

Machine Learning

Different Data Formats

Edgar F. Roman-Rangel.
`edgar.roman@itam.mx`

Digital Systems Department.
Instituto Tecnológico Autónomo de México, ITAM.

May 28th, 2021.

Outline

Data formats

Text

Images

Intro

So far:

- ▶ Supervised and non-supervised ML.
- ▶ Classification, regression, clustering, dimensionality reduction.
- ▶ Always assuming we already got numeric data: vectors.

What about other types of data? E.g.,

- ▶ Text,
- ▶ Images or video,
- ▶ Audio,
- ▶ Radio frequencies,
- ▶ Etc.

Data must be converted into a numeric descriptor, i.e., vector.

Different formats

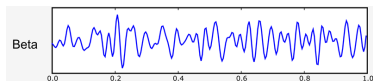
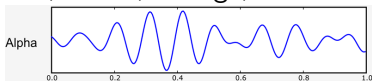
Most types of data can be understood as either:

Static data

Vectors, as we already know.

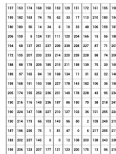
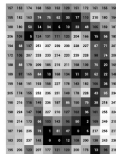
Sequential data

Text, sound, voltage, etc.



Spatial data

Images.



Or a combination of both, e.g., video.

Exploit other formats

Design filters that extract relevant statistics, create vectors.

Outline

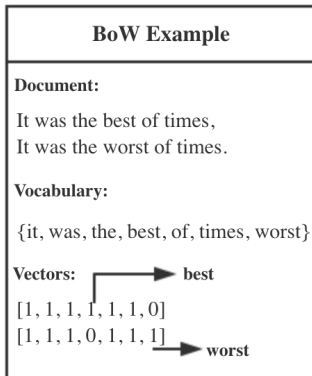
Data formats

Text

Images

BoW

Bag-of-words (BoW): vector that counts frequency of words.



Idea: documents of similar topic, have similar word distribution.

Considerations

- ▶ Put all characters in lowercase.
- ▶ Remove punctuation and special characters.
- ▶ Remove numbers.
- ▶ Remove stopwords (articles, prepositions, etc).
- ▶ Use lematization or stemming.

TD-IDF

Term frequency - inverse document frequency (tf-idf): used for weighting each term with a inverse frequency with respect to documents: terms appearing in all documents are of low relevance.

$$w_{x,y} = \text{tf}_{x,y} \times \log \left(\frac{N}{\text{df}_x} \right)$$

TF-IDF

Term x within document y

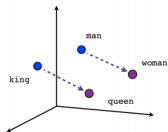
$\text{tf}_{x,y}$ = frequency of x in y

df_x = number of documents containing x

N = total number of documents

Embeddings

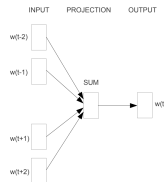
Starting from one-hot encoding vectors, find rich dense representation that capture semantic context of words.



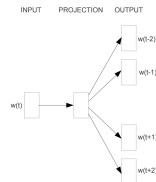
Male-Female



Verb tense



CBOW



Skip-gram

Outline

Data formats

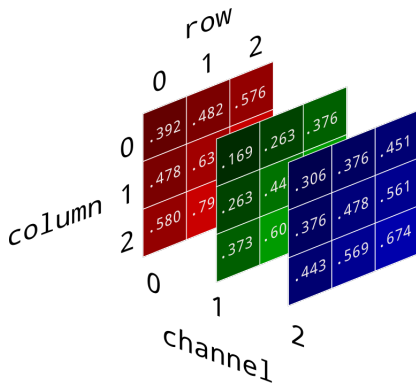
Text

Images

Image formation

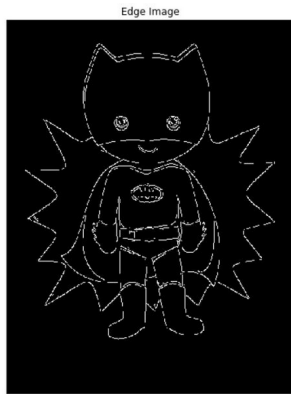
Each pixel indicates a relative amount of light captured by a sensor.

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	93	17	110	210	180	154
180	180	50	14	94	6	10	93	48	105	159	181
206	100	5	124	131	111	120	204	165	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	105	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	38	218	241
190	234	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	90	2	109	249	215
187	195	235	73	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	209	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218










Derivatives in 2D

Edge detector:



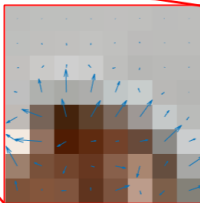
Convolution

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1																									
2																									
3																									
4																									
5																									
6																									
7																									
8																									
9																									
10																									
11																									
12																									
13																									
14																									
15																									
16																									
17																									
18																									
19																									
20																									
21																									

Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur 3 × 3 (approximation)	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

HOG

Histogram-of-Oriented-Gradients.



2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

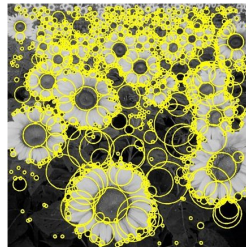
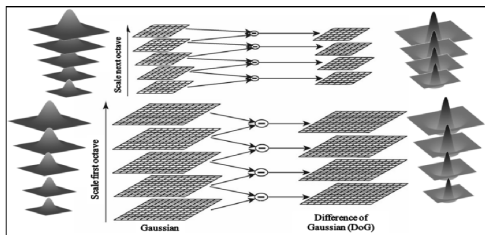
Gradient Magnitude

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

Local image descriptor

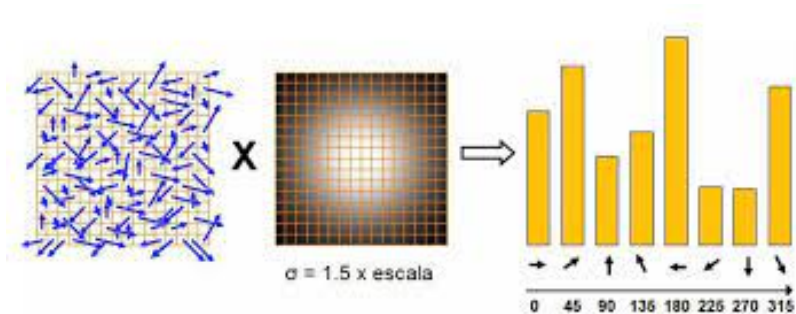
Detect points of interest (PoI): corners of blobs.



DoG images shown on the left, Difference of Gaussian images on the right

SIFT

Scale-Invariant Feature Transform



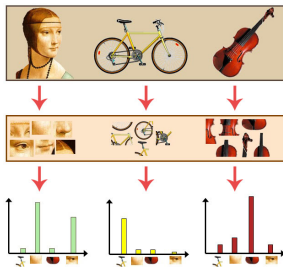
Each local descriptor is a 128-D vector. There are as many local descriptors as Pol's were detected.

BoVW

Count the frequency of *visual words* (types of local descriptors).

Descriptors are vectors in \mathbb{R}^N , let's map them to \mathbb{Z} .

1. Grab a set of local descriptors.
2. Use a clustering algorithm to group them in D clusters.
3. Label each descriptor with the index of its cluster.
4. Create a D -dimensional vector of visual words.



Q&A

Thank you!

`edgar.roman@itam.mx`