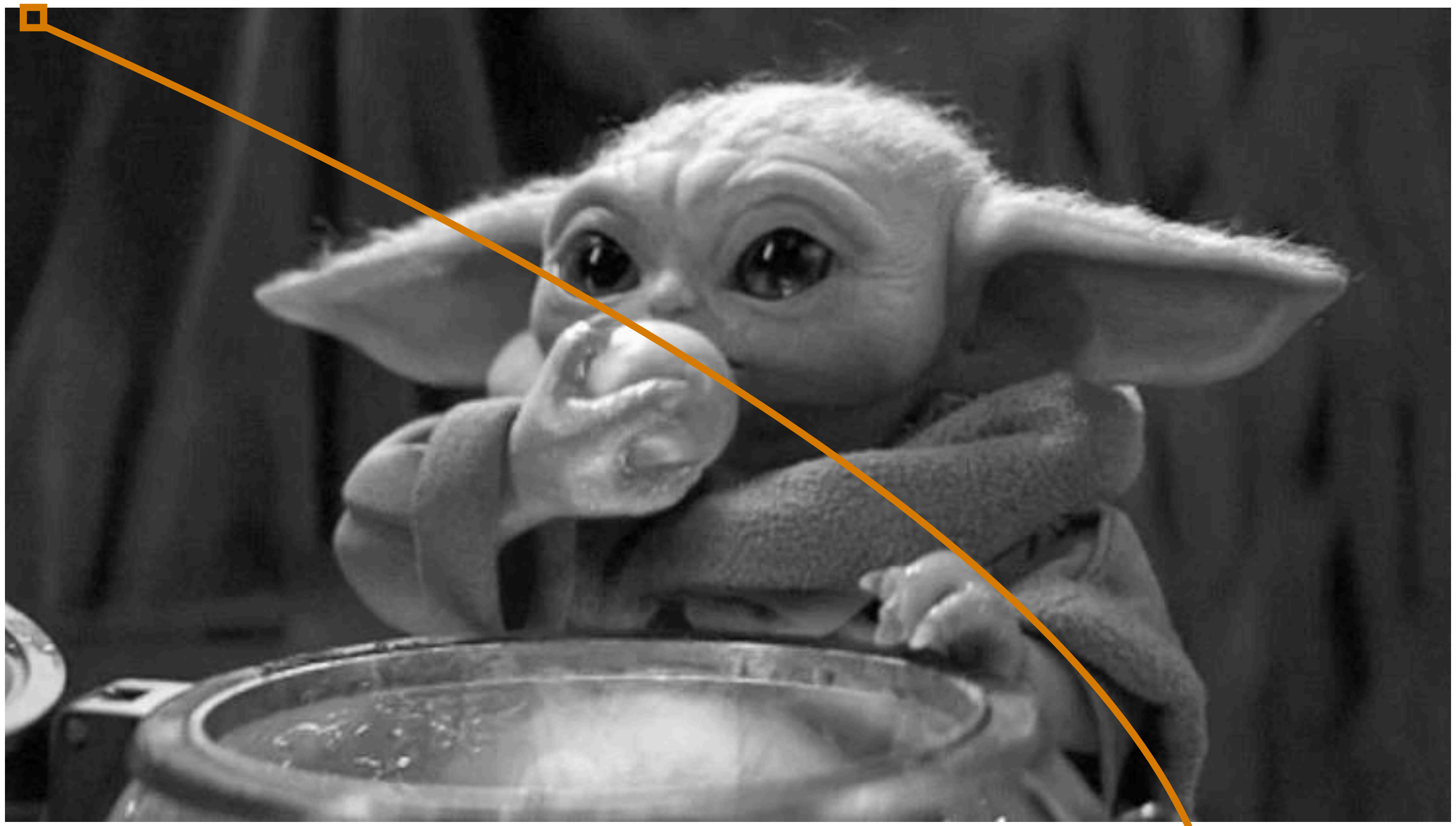
1: black
0: white

[1]

Image source: The Mandalorian

Example: Representing an Image

Go row by row and look at pixel values



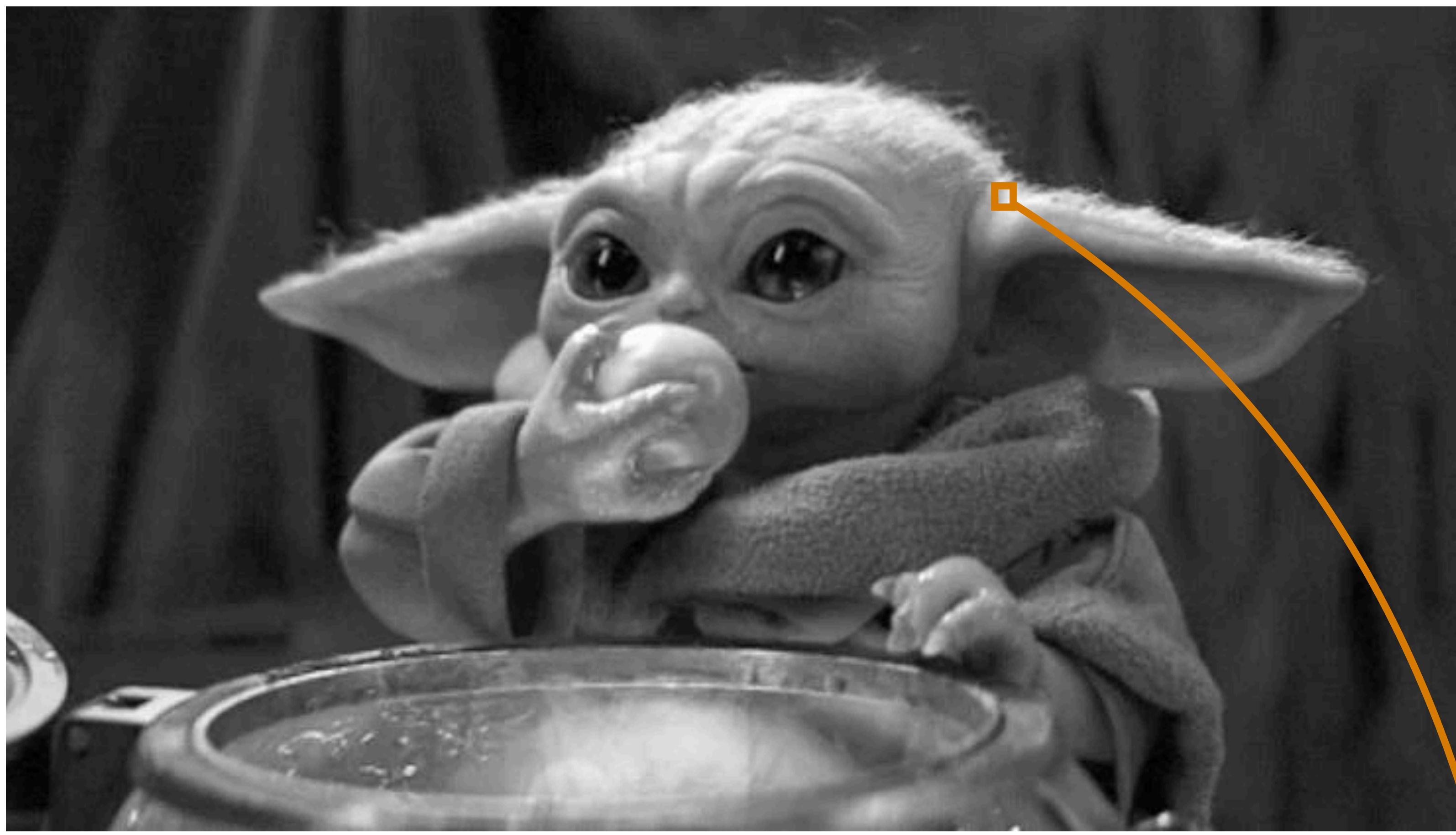
1: black
0: white

[1 0.9]

Image source: The Mandalorian

Example: Representing an Image

Go row by row and look at pixel values



1: black
0: white

$$[\ 1 \quad 0.9 \quad \cdots \quad 0.1 \qquad\qquad\qquad]$$

Image source: The Mandalorian

Example: Representing an Image

Go row by row and look at pixel values

1: black
0: white

$[\ 1 \quad 0.9 \quad \cdots \quad 0.1 \quad \cdots \quad 0.9\ ]$

dimensions = image height × image width

Image source: The Mandalorian

Very high dimensional!

Terminology Remark

$$[\ 1 \quad 0.9 \quad \cdots \quad 0.1 \quad \cdots \quad 0.9\ ]$$

⚠️ We use "dimension" to means two different things:

- number of axes we can index into for a table/array
  (e.g., 2D means there are rows & columns)

dimensions = 1

- total number of entries in the table/array

dimensions = image height × image width

Dimensionality Reduction for Images

Demo