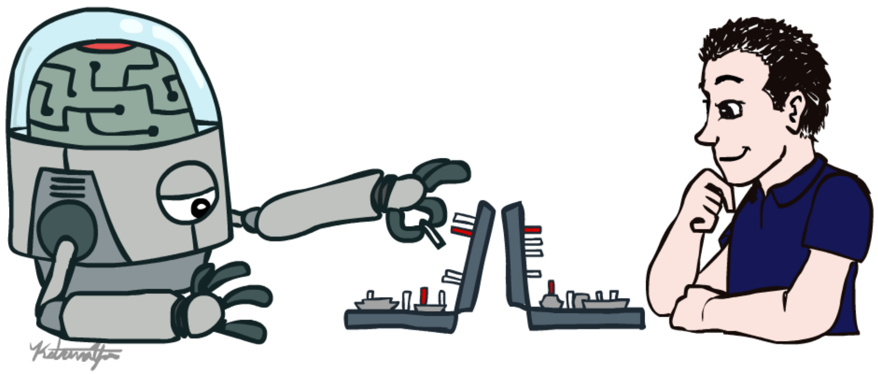
CSE 3521:
Introduction to Artificial Intelligence



[Many slides are adapted from the UC Berkeley. CS188 Intro to AI at UC Berkeley and previous CSE 3521 course at OSU.]

THE OHIO STATE UNIVERSITY

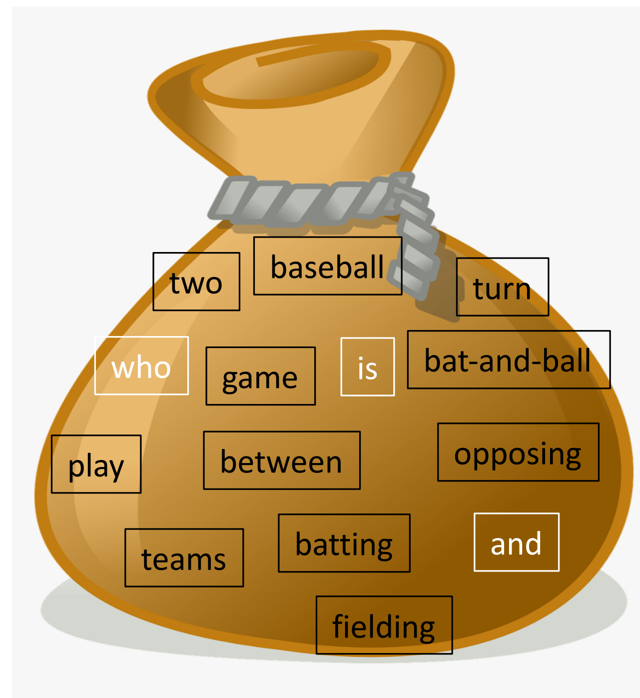
Bag of Words (BOW)

A simplified representation of sentences (documents) for classification or retrieval by counting the occurrences of each unique word (or phrase)

- ignore the grammar
- ignore the word order
- keep the word counts (frequencies)
- lead to a “fixed”-size vector representation

“Baseball” “is” “a” “bat-and-ball”
“game” “played” “between”
“two” “opposing” “teams”
“who” “take” “turns” “batting”
“and” “fielding” “.”

A sequence of tokens (and their indices)



Bag of Words (BOW)

Given:

- A dictionary, vocabulary, or codebook: f(token) → index ∈ $\{1, \cdots, D\}$ or N/A
- An all-0 vector: $\boldsymbol{x}$ = $\begin{bmatrix} x[1] \\ \vdots \\ x[D] \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$

Bag of Words (BOW)

Given:

- A dictionary, vocabulary, or codebook: f(token) → index ∈ $\{1, \cdots, D\}$ or N/A
- An all-0 vector: $\boldsymbol{x}$ = $\begin{bmatrix} x[1] \\ \vdots \\ x[D] \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$

Input: A sequence of word tokens w[1], w[2], ….. w[M]

for m = 1 : M

    if f(w[m]) != N/A

        $x[\mathrm{f(w[m])}]$ += 1

    end

end

Return: $\boldsymbol{x}$

Bag of Words (BOW)

Given:

- A dictionary, vocabulary, or codebook: f(token) → index ∈ $\{1, \cdots, D\}$ or N/A
- An all-0 vector: $\boldsymbol{x}$ = $\begin{bmatrix} x[1] \\ \vdots \\ x[D] \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$

Input: A sequence of word tokens w[1], w[2], ….. w[M]

for m = 1 : M

  if f(w[m]) != N/A

    x[f(w[m])] += 1

  end

end

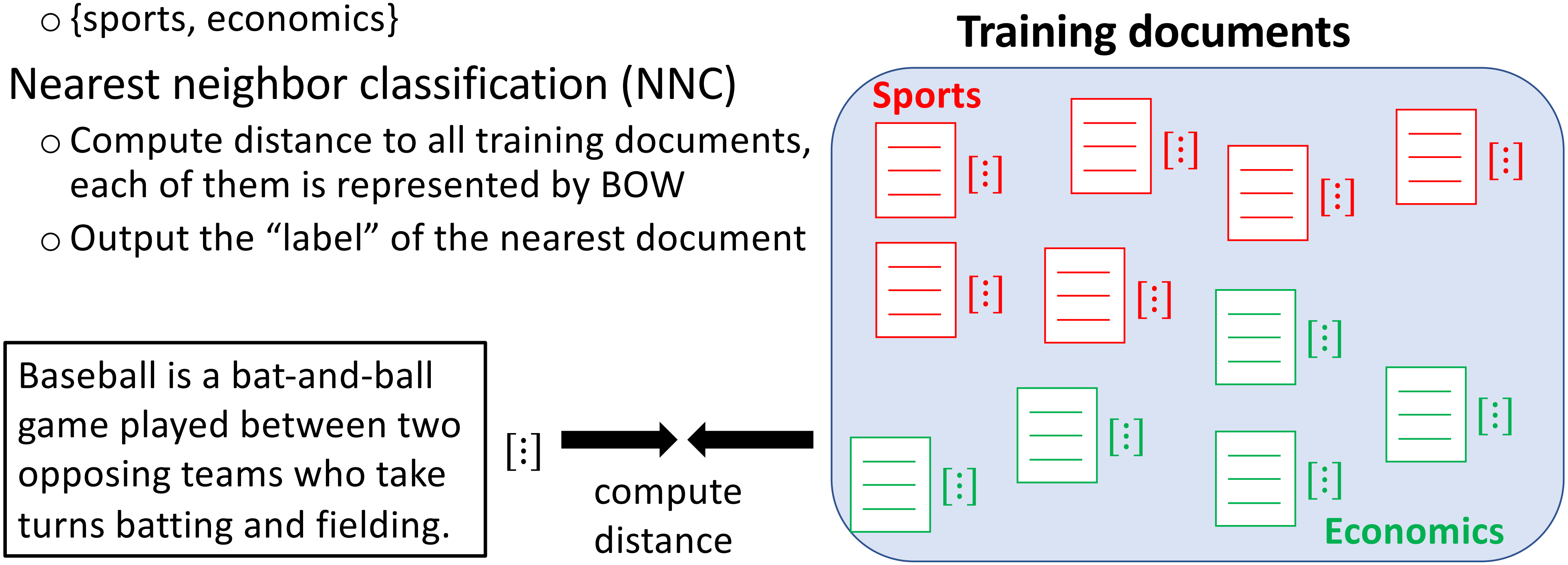
Return: $\boldsymbol{x}$

- **Out-of-vocabulary**
- **"Stop" words:** too frequent in all sentences/documents and less informative in differentiating them (e.g., "is", "are", "and", ……)

Word token counts

Example: BOW for classification

- Consider a document classification:
  - {sports, economics}
- Nearest neighbor classification (NNC)
  - Compute distance to all training documents, each of them is represented by BOW
  - Output the "label" of the nearest document



Example: BOW for classification

Distance

- Euclidean (L_2) distance:

$$\|\boldsymbol{x} - \boldsymbol{x_n}\|_2 = \left(\sum_{d=1}^{D} (x[d] - x_n[d])^2\right)^{1/2}$$

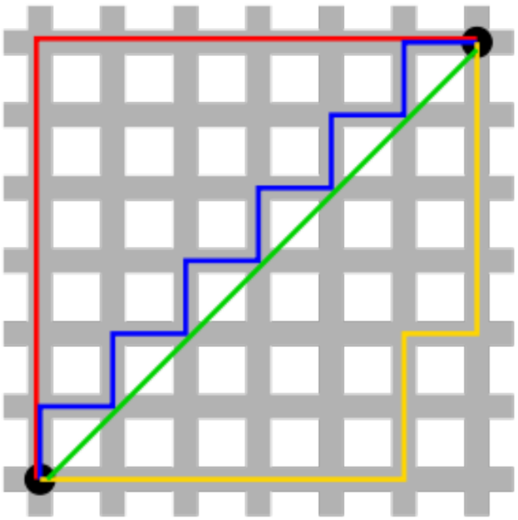
- L_1 distance:

$$\|\boldsymbol{x} - \boldsymbol{x_n}\|_1 = \sum_{d=1}^{D} |x[d] - x_n[d]|$$

- L_p norm:

$$\|\boldsymbol{x} - \boldsymbol{x_n}\|_p = \left(\sum_{d=1}^{D} |x[d] - x_n[d]|^p\right)^{1/p}$$

$\boldsymbol{x}$: 2-dimensional vectors



Green line is Euclidean distance. Red, Blue, and Yellow lines are L_1 distance

Example: BOW for classification

Given: a distance metric **dis**, a training set = $\{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_N, y_N)\}$, $t = \infty$ (min distance), y (predicted label)

Label:
$y \in$ {sports, economics}
$y \in \{-1, 1\}$ or $\{0, 1\}$
$y \in \{1, \dots, C\}$

Input: a test data instance $\boldsymbol{x}$

for n = 1 : N

  if dis($\boldsymbol{x}$, $\boldsymbol{x}_n$)$< t$

    $y = y_n$

    t = dis($\boldsymbol{x}$, $\boldsymbol{x}_n$)

  end

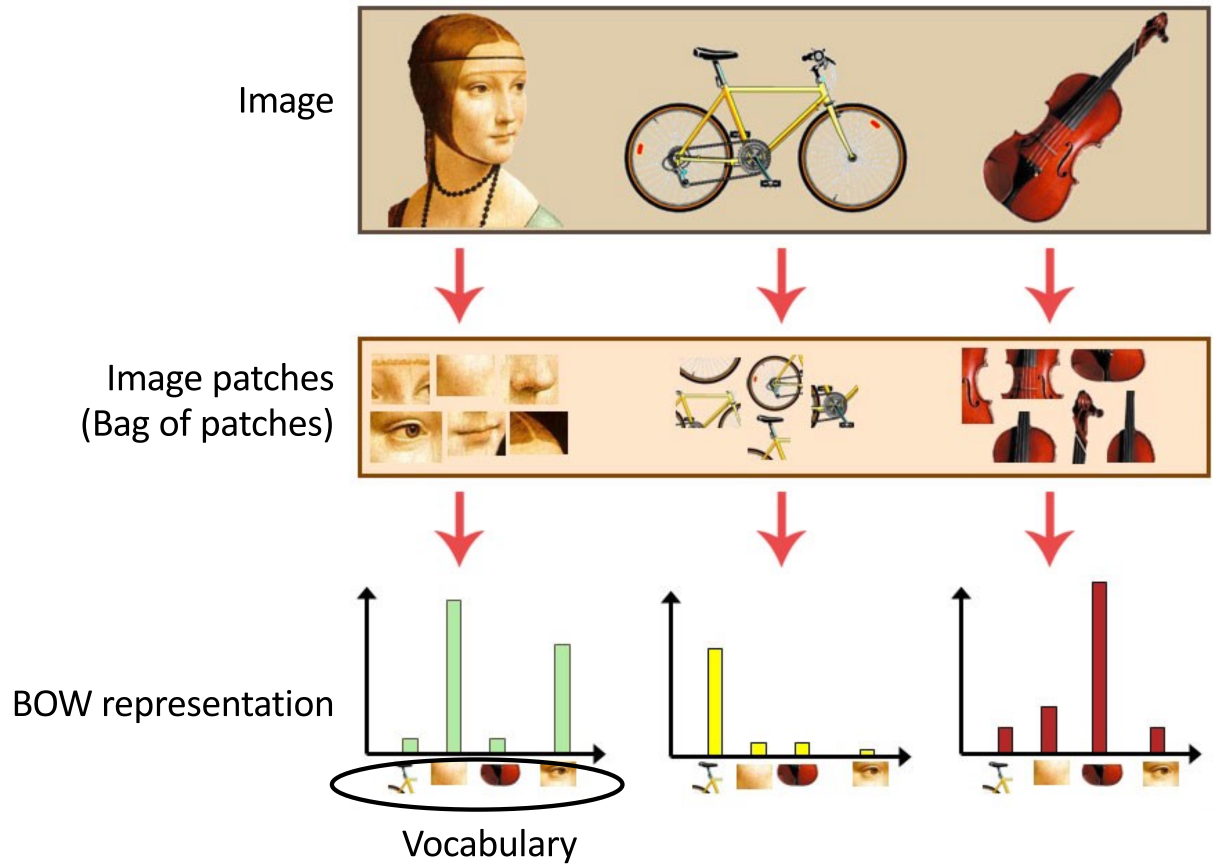
end

Return: y

Bag of Words (BOW) for images

BOW can be applied to vision

- Vocabulary: *D* image patches
- Each image: a bag of patches
- BOW representation: *D*-dim
  - For each image patch I[m], find the "nearest" patch in the vocabulary and use its index ∈ {1, …, D} to represent I[m]
  - Count the time that each index shows up in the image



Bag of Words (BOW)

Pros:

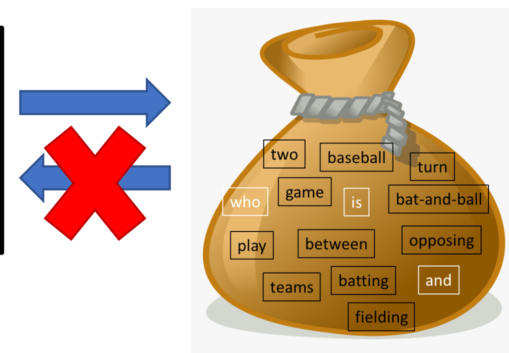
- Simplified, easily-understandable, fixed-size

What do we lose?

- What the original sequence was

Baseball is a bat-and-ball game played between two opposing teams who take turns batting and fielding.

x



Bag of Words (BOW)

Cons:

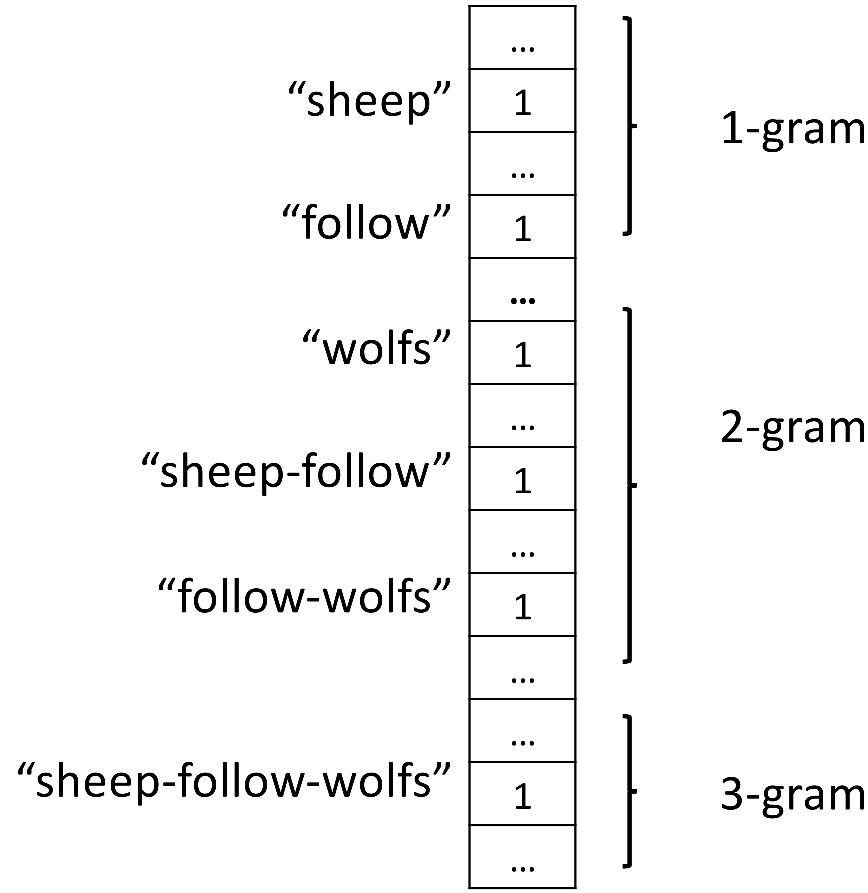
- missing the sequential information (e.g., “sheep follow wolfs” vs. “wolfs follow sheep”)
- not normalized (e.g., comparing long vs. short documents)
- words treated as independent (e.g., no synonyms and antonyms)
- highly sparse, high dimensional

N-gram vocabulary

Cons 1: Missing the sequential information

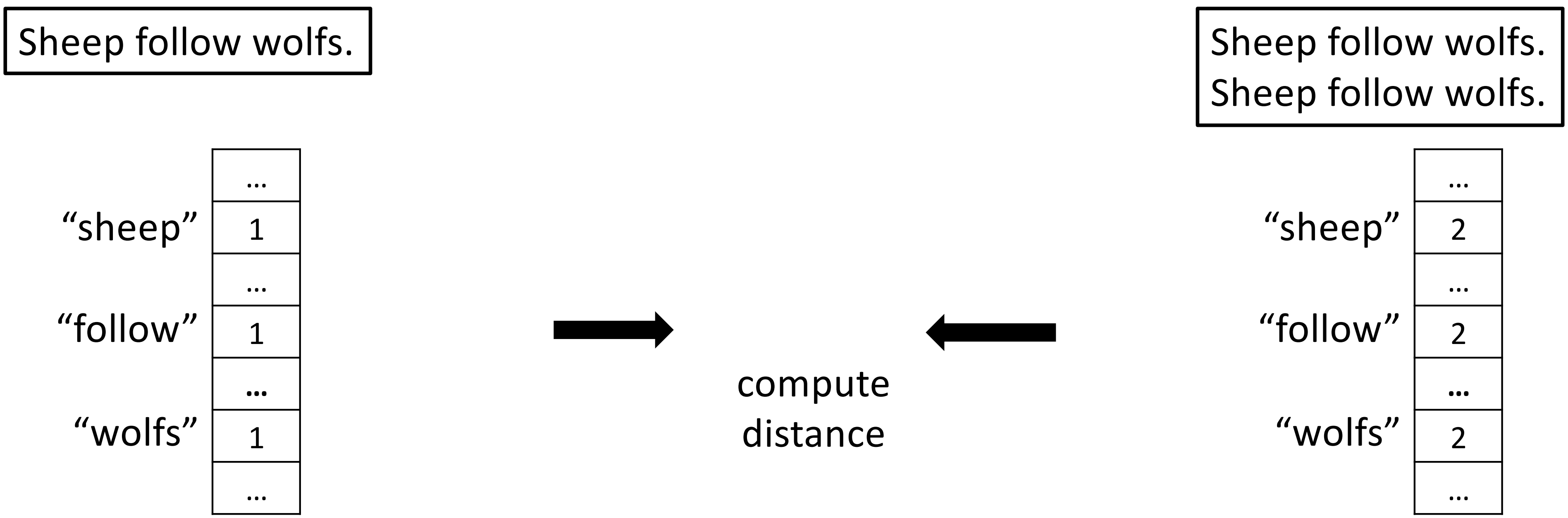
- One solution: **N-gram vocabulary**
- 1-gram (unigram): “sheep”, “follow”, “wolfs”
- 2-gram (bigram): “sheep-follow”, “follow-wolfs”
- 3-gram: “sheep-follow-wolfs”
- …
- Size of the vocabulary: $D + D^2 + D^3$

- “Sheep follow wolfs” becomes:



Dataset-independent normalization

Cons 2: not normalized



Dataset-independent normalization

Cons 2: not normalized

- One solution: **feature normalization**
- Vector norm (i.e., length or magnitude):

$$\|\boldsymbol{x}\|_p = \left(\sum_{d=1}^{D} |x[d]|^p\right)^{1/p}$$

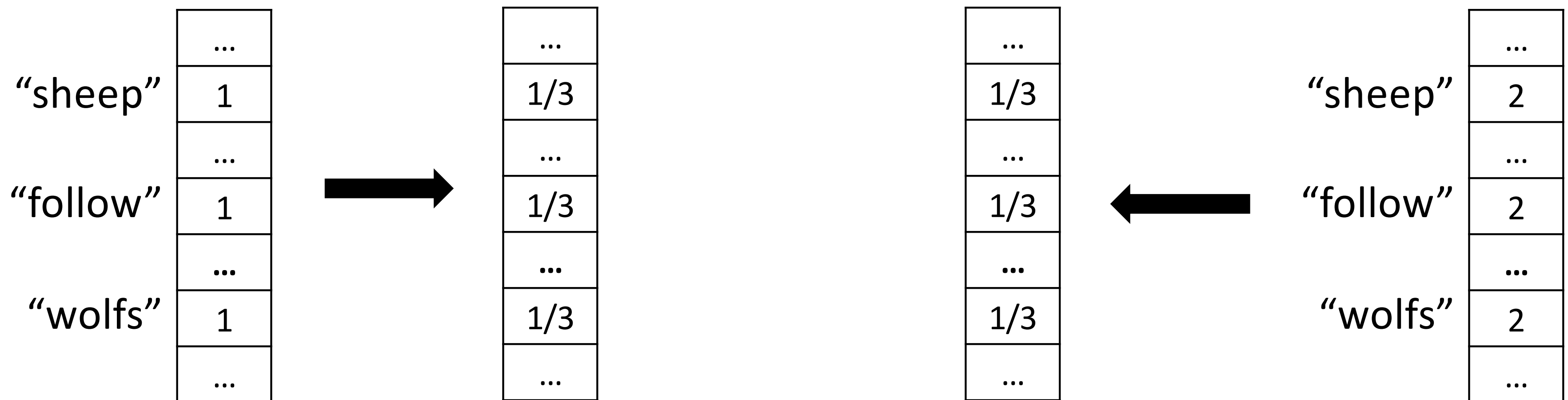
- **L_p normalization:**

$$\boldsymbol{z} = \frac{\boldsymbol{x}}{\|\boldsymbol{x}\|_p} = \begin{bmatrix} \frac{x[1]}{\|\boldsymbol{x}\|_p} \\ \vdots \\ \frac{x[D]}{\|\boldsymbol{x}\|_p} \end{bmatrix}$$

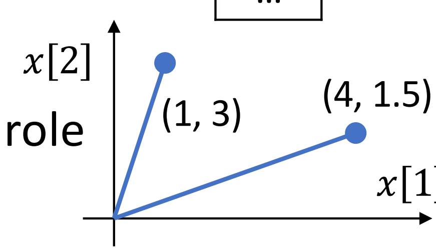
Dataset-independent normalization

Cons 2: not normalized

- One solution: **feature normalization**
- **L_1 normalization:** popular for counts (frequency, probability)



- **L_2 normalization:** widely used if the vector angle plays an important role



Bag of Words (BOW)

Cons:

- missing the sequential information (e.g., “sheep follow wolfs” vs. “wolfs follow sheep”)
- not normalized (e.g., comparing long vs. short documents)
- words treated as independent (e.g., no synonyms and antonyms)
- highly sparse, high dimensional

Summary

- Data and data representation (features)
  - Numerical vs. categorical variables
  - Feature extraction: from raw data to simplified, informative, non-redundant, or more interpretable representations
- Bag-of-words (BoW) representation
  - Fixed-sized representations for sentences and documents (and images)
  - Nearest neighbor classification based on distance metrics