L8: Building Embeddings : Feature Encodings

We want to analyze complex data … as vectors
Old view: build arrays of features

Two examples - document (text) encoding - image encoding

Document encoding

Document x: news articles (NYT), web pages, emails
strings of words

Bag of words

100,000 meaningful words in English
300,000 words ever in print … including emojis, slang, misspellings

$d = 100,000$

$f_j(x) = \#$ instances of jth word in doc x
$f(x) = (f_1(x), f_2(x), ..., f_d(x)) \in \mathbb{R}^d$
the **term frequency** vector

Two documents x_1, x_2 similar and related if $D_{\cos}(f(x_1), f(x_2))$ is small.

Will this work?

Most documents have very common words: "the", "be", "to", "of", "and", "a", "in"
may make up more than 10% of all words
and roughly the same in all documents

IDF : Inverse Document Frequency

Assume set of documents $X = \{x_1, x_2, ..., x_n\}$

$\text{IDF}_j = 1/(\#$ documents in X where $f_j(x_i) > 0)$
How important / interesting a word is (its specificity)

Similarity of two documents:

$$\text{TF-IDF}(x_1, x_2) = \sum_{j=1}^{d} \text{IDF}_j * (f_j(x_1) \cdot f_j(x_2))$$

term frequency - inverse document frequence

Search Engine

Query: $q_1, ..., q_m$ a small set of m keywords

Want documents x with high similarity score to query Q
$Q = [q_1 q_2 ... q_m]$
Sim$(Q,x) = $ TF-IDF(Q, x)
$= \sum_{j=1}^{m} \text{IDF}(q_j) * (1 \cdot f_{q_j}(x))$

Okapi BM25

Enormously widely used score for document retrieval.
[Robertson + Jones 1970s, 80s –> Okapi IR system at UCL]
Still the best method under some evaluations

First, redefine IDF:

$$\text{IDF}_{25}(q_j) = \ln\left(\frac{|X| - n(q_j) + 0.5}{n(q_j) + 0.5} + 1\right)$$

$n(q_j)$ = frequency of work q_j
avoids divide by 0

With $|X| = 100$

n	IDF	IDF-25
1	1.0	4.21
2	0.5	3.70
3	0.33	3.36
4	0.25	3.11
5	0.2	2.9
10	0.1	2.26
20	0.05	1.59

Set parameters: $k = 1.5$ and $b = 0.75$
Let $|x|$ be the length of document x, and $A = \frac{1}{|X|}\sum_{x \in X} |x|$ is the average length

$$\text{BM25}(Q, x) = \sum_{j=1}^{m} \text{IDF}(q_j) \frac{f_{q_j}(x)(k+1)}{f_{q_j}(x) + k(1 - b + b\frac{|x|}{A})}$$

Image Encoding

Image is a grid ($d = g_1 \times g_2$) of pixels
Each pixel typically has 3 scalar values (red, green, blue)
For today, pretend black+white, so pixel has single scalar value

image -> vector in $\mathbb{R}^d$.
Not useful for much other than color matching

circa 2000: Computer vision used edge detectors and convolutions-based smoothing
edge detectors identify ridges of high gradient
endpoints of edges –> corners were key features: - edge of mouth - corner of eye - tip of nose

Given a feature, how similar are they?

SIFT : Scale Invariant Feature Transform

David Lowe 1999 [had lots of trouble publishing!]

1. Use edge detector (via convolutions) to

- identify features
- the scale of features (how large a neighborhood defines it)

2. Determine orientation

- direction with largest gradient magnitude
- check every 10 degrees (36 options)

3. Feature descriptor

- 16x16 pixel window around feature point

- 16 cells, each 4x4 pixels
- in cell, build 8-dimensional histogram of gradients
- subtract angle from feature orientation (from 2)
- $16 \times 8 = 128$

Each feature a ($d = 128$)-dimensional vector
Recommend similarity by Euclidean distance
Was state of art for over 10 years!