

Introduction to image analysis

Michelle Torres
Rice University

July 19, 2021

IMAGE AS DATA: MOTIVATION

- Images are **powerful**: extra information + emotional activation + *see to believe* = recall and engagement
- Images are **(kind of)** universal (e.g. compare them to spoken languages)
- Visuals are **frames**
- New possibilities for measurement: ethnic visibility, natural disaster prevention and evaluation, state capacity, data collection, etc.
- **Computer Vision:**
 - Teaching computers to see
 - Process large pools of images
 - Increase consistency/reliability and decrease bias (*)
 - Helping humans to “see” and discover (*)

IMAGE AS DATA: TEACHING THE COMPUTER TO SEE

IMAGE AS DATA: TEACHING THE COMPUTER TO SEE

A very hard task!

IMAGE AS DATA: TEACHING THE COMPUTER TO SEE

A very hard task!

- Computers are great at following instructions reliably...



IMAGE AS DATA: TEACHING THE COMPUTER TO SEE

A very hard task!

- Computers are great at following instructions reliably...
- ... but they are bad at inferences



IMAGE AS DATA: TEACHING THE COMPUTER TO SEE

A very hard task!

- Computers are great at following instructions reliably...
- ... but they are bad at inferences



IMAGE AS DATA: TEACHING THE COMPUTER TO SEE

A very hard task!

- Computers are great at following instructions reliably...
- ... but they are bad at inferences



IMAGE AS DATA: TEACHING THE COMPUTER TO SEE

A very hard task!

- Computers are great at following instructions reliably...
- ... but they are bad at inferences



OVERVIEW

① Image Basics

- Pixels, color channels, and more
- Software and computational requirements

② Extracting basic features

- Color statistics

③ The Bag of Visual Words

④ Convolutional Neural Networks

- Overview
- Structure and implementation
- Application

⑤ Miscellaneous

- GoogleVision API
- Validation and diagnosis
- Warnings: scope, biases and unintended problems

⑥ See them in action!

GETTING READY

- Workshop material available at my GitHub page, and official website
- Google Colab notebook:
<https://colab.research.google.com/drive/1KFHwz8wjDdcFfsTmXfo-gwkKc-itN3MS?usp=sharing>
- When doing your own projects:
 - Install Keras (<https://keras.io/#installation>)
 - Install the following python libraries: numpy, scipy, cv2, matplotlib, PIL, sklearn
 - Check tutorials for OpenCV installation:
<https://www.pyimagesearch.com/opencv-tutorials-resources-guides/>
 - I suggest OpenCV 3.X and its compilation from source for full functionality

IMAGE BASICS

- An image is a set of **pixels**:

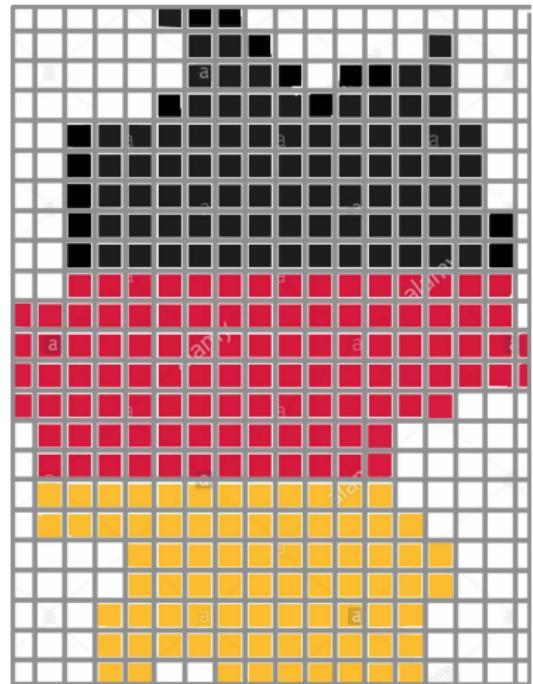


IMAGE BASICS

- An image is a set of **pixels**:
 - Finest unit (defines **height** and **width**)

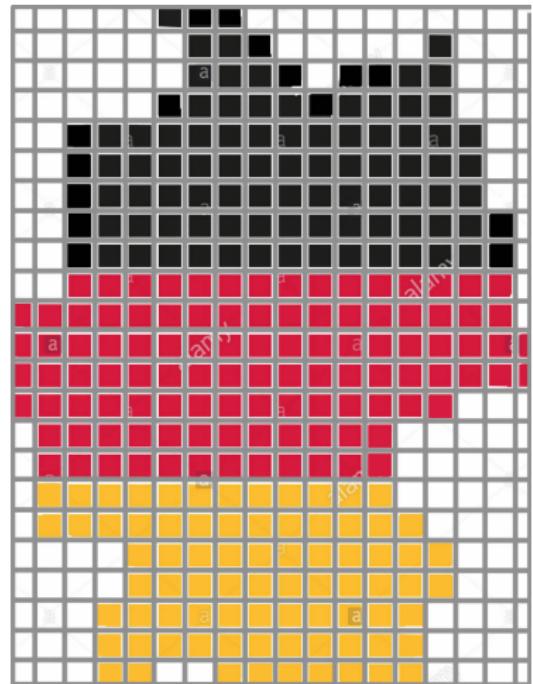


IMAGE BASICS

- An image is a set of **pixels**:
 - Finest unit (defines **height** and **width**)
 - Grayscale: intensity of light, **Color**: color intensity per channel.

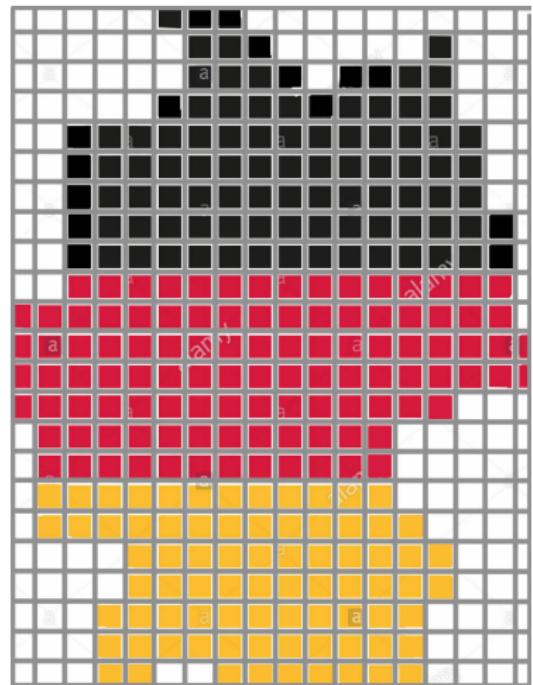


IMAGE BASICS

- An image is a set of **pixels**:
 - Finest unit (defines **height** and **width**)
 - Grayscale: intensity of light, **Color**: color intensity per channel.
- Matrix representation

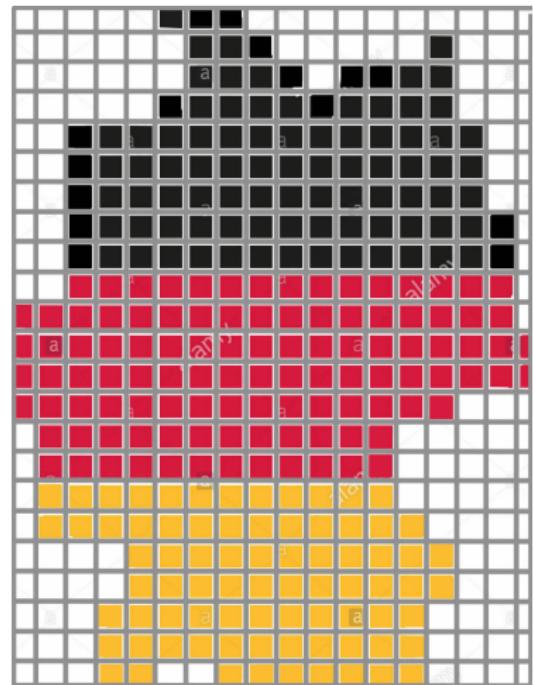


IMAGE BASICS

- An image is a set of **pixels**:
 - Finest unit (defines **height** and **width**)
 - Grayscale: intensity of light, **Color**: color intensity per channel.
- Matrix representation
 - Grayscale: one matrix

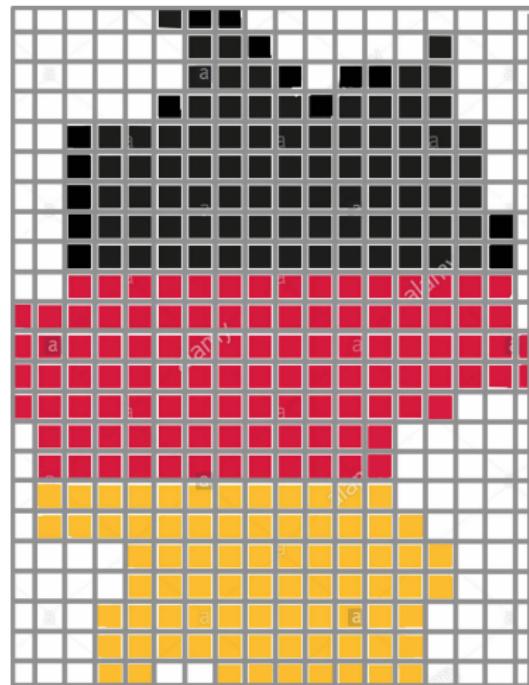


IMAGE BASICS

- An image is a set of **pixels**:
 - Finest unit (defines **height** and **width**)
 - Grayscale: intensity of light, **Color**: color intensity per channel.
- Matrix representation
 - Grayscale: one matrix
 - Color: array with a matrix for each color channel (**Red**, **Green**, and **Blue**)

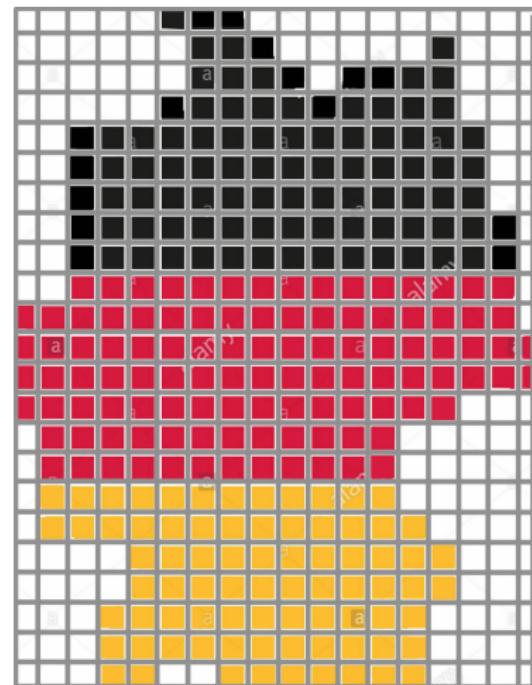


IMAGE BASICS

- An image is a set of **pixels**:
 - Finest unit (defines **height** and **width**)
 - Grayscale: intensity of light, **Color**: color intensity per channel.
- Matrix representation
 - Grayscale: one matrix
 - Color: array with a matrix for each color channel (**Red**, **Green**, and **Blue**)
- Notice that in OpenCV:

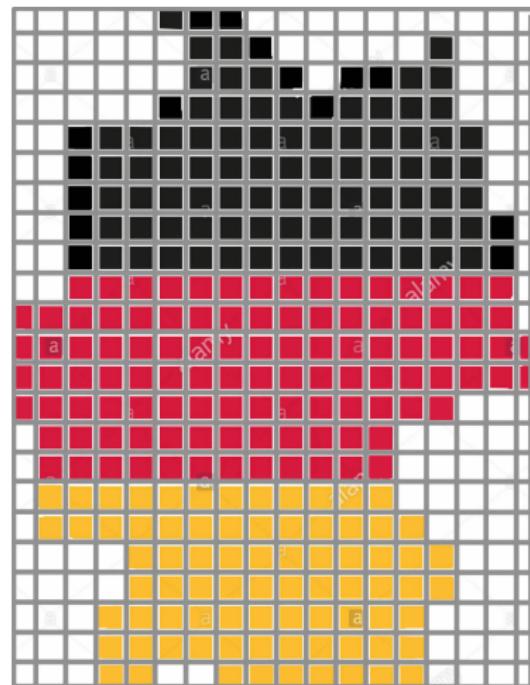


IMAGE BASICS

- An image is a set of **pixels**:
 - Finest unit (defines **height** and **width**)
 - Grayscale: intensity of light, **Color**: color intensity per channel.
- Matrix representation
 - Grayscale: one matrix
 - Color: array with a matrix for each color channel (**Red**, **Green**, and **Blue**)
- Notice that in OpenCV:
 - Color channel specification is **BRG** instead of **RGB**

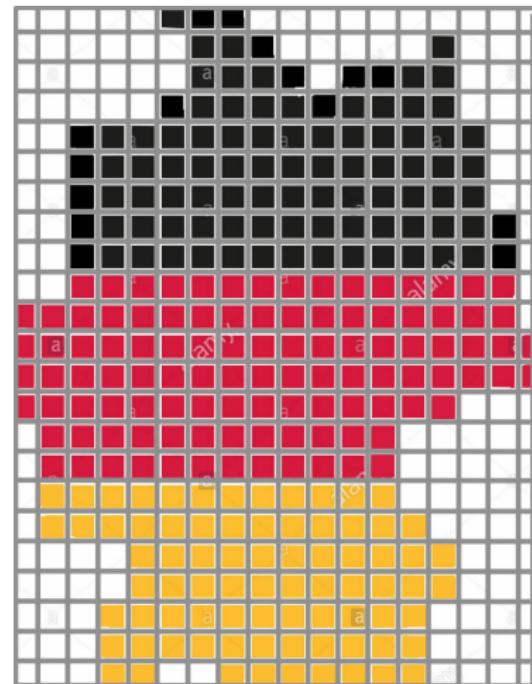


IMAGE BASICS

- An image is a set of **pixels**:
 - Finest unit (defines **height** and **width**)
 - Grayscale: intensity of light, **Color**: color intensity per channel.
- Matrix representation
 - Grayscale: one matrix
 - Color: array with a matrix for each color channel (**Red**, **Green**, and **Blue**)
- Notice that in OpenCV:
 - Color channel specification is BRG instead of RGB
 - Origin of image is different (top left corner)

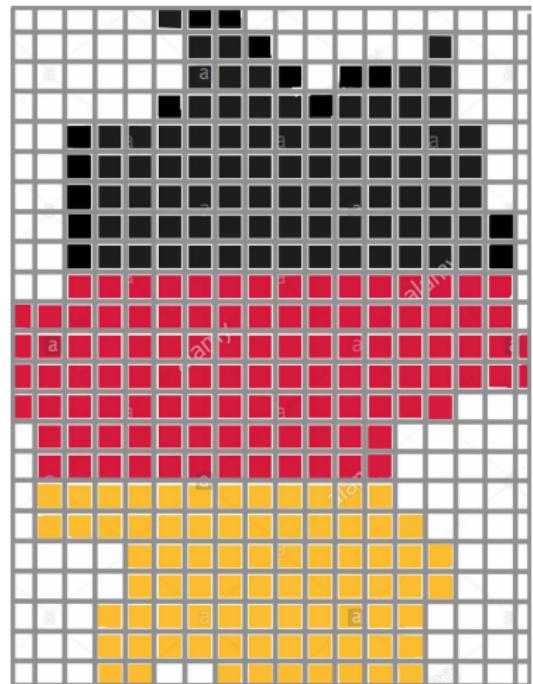
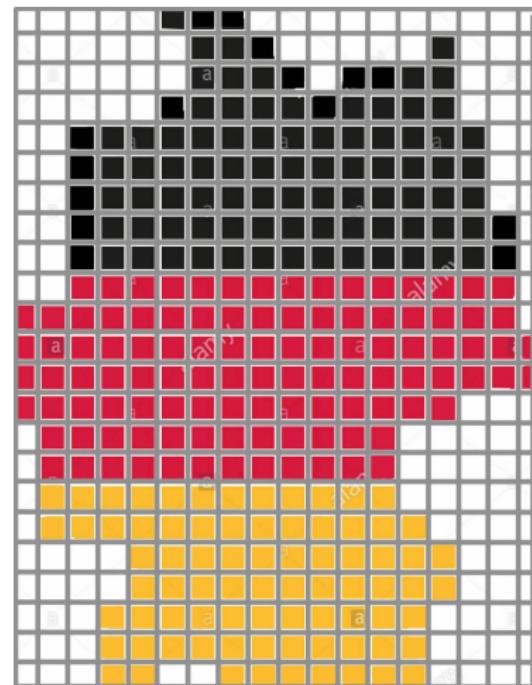


IMAGE BASICS

- An image is a set of **pixels**:
 - Finest unit (defines **height** and **width**)
 - Grayscale: intensity of light, **Color**: color intensity per channel.
- Matrix representation
 - Grayscale: one matrix
 - Color: array with a matrix for each color channel (**Red**, **Green**, and **Blue**)
- Notice that in OpenCV:
 - Color channel specification is BRG instead of RGB
 - Origin of image is different (top left corner)
 - In numpy you specify the y -coordinates of an image first: $x2 = \text{image}[y0:y1, x0:x1]$



Color descriptors

DESCRIBING AN IMAGE

- A **challenge**: a lot of pixels that mostly make sense when analyzed in clusters and not as units.
- Therefore, we use **image descriptors** to characterize the content on an image ***globally*** or **feature descriptors** to locally quantify ***regions*** of the image.
 - Color
 - Texture
 - Shape
- Feature vectors: A series of numbers used to numerically quantify the contents of an image (or regions of it)

COLOR STATISTICS

Channel statistics

- Very intuitive and simple
- Basic statistics of each color channel
 - ① Separate channels
 - ② Compute moments for each channel

④ Concatenate to form *feature vector*

⑤ Compute vector similarity

Histograms

- More information based on distribution
- 3D histogram of colors
 - ① Convert image to L*a*b color space
 - ② Compute 3D histogram

COLOR STATISTICS

Channel statistics

- Very intuitive and simple
- Basic statistics of each color channel
 - ① Separate channels
 - ② Compute moments for each channel

Histograms

- More information based on distribution
- 3D histogram of colors
 - ① Convert image to L*a*b color space
 - ② Compute 3D histogram

④ Concatenate to form *feature vector*

⑤ Compute vector similarity

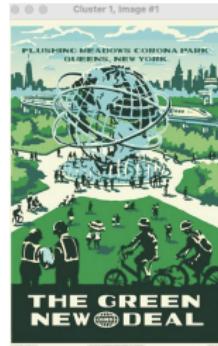
$$d(p, q) = \sqrt{\sum_{i=1}^N (q_i - p_i)^2}$$

COLOR STATISTICS, CONT.



OR RUN K-MEANS!

Cluster 1 →



Cluster 2 →



The Bag of Visual Words

THE BoVW

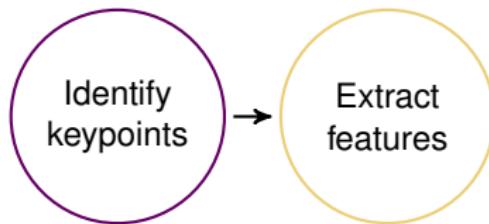
- Emulate a Document-Term matrix
- No words *but patches*

Identify
keypoints



THE BoVW

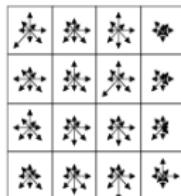
- Emulate a Document-Term matrix
- No words *but patches*



*	*	*	*
*	*	*	*
*	*	*	*
*	*	*	*

THE BOVW

- Emulate a Document-Term matrix
- No words *but patches*



THE BOVW

- Emulate a Document-Term matrix
- No words *but patches*

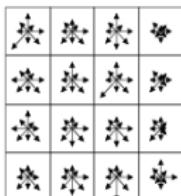
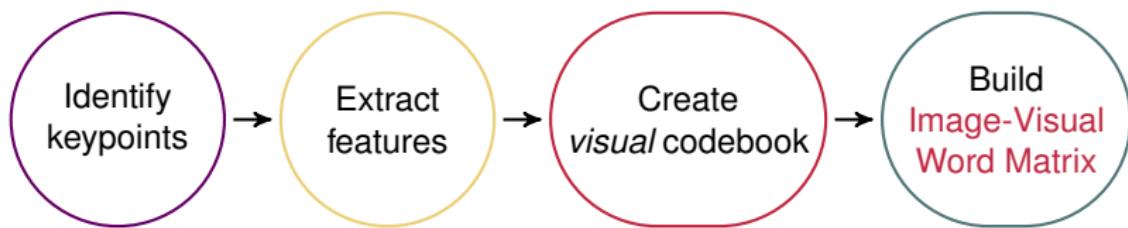


Image	Image 1	Image 2	...	Image n
Image 1	0	1	...	0
Image 2	0	1	...	1
Image n	5	8	...	4

KEY POINT DETECTION

- Identify salient regions, edges and corners based on pixel intensity changes
- FAST Hessian: fast and scale-invariant (Bay et al. 2006)



KEY POINT DETECTION: HOW TO IDENTIFY THEM?

- FAST Hessian: fast and scale-invariant (Bay et al. 2006)
 - ① Scale (smooth) images to create *octaves*
 - Convolution with an approximation of a Gaussian second-order derivatives filter

KEY POINT DETECTION: HOW TO IDENTIFY THEM?

- FAST Hessian: fast and scale-invariant (Bay et al. 2006)
 - ➊ Scale (smooth) images to create *octaves*
 - Convolution with an approximation of a Gaussian second-order derivatives filter

$$L_{xx} = I(\mathbf{x}) \otimes \frac{\partial^2}{\partial x^2} g(\sigma) \quad L_{yy} = I(\mathbf{x}) \otimes \frac{\partial^2}{\partial y^2} g(\sigma) \quad L_{xy} = I(\mathbf{x}) \otimes \frac{\partial^2}{\partial x \partial y} g(\sigma)$$

$$\mathcal{H}(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix}$$

KEY POINT DETECTION: HOW TO IDENTIFY THEM?

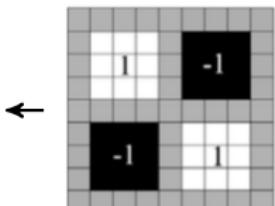
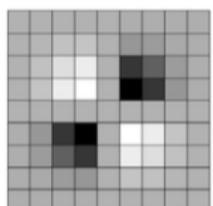
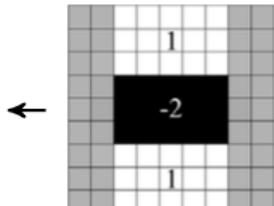
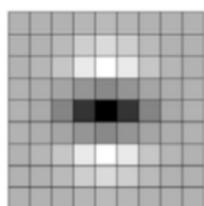
- FAST Hessian: fast and scale-invariant (Bay et al. 2006)

① Scale (smooth) images to create *octaves*

- Convolution with an approximation of a Gaussian second-order derivatives filter

$$L_{xx} = I(\mathbf{x}) \otimes \frac{\partial^2}{\partial x^2} g(\sigma) \quad L_{yy} = I(\mathbf{x}) \otimes \frac{\partial^2}{\partial y^2} g(\sigma) \quad L_{xy} = I(\mathbf{x}) \otimes \frac{\partial^2}{\partial x \partial y} g(\sigma)$$

$$\mathcal{H}(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix}$$



KEY POINT DETECTION: HOW TO IDENTIFY THEM?

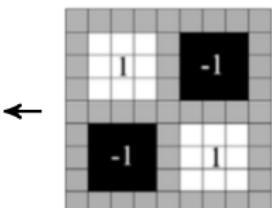
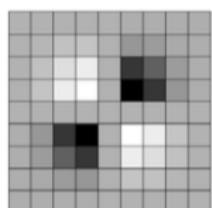
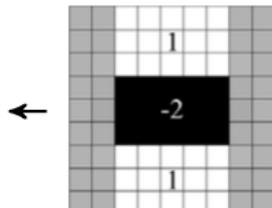
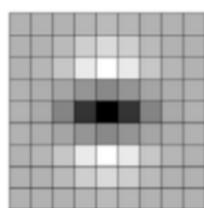
- FAST Hessian: fast and scale-invariant (Bay et al. 2006)

- ① Scale (smooth) images to create *octaves*

- Convolution with an approximation of a Gaussian second-order derivatives filter

$$L_{xx} = I(\mathbf{x}) \otimes \frac{\partial^2}{\partial x^2} g(\sigma) \quad L_{yy} = I(\mathbf{x}) \otimes \frac{\partial^2}{\partial y^2} g(\sigma) \quad L_{xy} = I(\mathbf{x}) \otimes \frac{\partial^2}{\partial x \partial y} g(\sigma)$$

$$\mathcal{H}(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix}$$

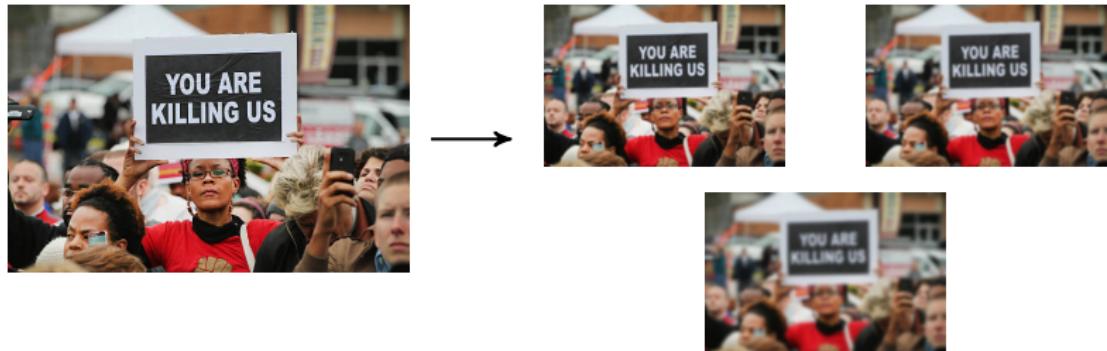


$$\text{Approximation} \Rightarrow \det(\mathcal{H}_{approx}) = D_{xx}D_{yy} - (0.9D_{xy})^2$$

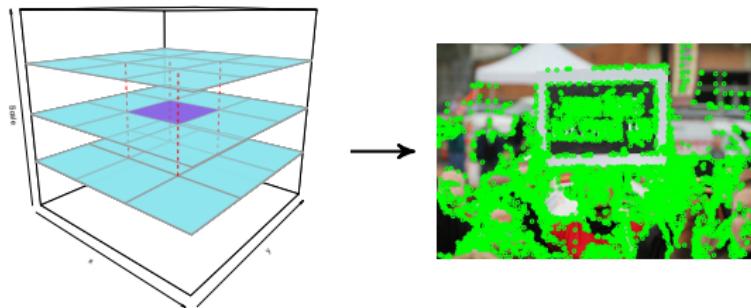
KEY POINT DETECTION: HOW TO IDENTIFY THEM?



KEY POINT DETECTION: HOW TO IDENTIFY THEM?



- ② Compare pixels to their $3 \times 3 \times 3$ neighborhood



FEATURE EXTRACTION

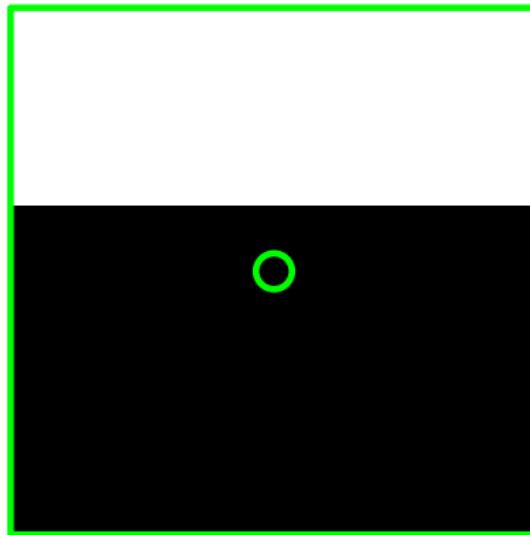
- ① Gaussian blurred image (A)
- ② 16×16 neighborhood around each key point, broken into 4×4 cells
- ③ 8-bin histogram of gradients' orientations weighted by magnitude, per cell
 - $G_x = A(x, y) - A(x + 1, y)$, $G_y = A(x, y) - A(x, y + 1)$
 - $M_{x,y} = \sqrt{G_x^2 + G_y^2}$
 - $\theta_{x,y} = \text{arctan2}(G_y, G_x) \times \left(\frac{180}{\pi}\right)$
- ④ Square root and normalization

Feature vector

128-dimensional vector: 8-bins per histogram \times 16 cells

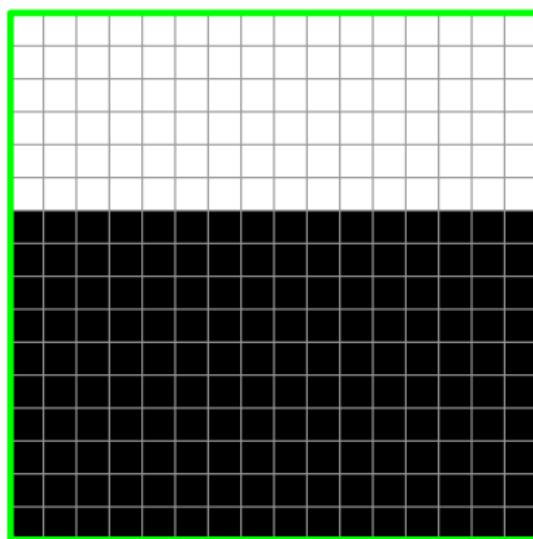
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Take a key point



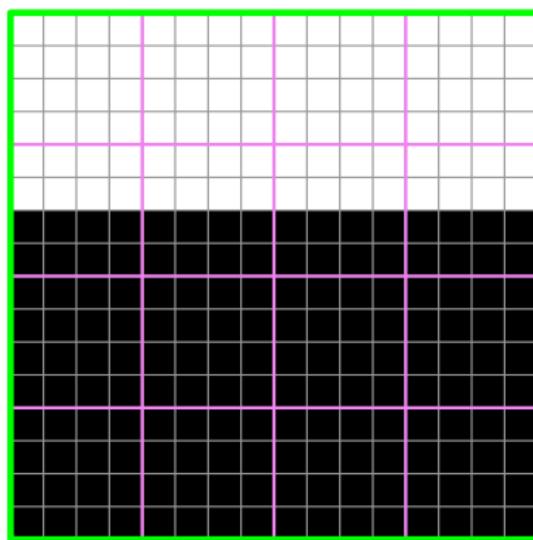
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Draw a 16×16 pixel neighborhood



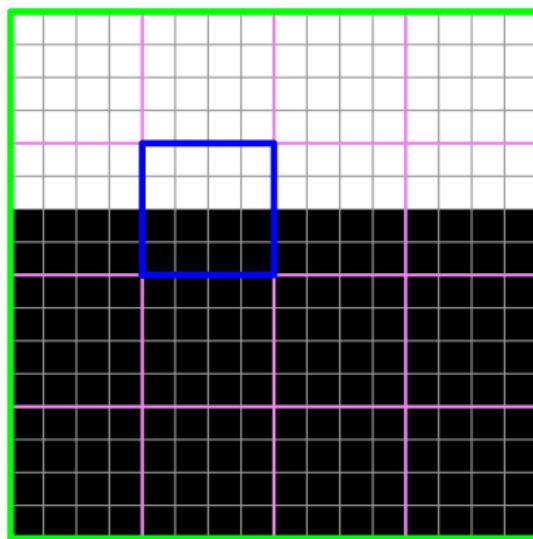
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Divide in 4×4 pixel cells



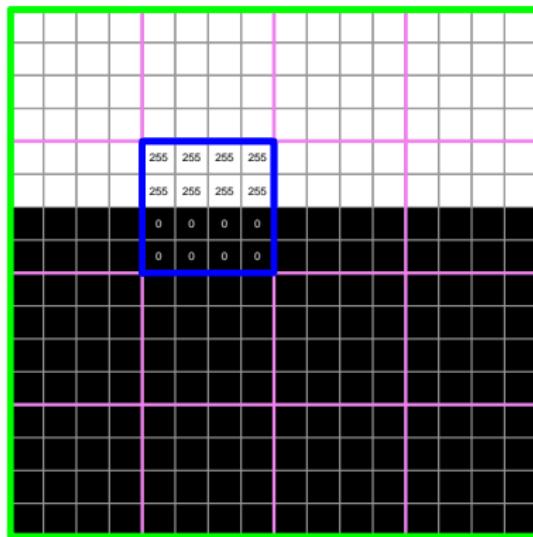
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Let's focus on one cell...



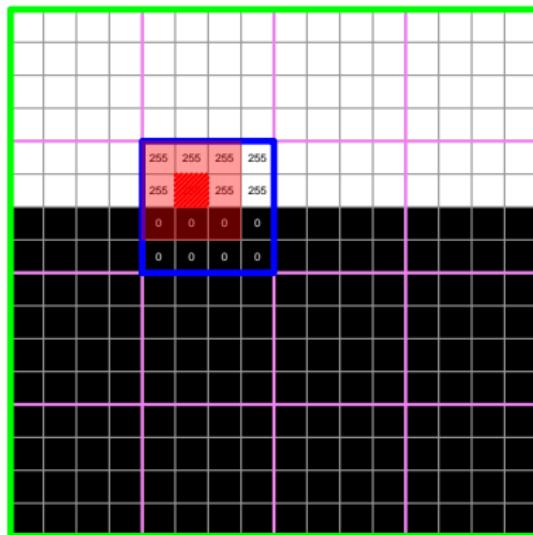
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Let's focus on one cell...



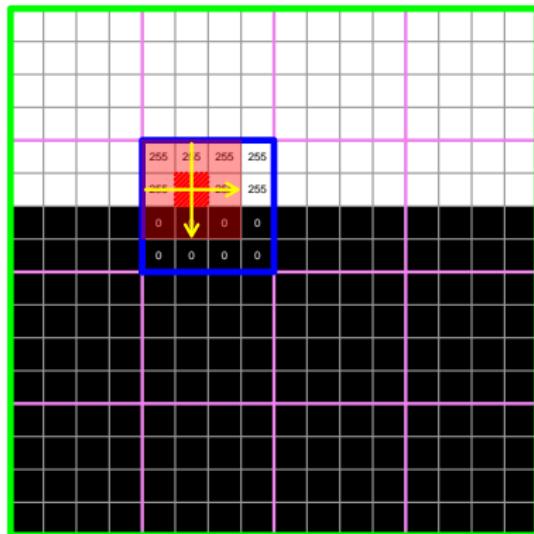
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Estimate gradient orientation and magnitude



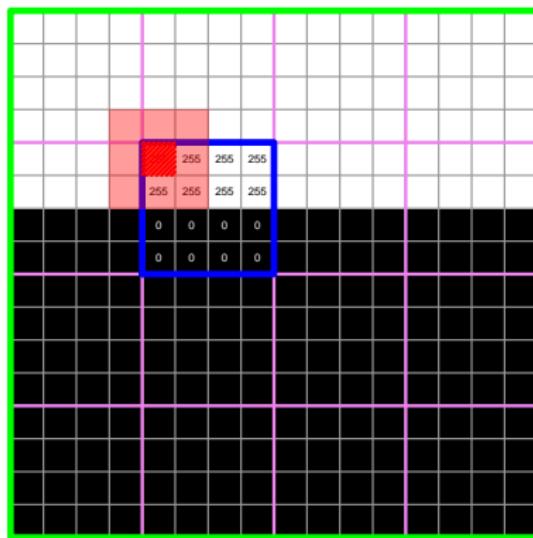
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

$$|G| = \sqrt{G_x^2 + G_y^2}$$
$$\theta = \arctan2(G_x, G_y) * \frac{180}{\pi}$$



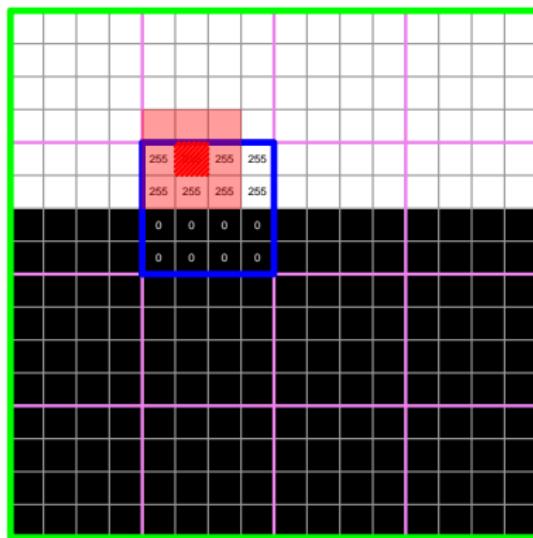
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



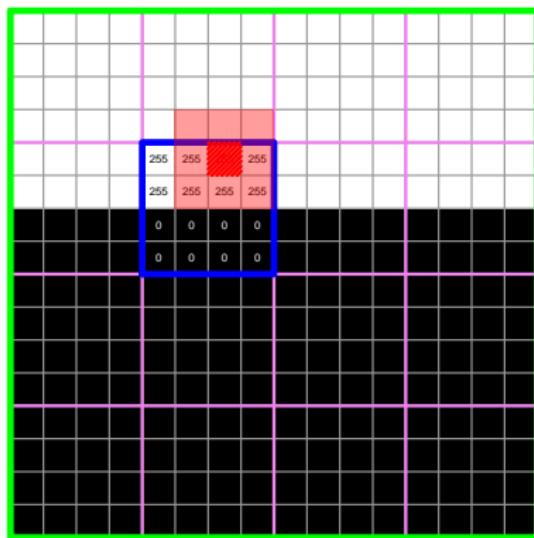
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



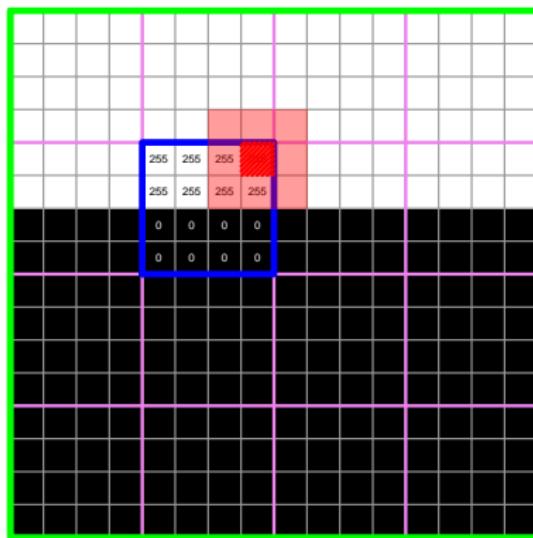
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



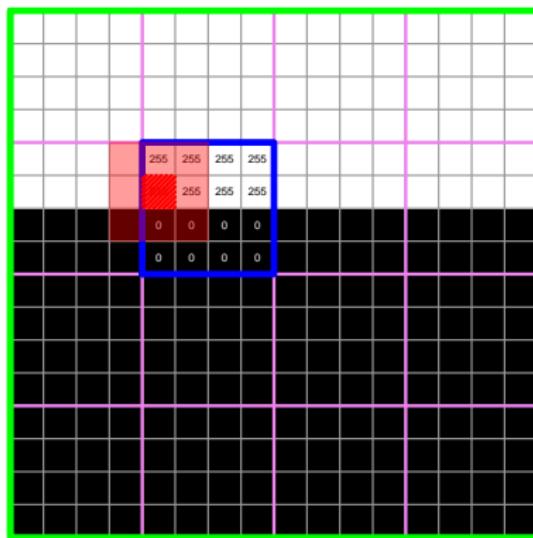
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



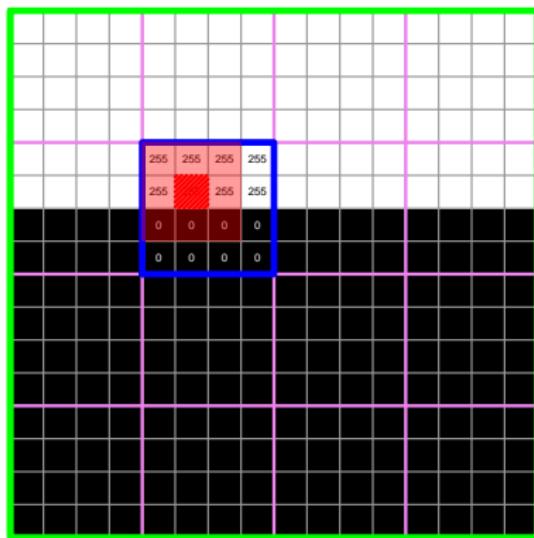
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



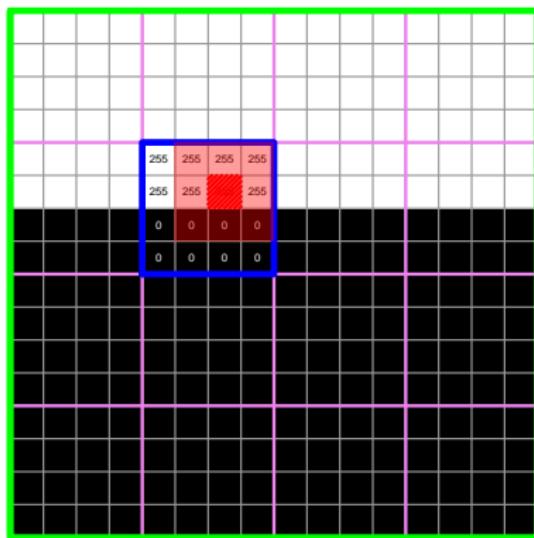
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



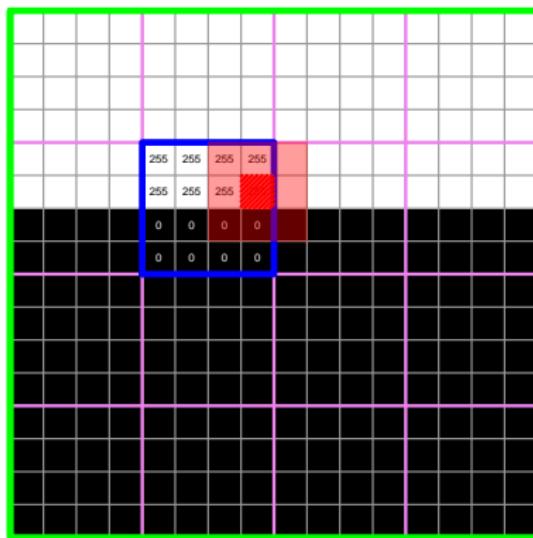
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



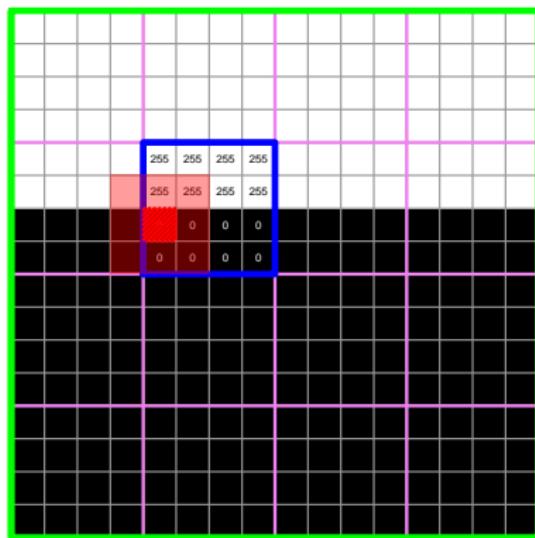
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



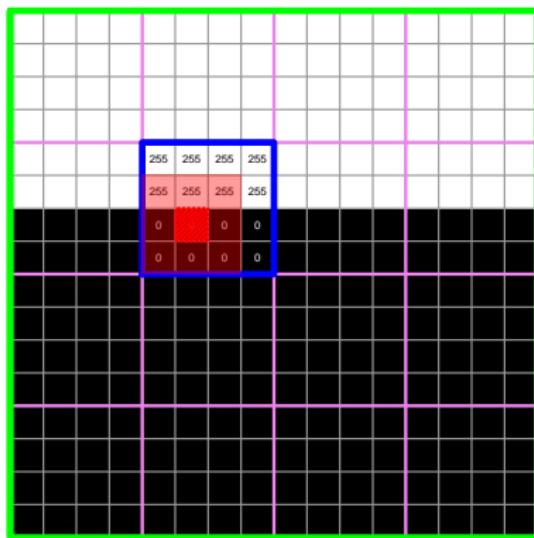
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



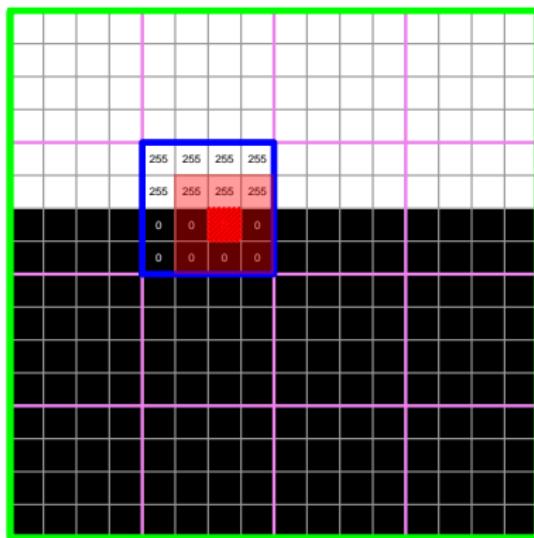
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



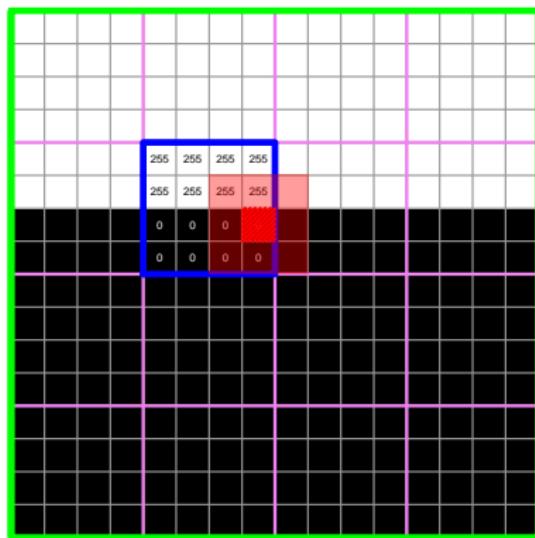
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



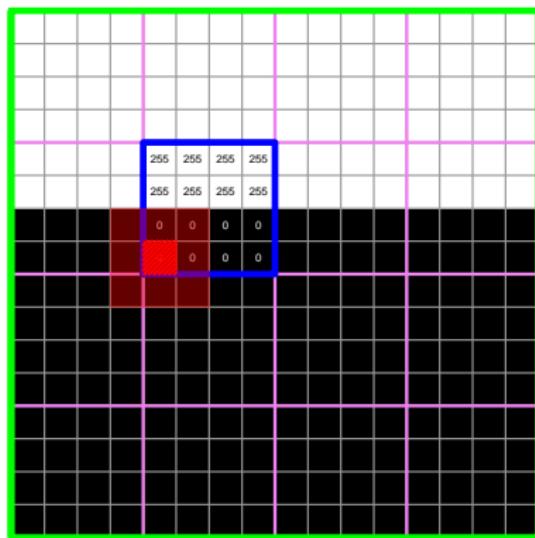
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



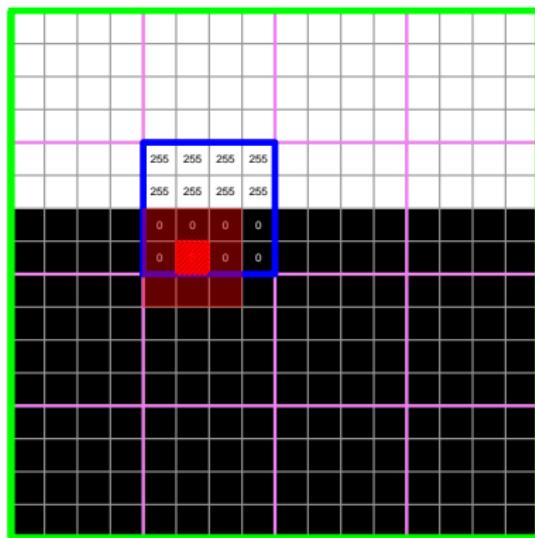
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



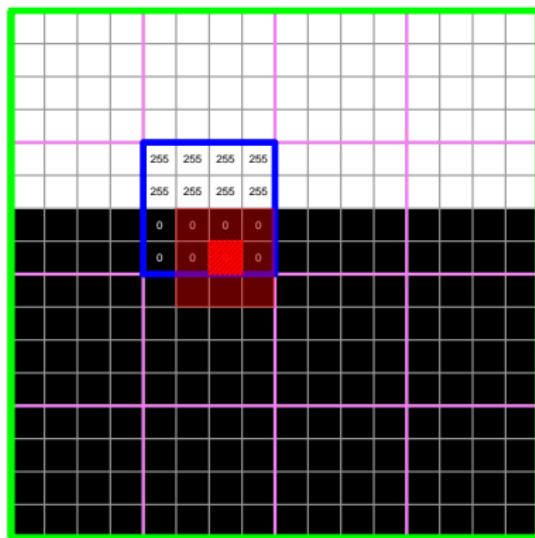
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



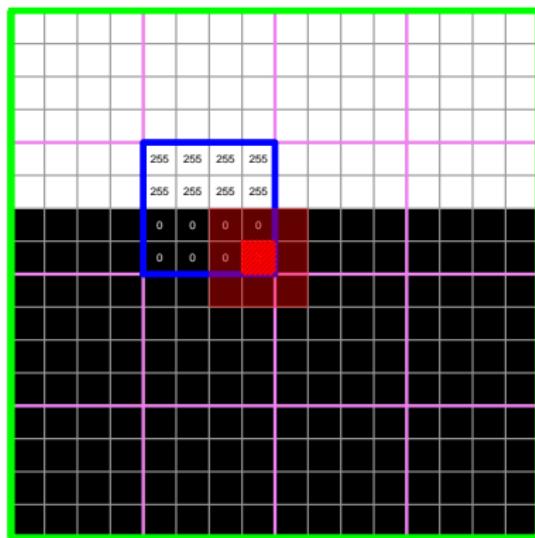
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



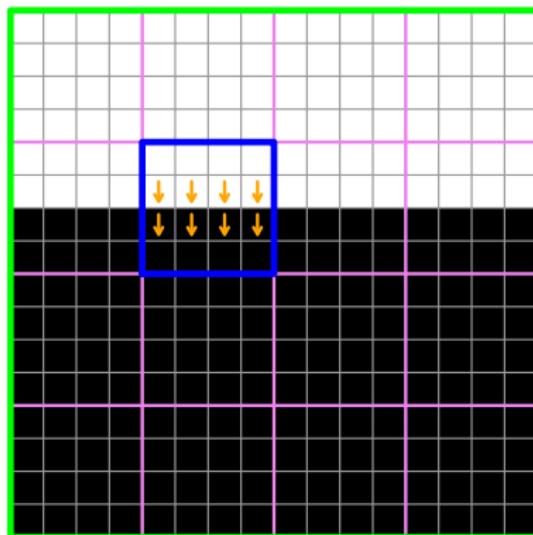
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Slide a kernel along pixels to estimate gradients:



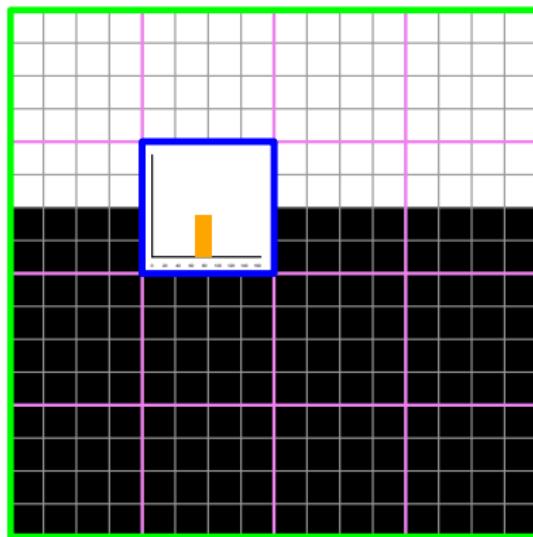
FEATURE IDENTIFICATION: FEATURE EXTRACTION (CONT.)

- Summarize gradient orientations and magnitudes

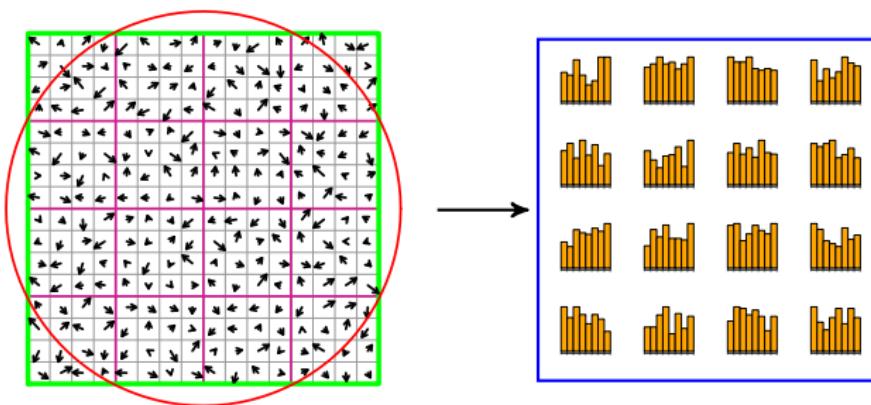


FEATURE EXTRACTION: STEP 4

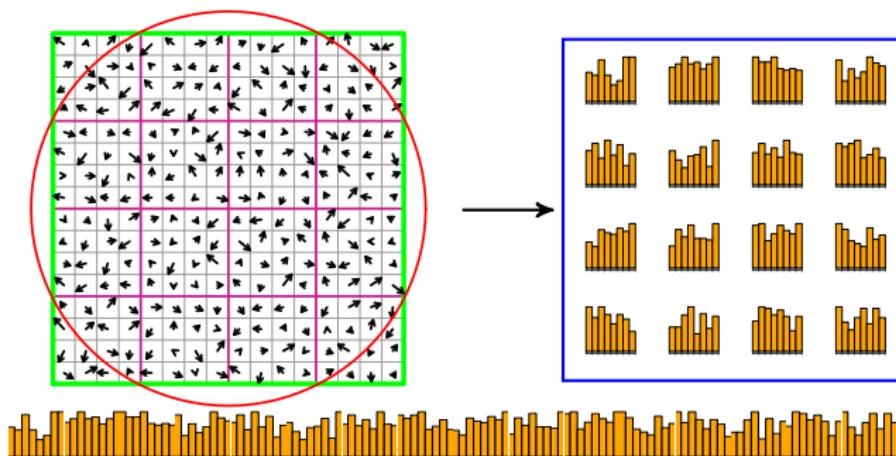
- Build HoG: 8-bins for the multiple orientations, weighted by magnitude and distance to the key point



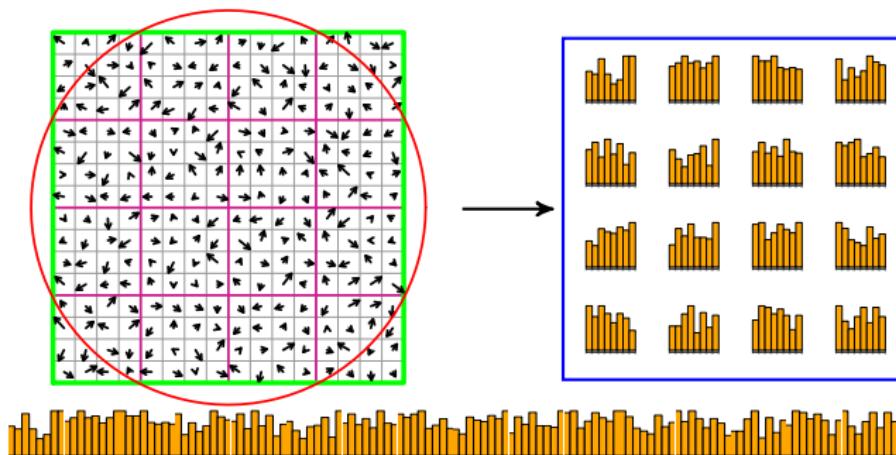
FEATURE EXTRACTION: FROM PIXELS TO VECTORS



FEATURE EXTRACTION: FROM PIXELS TO VECTORS

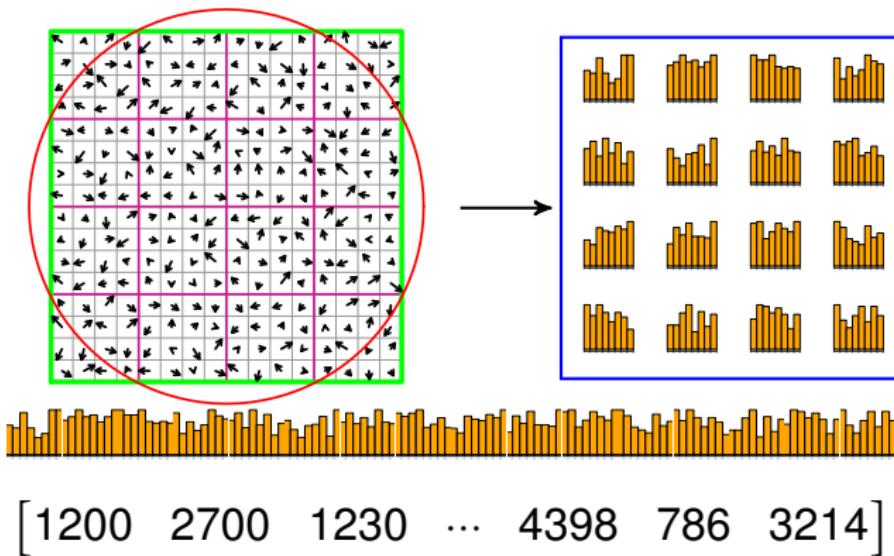


FEATURE EXTRACTION: FROM PIXELS TO VECTORS



[1200 2700 1230 ... 4398 786 3214]

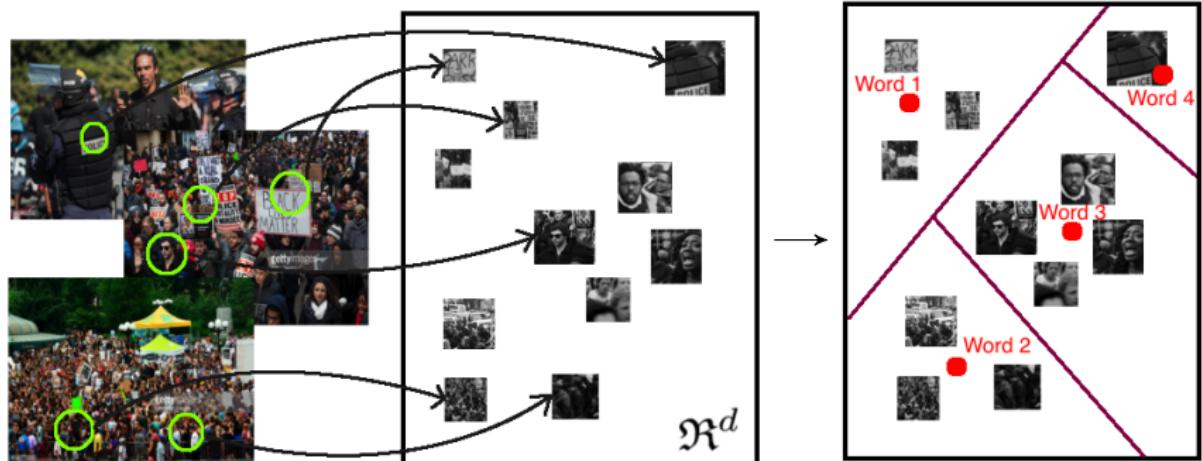
FEATURE EXTRACTION: FROM PIXELS TO VECTORS



Each point represented with a vector with 128 elements:
8-bins per histogram \times 16 cells

VISUAL VOCABULARY DEFINITION

- Cluster the features and take the centroid
- Mini-batch k -means



DEFINING A VISUAL WORD

- Intuitively: Clusters of mini-patches
- Formally: the vector representing the centroid of the cluster

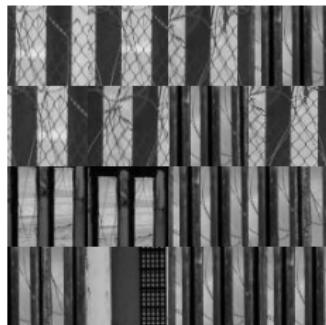


IMAGE-VISUAL WORD MATRIX

Document/Term	President	elections	...	migrants	troops	Central
<p>President Donald Trump has focused heavily on issues related to immigration in the run-up to the midterm elections, warning of an “invasion” of Central American migrants, and sending thousands of troops to the border.</p>	1	1	...	1	1	1
<p>President Donald Trump is trying to frame the upcoming midterm elections as a national referendum on immigration issues. The President complains that Mexico is not doing enough to stop the caravan of migrants.</p>	2	1	...	1	0	0
<p>Thousands of Central American migrants have again resumed their trek through southern Mexico after failing find buses to carry them. President Donald Trump said Wednesday that the deployment of active troops to the southern U.S. border could increase dramatically.</p>	1	1	...	1	1	1

IMAGE-VISUAL WORD MATRIX (CONT.)

Image/Visual Word			...	
	20	0	...	3
	0	7	...	5
	12	9	...	0

IMAGE-VISUAL WORD MATRIX

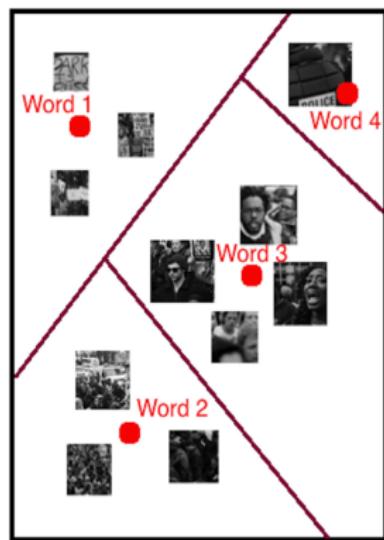


IMAGE-VISUAL WORD MATRIX

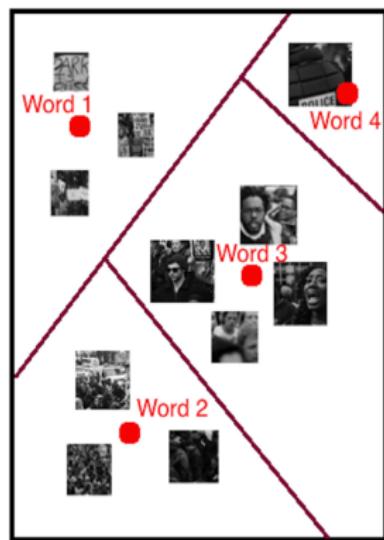


IMAGE-VISUAL WORD MATRIX

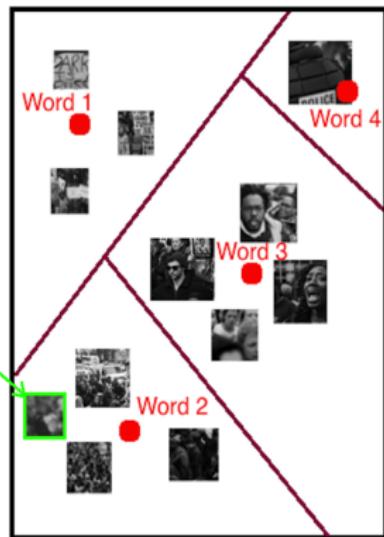


IMAGE-VISUAL WORD MATRIX

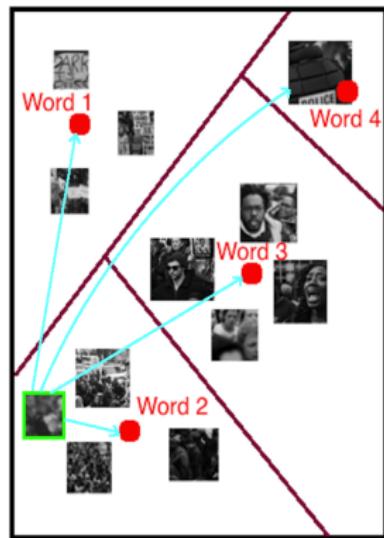


IMAGE-VISUAL WORD MATRIX

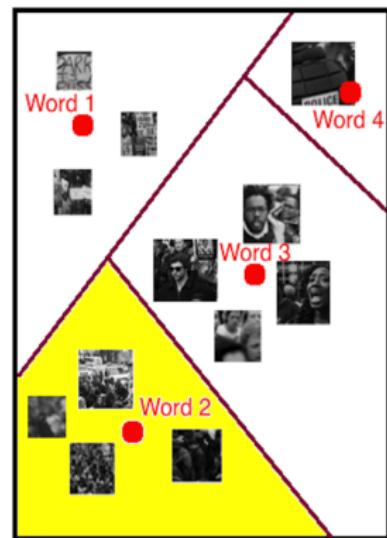


IMAGE-VISUAL WORD MATRIX: SOME CONSIDERATIONS

- Clusters = visual words
- Euclidean distance to determine similarity
- Every key-point is assigned to one word (something to consider)

THE IVWM IN PRACTICE

THE IVWM IN PRACTICE

Speed bump!

THE IVWM IN PRACTICE

Speed bump!

- ① BoVW approach to reduce the dimensionality the pictures

THE IVWM IN PRACTICE

Speed bump!

- ① BoVW approach to reduce the dimensionality the pictures
- ② We extracted **features** and linked them to tokens/references → **visual words**

THE IVWM IN PRACTICE

Speed bump!

- ① BoVW approach to reduce the dimensionality the pictures
- ② We extracted **features** and linked them to tokens/references → **visual words**
- ③ We built a Document-Term matrix style input → IVWM

THE IVWM IN PRACTICE

Speed bump!

- ① BoVW approach to reduce the dimensionality the pictures
- ② We extracted **features** and linked them to tokens/references → **visual words**
- ③ We built a Document-Term matrix style input → IVWM
- ④ And then...?

THE IVWM IN PRACTICE

Speed bump!

- ① BoVW approach to reduce the dimensionality the pictures
- ② We extracted **features** and linked them to tokens/references → **visual words**
- ③ We built a Document-Term matrix style input → IVWM
- ④ And then...? **Choose your detour!**

THE IVWM IN PRACTICE

Speed bump!

- ① BoVW approach to reduce the dimensionality the pictures
- ② We extracted **features** and linked them to tokens/references → **visual words**
- ③ We built a Document-Term matrix style input → IVWM
- ④ And then...? **Choose your detour!**

Sentiment
Analysis

THE IVWM IN PRACTICE

Speed bump!

- ① BoVW approach to reduce the dimensionality the pictures
- ② We extracted **features** and linked them to tokens/references → **visual words**
- ③ We built a Document-Term matrix style input → IVWM
- ④ And then...? **Choose your detour!**

Sentiment
Analysis

Binary
Classification

THE IVWM IN PRACTICE

Speed bump!

- ① BoVW approach to reduce the dimensionality the pictures
- ② We extracted **features** and linked them to tokens/references → **visual words**
- ③ We built a Document-Term matrix style input → IVWM
- ④ And then...? **Choose your detour!**

Sentiment
Analysis

Binary
Classification

Clustering
Analysis

THE IVWM IN PRACTICE

Speed bump!

- ① BoVW approach to reduce the dimensionality the pictures
- ② We extracted **features** and linked them to tokens/references → **visual words**
- ③ We built a Document-Term matrix style input → IVWM
- ④ And then...? **Choose your detour!**

Sentiment
Analysis

Binary
Classification

Clustering
Analysis

Topic
Modeling

THE IVWM IN PRACTICE

Speed bump!

- ① BoVW approach to reduce the dimensionality the pictures
- ② We extracted **features** and linked them to tokens/references → **visual words**
- ③ We built a Document-Term matrix style input → IVWM
- ④ And then...? **Choose your detour!**

Topic
Modeling

THE IVWM IN PRACTICE: STRUCTURAL TOPIC MODEL

- Structural Topic Model (Roberts et al. 2014)

THE IVWM IN PRACTICE: STRUCTURAL TOPIC MODEL

- Structural Topic Model (Roberts et al. 2014)
- Tool for topic modeling of texts with document-level covariate information

THE IVWM IN PRACTICE: STRUCTURAL TOPIC MODEL

- Structural Topic Model (Roberts et al. 2014)
- Tool for topic modeling of texts with document-level covariate information
- Mixture model:

THE IVWM IN PRACTICE: STRUCTURAL TOPIC MODEL

- Structural Topic Model (Roberts et al. 2014)
- Tool for topic modeling of texts with document-level covariate information
- Mixture model:
 - Probability that words belong to each of the “topics” or groups of interest

THE IVWM IN PRACTICE: STRUCTURAL TOPIC MODEL

- Structural Topic Model (Roberts et al. 2014)
- Tool for topic modeling of texts with document-level covariate information
- Mixture model:
 - Probability that words belong to each of the “topics” or groups of interest
 - Not a single classification outcome, but proportions of all potential topics for each document

THE IVWM IN PRACTICE: STRUCTURAL TOPIC MODEL

- Structural Topic Model (Roberts et al. 2014)
- Tool for topic modeling of texts with document-level covariate information
- Mixture model:
 - Probability that words belong to each of the “topics” or groups of interest
 - Not a single classification outcome, but proportions of all potential topics for each document



THE IVWM IN PRACTICE: STRUCTURAL TOPIC MODEL

- Structural Topic Model (Roberts et al. 2014)
- Tool for topic modeling of texts with document-level covariate information
- Mixture model:
 - Probability that words belong to each of the “topics” or groups of interest
 - Not a single classification outcome, but proportions of all potential topics for each document



Sky:

THE IVWM IN PRACTICE: STRUCTURAL TOPIC MODEL

- Structural Topic Model (Roberts et al. 2014)
- Tool for topic modeling of texts with document-level covariate information
- Mixture model:
 - Probability that words belong to each of the “topics” or groups of interest
 - Not a single classification outcome, but proportions of all potential topics for each document



Sky:

Crowd:

THE IVWM IN PRACTICE: STRUCTURAL TOPIC MODEL

- Structural Topic Model (Roberts et al. 2014)
- Tool for topic modeling of texts with document-level covariate information
- Mixture model:
 - Probability that words belong to each of the “topics” or groups of interest
 - Not a single classification outcome, but proportions of all potential topics for each document



Sky:

Crowd:

Pavement:

THE IVWM IN PRACTICE: STRUCTURAL TOPIC MODEL

- Structural Topic Model (Roberts et al. 2014)
- Tool for topic modeling of texts with document-level covariate information
- Mixture model:
 - Probability that words belong to each of the “topics” or groups of interest
 - Not a single classification outcome, but proportions of all potential topics for each document



Sky:

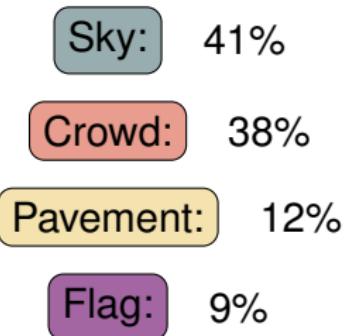
Crowd:

Pavement:

Flag:

THE IVWM IN PRACTICE: STRUCTURAL TOPIC MODEL

- Structural Topic Model (Roberts et al. 2014)
- Tool for topic modeling of texts with document-level covariate information
- Mixture model:
 - Probability that words belong to each of the “topics” or groups of interest
 - Not a single classification outcome, but proportions of all potential topics for each document



ILLUSTRATING EXAMPLE: MIGRANT CARAVAN

ILLUSTRATING EXAMPLE: MIGRANT CARAVAN

- Groups of migrants from Central America fleeing violence in their countries and seeking refugee in the U.S.

ILLUSTRATING EXAMPLE: MIGRANT CARAVAN

- Groups of migrants from Central America fleeing violence in their countries and seeking refugee in the U.S.
- Very polarized coverage of this phenomenon

ILLUSTRATING EXAMPLE: MIGRANT CARAVAN

- Groups of migrants from Central America fleeing violence in their countries and seeking refugee in the U.S.
- Very polarized coverage of this phenomenon
- Emphasis on **magnitude**: threat, invasion

ILLUSTRATING EXAMPLE: MIGRANT CARAVAN

- Groups of migrants from Central America fleeing violence in their countries and seeking refugee in the U.S.
- Very polarized coverage of this phenomenon
- Emphasis on **magnitude**: threat, invasion

“**Massive** migrant caravan on the way”

“Looks more like an **invasion** than anything”



ILLUSTRATING EXAMPLE: MIGRANT CARAVAN

- Groups of migrants from Central America fleeing violence in their countries and seeking refugee in the U.S.
- Very polarized coverage of this phenomenon
- Emphasis on **magnitude**: threat, invasion

“**Massive** migrant caravan on the way”

“Looks more like an **invasion** than anything”



“See them as they are: **Desperate**, leaving behind whatever they had, and whomever they knew, all for a **better chance** at life”

IDENTIFYING POLITICAL COMPONENTS IN THE DATA GENERATION PROCESS OF IMAGES

- Goal: Identify and quantify the visual framing of the magnitude of the caravan

IDENTIFYING POLITICAL COMPONENTS IN THE DATA GENERATION PROCESS OF IMAGES

- Goal: Identify and quantify the visual framing of the magnitude of the caravan
- Structural Topic Model to identify underlying “topics”, understood as frames, in the images

IDENTIFYING POLITICAL COMPONENTS IN THE DATA GENERATION PROCESS OF IMAGES

- **Goal:** Identify and quantify the visual framing of the **magnitude** of the caravan
- Structural Topic Model to identify underlying “topics”, understood as frames, in the images
- Visual codebook generated from \approx 6,000 pictures from *Getty*

IDENTIFYING POLITICAL COMPONENTS IN THE DATA GENERATION PROCESS OF IMAGES

- Goal: Identify and quantify the visual framing of the magnitude of the caravan
- Structural Topic Model to identify underlying “topics”, understood as frames, in the images
- Visual codebook generated from \approx 6,000 pictures from *Getty*
- 500 words vocabulary

IDENTIFYING POLITICAL COMPONENTS IN THE DATA GENERATION PROCESS OF IMAGES

- Goal: Identify and quantify the visual framing of the magnitude of the caravan
- Structural Topic Model to identify underlying “topics”, understood as frames, in the images
- Visual codebook generated from \approx 6,000 pictures from *Getty*
- 500 words vocabulary
- Prevalence covariates: agency and date

IDENTIFYING POLITICAL COMPONENTS IN THE DATA GENERATION PROCESS OF IMAGES

- Goal: Identify and quantify the visual framing of the magnitude of the caravan
- Structural Topic Model to identify underlying “topics”, understood as frames, in the images
- Visual codebook generated from \approx 6,000 pictures from *Getty*
- 500 words vocabulary
- Prevalence covariates: agency and date
- Selection of 6 topics:

IDENTIFYING POLITICAL COMPONENTS IN THE DATA GENERATION PROCESS OF IMAGES

- Goal: Identify and quantify the visual framing of the magnitude of the caravan
- Structural Topic Model to identify underlying “topics”, understood as frames, in the images
- Visual codebook generated from \approx 6,000 pictures from *Getty*
- 500 words vocabulary
- Prevalence covariates: agency and date
- Selection of 6 topics:
 - Crowd

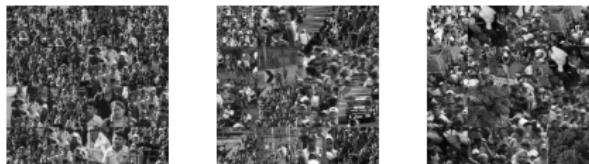
IDENTIFYING POLITICAL COMPONENTS IN THE DATA GENERATION PROCESS OF IMAGES

- Goal: Identify and quantify the visual framing of the magnitude of the caravan
- Structural Topic Model to identify underlying “topics”, understood as frames, in the images
- Visual codebook generated from \approx 6,000 pictures from *Getty*
- 500 words vocabulary
- Prevalence covariates: agency and date
- Selection of 6 topics:
 - Crowd
 - Border/Fence, Small group/Portrait, Water/Sky, Camps, Darkness

UNDERLYING TOPICS IN THE CARAVAN: FREX WORDS

UNDERLYING TOPICS IN THE CARAVAN: FREX WORDS

Topic 1: Crowds



Topic 2: Border/Fence



Topic 3: Water/Sky



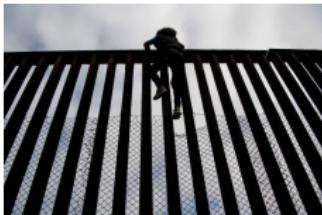
UNDERLYING TOPICS IN THE CARAVAN: REPRESENTATIVE IMAGES

UNDERLYING TOPICS IN THE CARAVAN: REPRESENTATIVE IMAGES

Crowd



Border/
Fence



Water/
Sky



WRAP-UP

- Clean and traceable feature extraction

WRAP-UP

- Clean and traceable feature extraction
- Allows you to use models that DO NOT require training or labeled data

WRAP-UP

- Clean and traceable feature extraction
- Allows you to use models that DO NOT require training or labeled data
- Helpful for exploratory analyses

WRAP-UP

- Clean and traceable feature extraction
- Allows you to use models that DO NOT require training or labeled data
- Helpful for exploratory analyses
- Allows a better understanding and interpretation of the results/classification

WRAP-UP

- Clean and traceable feature extraction
- Allows you to use models that DO NOT require training or labeled data
- Helpful for exploratory analyses
- Allows a better understanding and interpretation of the results/classification
- No color though...

WRAP-UP

- Clean and traceable feature extraction
- Allows you to use models that DO NOT require training or labeled data
- Helpful for exploratory analyses
- Allows a better understanding and interpretation of the results/classification
- No color though...
- ... and not SUPER accurate if there are very particular concepts that you want to measure

WRAP-UP

- Clean and traceable feature extraction
- Allows you to use models that DO NOT require training or labeled data
- Helpful for exploratory analyses
- Allows a better understanding and interpretation of the results/classification
- No color though...
- ... and not SUPER accurate if there are very particular concepts that you want to measure
- (CNNs to the rescue!)

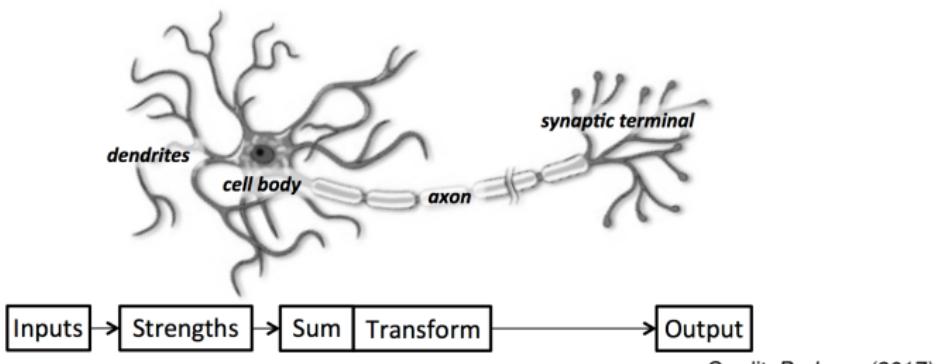
Convolutional Neural Networks

SEEING LIKE A HUMAN

- Modern computer vision systems are meant to emulate how human brains transform sensual stimuli into conceptual understanding
- The process allows computers to set their own set of rules to classify information

SEEING LIKE A HUMAN

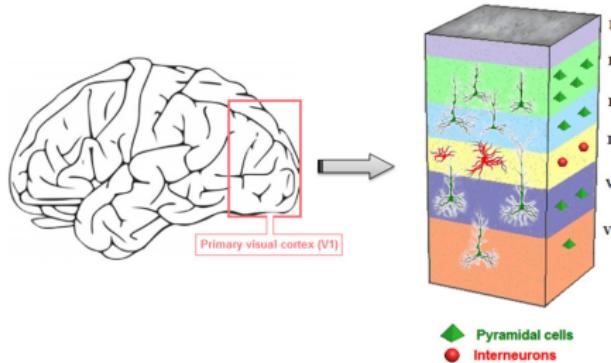
- Modern computer vision systems are meant to emulate how human brains transform sensual stimuli into conceptual understanding
- The process allows computers to set their own set of rules to classify information



Credit: Buduma (2017)

- Neurons are the core unit of brains.
- They receive information from other neurons, process the input, and send the result to other cells for further processing.

SEEING LIKE A HUMAN, CONT.



Credit: Bachatene, Bharmauria and Molotchnikoff (2012).

- Neurons are organized into layers.
- Every layer breaks down the signal into small pieces, allowing each of its neurons to focus on a unique piece of information.
- The first layers identify basic visual patterns, intermediate layers transform patterns into shapes, and the last layers convert shapes into objects.

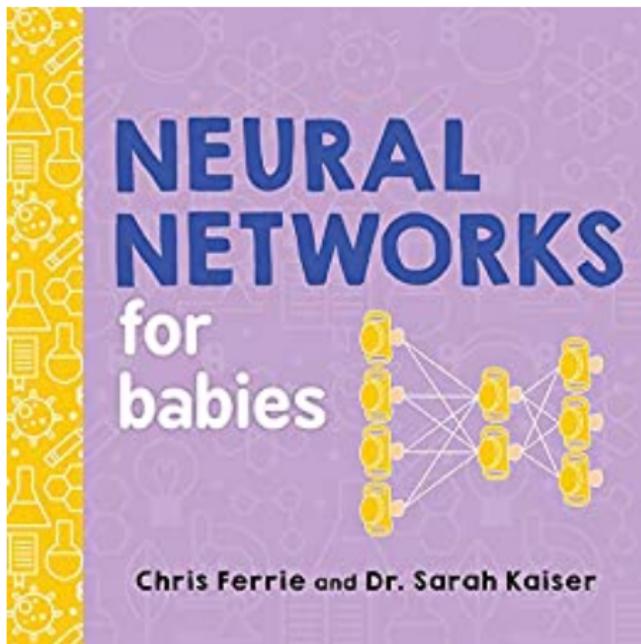
WAIT... WHAT?

WAIT... WHAT?

A very sophisticated text that I've been reading a lot recently:

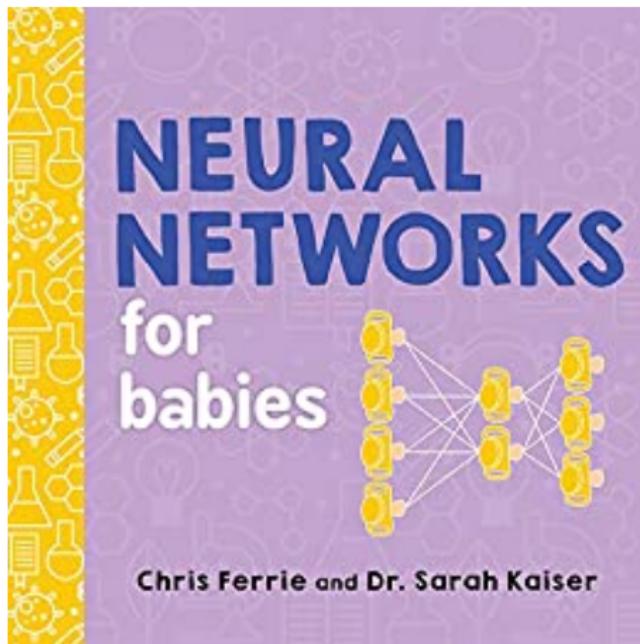
WAIT... WHAT?

A very sophisticated text that I've been reading a lot recently:



WAIT... WHAT?

A very sophisticated text that I've been reading a lot recently:

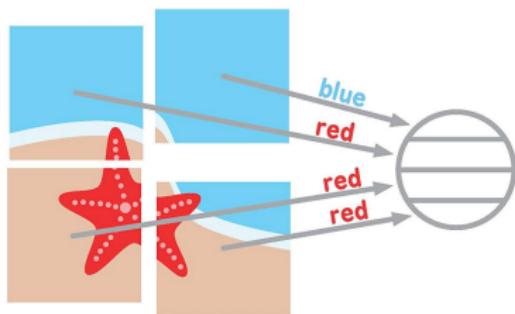


(No... I am not joking)

THE LOGIC OF CNNs

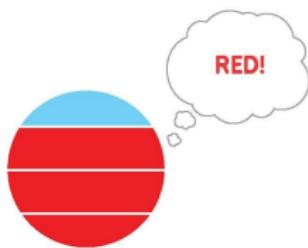


Is there a red animal in this picture?

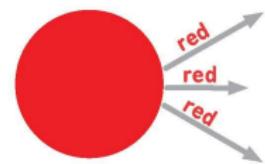


The neuron can decide based on its input.

THE LOGIC OF CNNs



When the neuron has an answer,

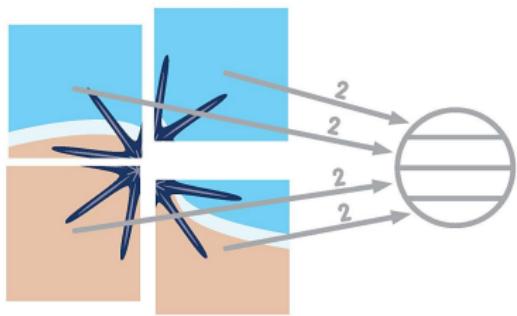


it sends its own message.

THE LOGIC OF CNNs

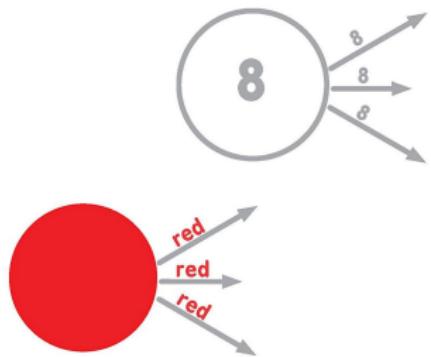


Does this animal have 8 arms?

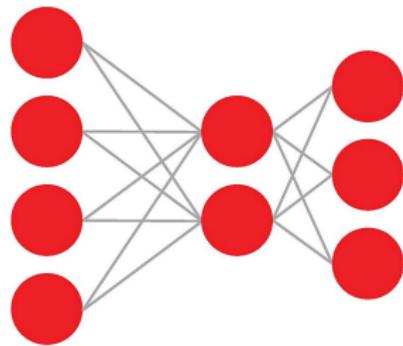


The neuron can decide based on its input.

THE LOGIC OF CNNs



Where do the messages go?



Neurons talk to each other.
They connect together in a network.

CONVOLUTIONAL NEURAL NETWORKS (CNNs)

- Convolutional Neural Networks (CNNs) is a supervised learning algorithm to classify images
- CNNs gradually learn what visual features of the image are more important in a classification task by transforming the image into multiple representations or *feature maps*.
- CNNs are organized into multiple layers. Each layer contains multiple representations of the original image through maps of visual features such as edges, blobs or color combinations.
- The part of learning and reaching a semantic concept that humans conduct by trial and error is achieved through the training, validation and testing procedures in CNNs.

NETWORK STRUCTURE

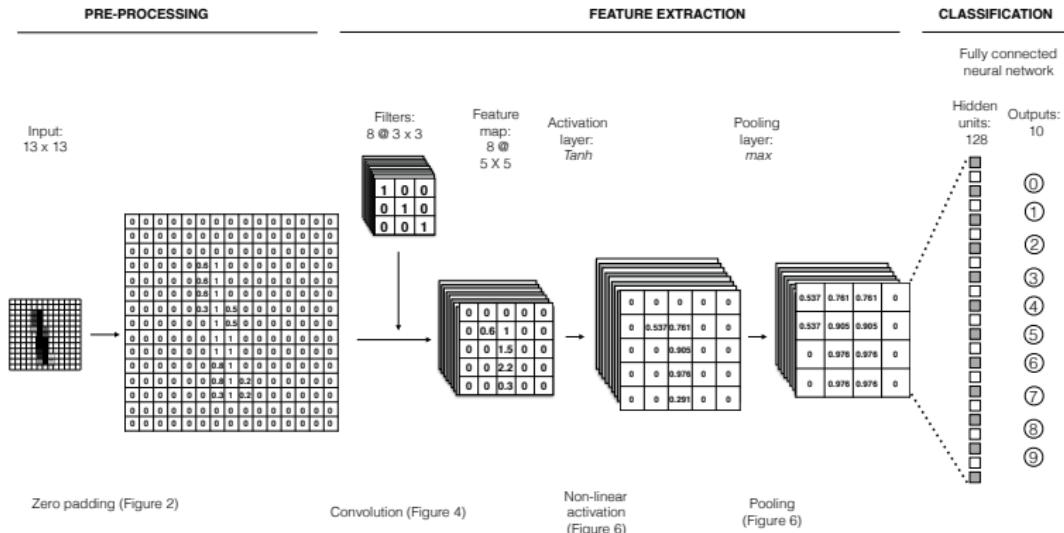
- **GOAL:** learn the features associated w/ outcomes

NETWORK STRUCTURE

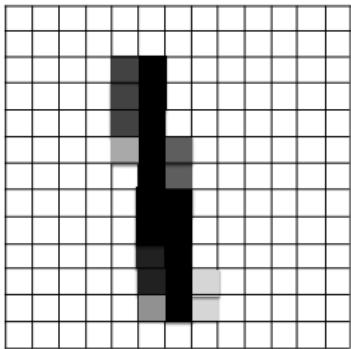
- **GOAL:** learn the features associated w/ outcomes
- Translation: obtain “coefficients” [weights in feature maps]
- Mainly, a data reduction technique → **Why?**

NETWORK STRUCTURE

- **GOAL:** learn the features associated w/ outcomes
- Translation: obtain “coefficients” [weights in feature maps]
- Mainly, a data reduction technique → **Why?**
- Not a black-box! → Optimization of error



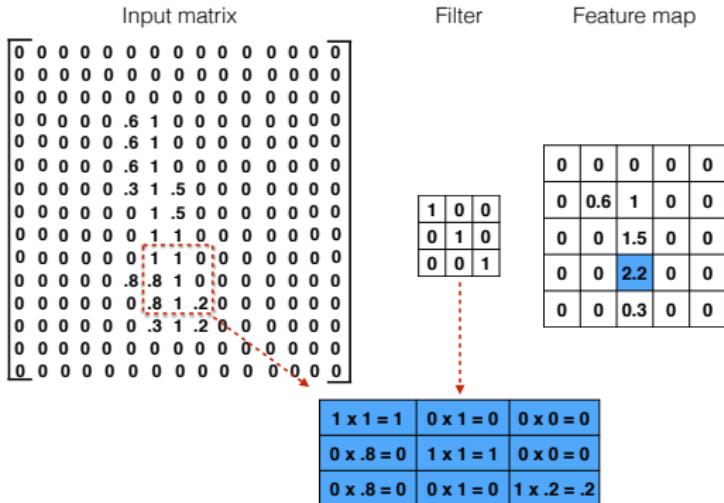
REPRESENTING IMAGES



0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	.6	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	.6	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	.6	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	.3	1	.5	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	.5	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	.8	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	.8	1	2	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	.3	1	2	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

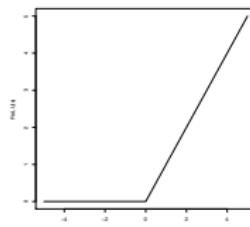
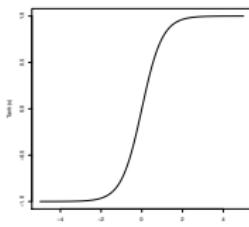
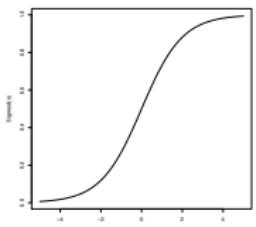
The image is transformed into a numerical matrix, where each element represents the value of a specific pixel of the image measured as light intensity (in grayscale images) or color intensity (in color images).

FEATURE EXTRACTION



Filters are matrixes made of *weights*, that maximize or minimize the “intensity” of a pixel. Every filter slides through each 3×3 pixel area of the image, and computes the dot product of the region. The result is recorded on a smaller matrix to create *feature maps*. Intuitively, we want to detect whether and where a feature represented by a filter is prominent in the image.

ACTIVATION FUNCTIONS



$$(a) \text{Sigmoid}(x) = \frac{1}{1+e^{-x}}$$

$$(b) \text{Tanh}(x) = \frac{2}{1+e^{-2x}} - 1$$

$$(c) \text{ReLU}(x) = \begin{cases} 0 & \text{if } x < 0, \\ x & \text{otherwise.} \end{cases}$$

We add non-linearity by including an *activation layer*.

POOLING STAGE

Non-linear activation:
 $\text{Tanh}(x)$

max pooling

0	0	0	0	0	0	0	0	0
0	0.6	1	0	0	0.537	0.761	0	0
0	0	1.5	0	0	0	0.905	0	0
0	0	2.2	0	0	0	0.976	0	0
0	0	0.3	0	0	0	0.291	0	0

0	0	0	0	0	0.537	0.761	0.761	0
0.537	0.905	0.905	0.905	0	0.976	0.976	0.976	0
0	0.976	0.976	0.976	0	0	0.976	0.976	0
0	0.976	0.976	0.976	0	0	0	0	0

Once the activation map shows non-linear outputs, we reduce its dimensionality using a *pooling layer*. A pooling layer shrinks the size of the matrix while keeping the most important information in the feature map.

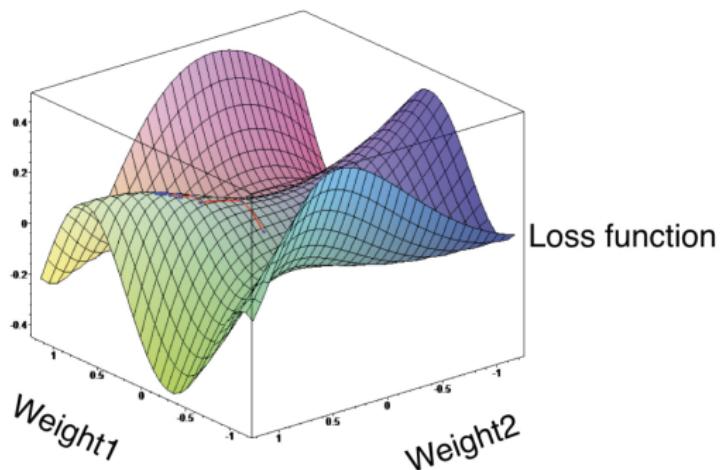
LEARNING

- The last stage of the network involves the classification of the image. The way in which the CNN learns the features that correlate to each outcome follows a procedure called back-propagation.

[More on back-propagation](#)

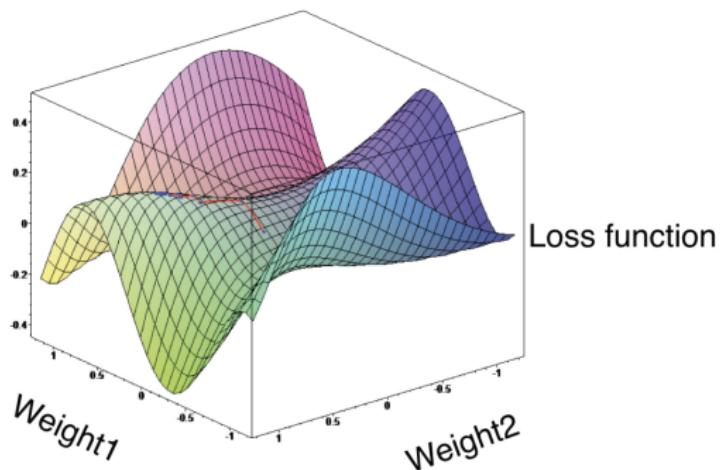
ACTUALLY, THIS SHOULD BE FAMILIAR...

Loss function



ACTUALLY, THIS SHOULD BE FAMILIAR...

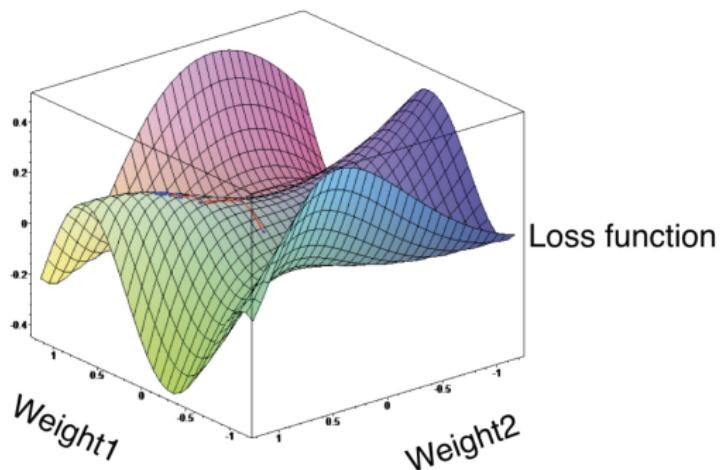
Loss function



- Minimize multidimensional loss function →

ACTUALLY, THIS SHOULD BE FAMILIAR...

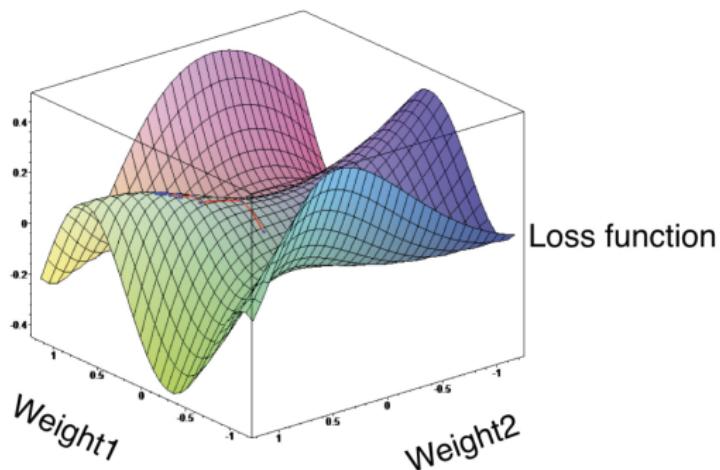
Loss function



- Minimize multidimensional loss function →

ACTUALLY, THIS SHOULD BE FAMILIAR...

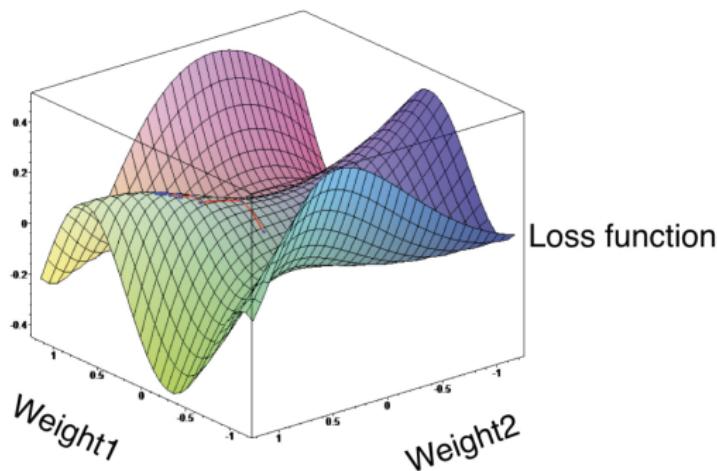
Loss function



- Minimize multidimensional loss function → (OLS anyone?)
- By finding the minimum point [=minimum prediction error]

ACTUALLY, THIS SHOULD BE FAMILIAR...

Loss function



- Minimize multidimensional loss function → (OLS anyone?)
- By finding the minimum point [=minimum prediction error]
- Explore the “field” step by step

BACK-PROPAGATION

- Allows gradient descent to update all weights (or coefficients) in a neural network

BACK-PROPAGATION

- Allows gradient descent to update all weights (or coefficients) in a neural network
- The core: estimating derivatives (use of chain rule!)

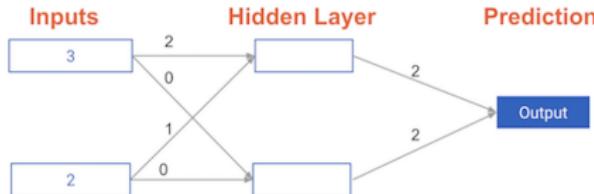
BACK-PROPAGATION

- Allows gradient descent to update all weights (or coefficients) in a neural network
- The core: estimating derivatives (use of chain rule!)

BACK-PROPAGATION

- Allows gradient descent to update all weights (or coefficients) in a neural network
- The core: estimating derivatives (use of chain rule!)

What is the error (predicted - actual) for the following network using the ReLU activation function when the input data is [3, 2] and the actual value of the target (what you are trying to predict) is 5? It may be helpful to get out a pen and piece of paper to calculate these values.



EXAMPLE: HANDWRITING RECOGNITION

13. RESULTADOS DE LA VOTACIÓN DE DIPUTADOS FEDERALES (Escriba los votos para cada partido político, candidatos no registrados y votos nulos, sumélos y escriba el resultado en TOTAL). En caso de no recibir votos para algún partido o candidato escriba ceros.

PARTIDO POLÍTICO	RESULTADOS DE LA VOTACIÓN DE DIPUTADOS FEDERALES DE MAYORÍA RELATIVA <small>(Con ítems)</small>
PAN	0 3 8
PRD	0 7 2
PPS	0 0 3
PVEM	0 1 4
PT	0 0 3
PRD	0 0 2
MARCO	2 3 6
morena	0 0 2
IPN	0 2 0
ENCUENTRO SOCIA	0 0 1
CANDIDATOS NO REGISTRADOS	0 0 0
VOTOS NULOS	3 1 9
TOTAL	7 1 0

14. ¿ES IGUAL LA CANTIDAD DEL APARTADO 13 CON EL TOTAL DE LOS VOTOS DEL APARTADO 8? SI NO
(Marque con "X")

15. SE PRESENTARON INCIDENTES DURANTE EL ESCRUTINIO Y CÓMPUTO DE LA ELECCIÓN DE DIPUTADOS FEDERALES? SÍ NO
(Marque con "X")

DESCRIBA BREVEMENTE:

EN SU CASO, SE ESCRIBIERON EN _____ HOJA(S) DE INCIDENTES, MISMA(S) QUE SE ANEXA(N) A LA PRESENTE ACTA.

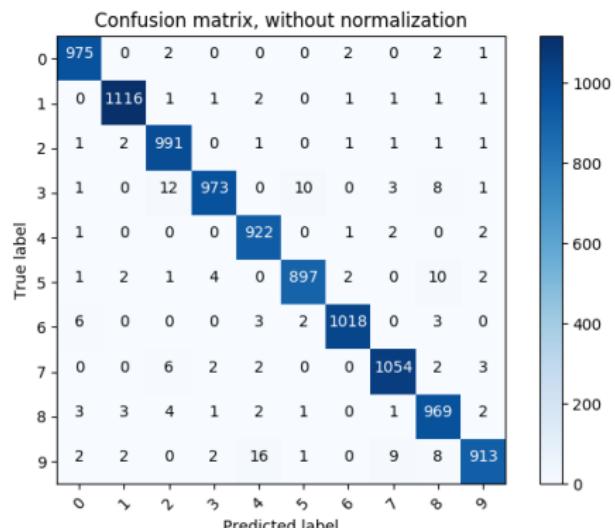
(Con número)

TRAIN AND TEST

- MNIST dataset: 70,000 samples of numbers
- Train: 60,000
- Test: 10,000
- Accuracy rate: Only MNIST=98.3%, Real data= 96.4%

TRAIN AND TEST

- MNIST dataset: 70,000 samples of numbers
- Train: 60,000
- Test: 10,000
- Accuracy rate: Only MNIST=98.3%, Real data= 96.4%



PREDICT THE NUMBERS

2

3

6

0

0

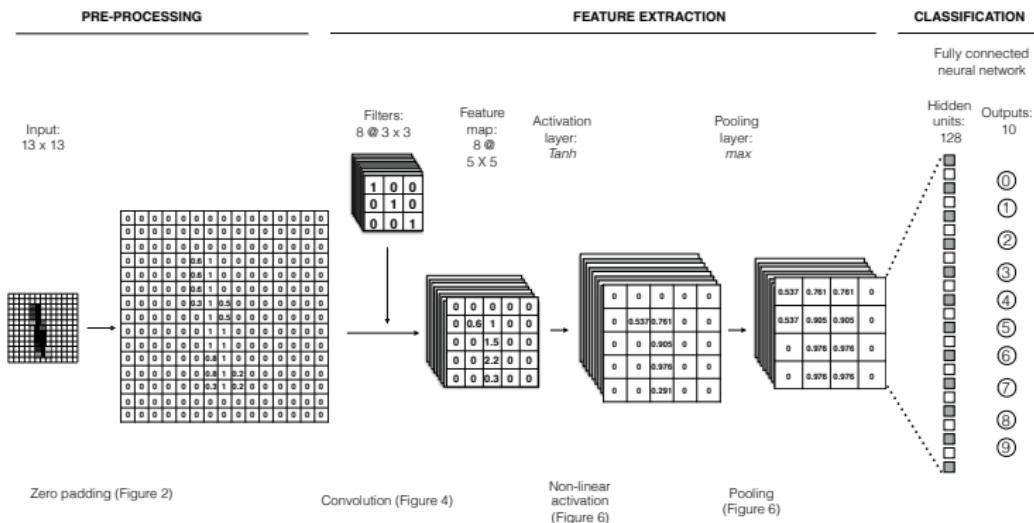
2

CHALLENGES AND RECOMMENDATIONS

- Prevent overfitting
 - Increase number of training images
 - Data augmentation
 - Dropout random neurons
- Optimize your training set
 - Active learning: Informativeness vs. Representativeness
 - Class balance
 - “Denoise” images
 - Batch normalization
 - CAUTION: Bias training
- Post-CNN diagnosis
 - Know your training, testing and out-of-sample data
 - Always check mislabeled examples: validate, validate, validate...
 - Diagnosis

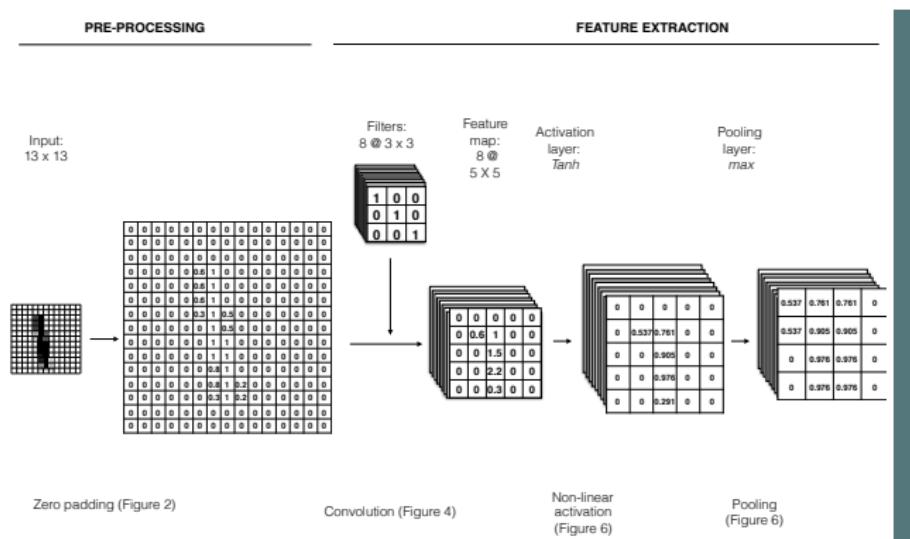
A QUICK NOTE ON TRANSFER LEARNING

- “Freeze” some layers and retrain the active ones
 - Idea: keep something useful learned features and fine-tune your labels of interest



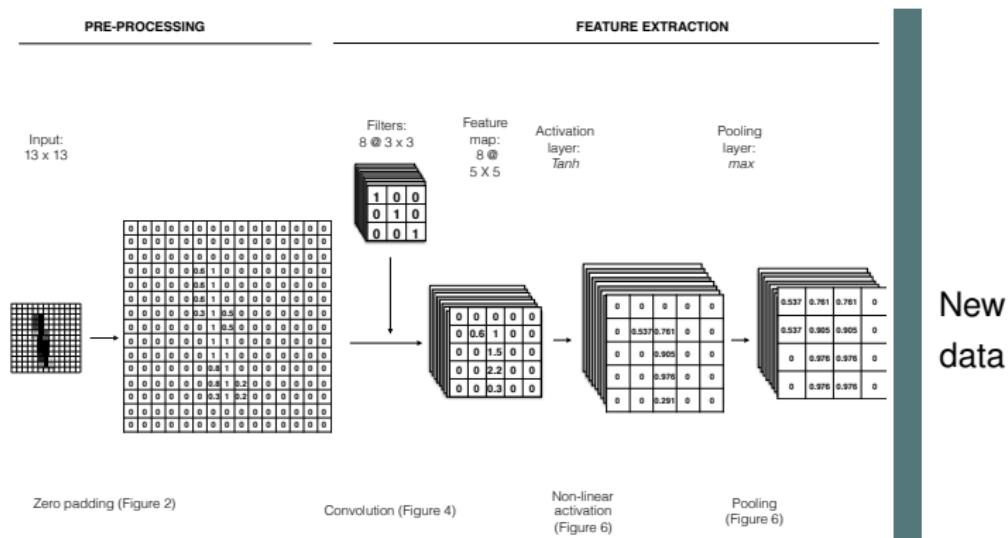
A QUICK NOTE ON TRANSFER LEARNING

- “Freeze” some layers and retrain the active ones
- Idea: keep something useful learned features and fine-tune your labels of interest



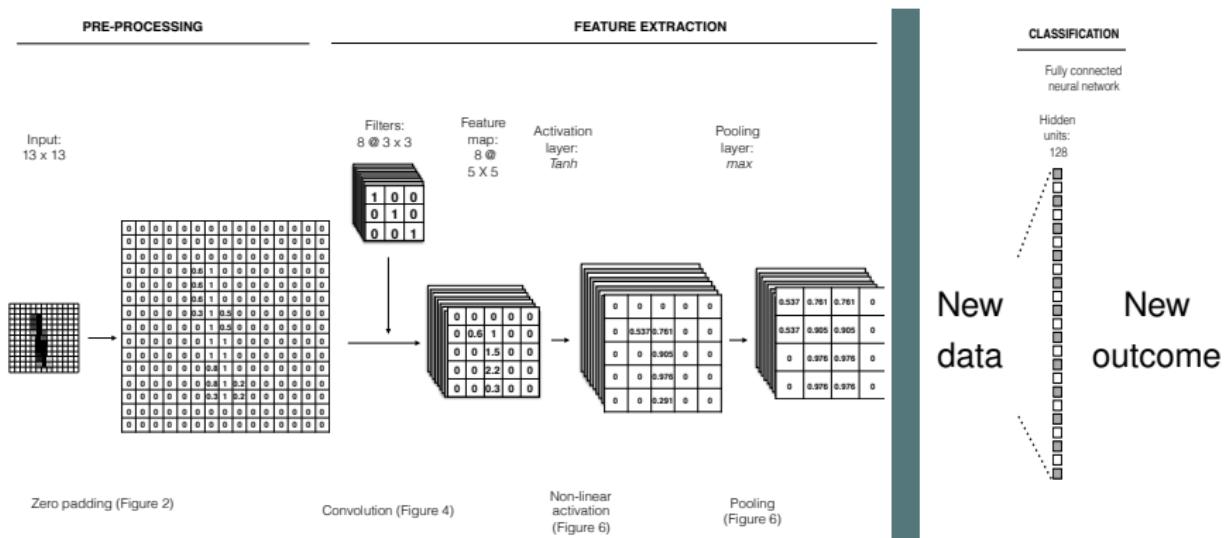
A QUICK NOTE ON TRANSFER LEARNING

- “Freeze” some layers and retrain the active ones
- Idea: keep something useful learned features and fine-tune your labels of interest



A QUICK NOTE ON TRANSFER LEARNING

- “Freeze” some layers and retrain the active ones
- Idea: keep something useful learned features and fine-tune your labels of interest



IN PRACTICE

- Design your structure and create your training data
 - AWS machines, HPC or GPUs [computational power needed]
 - Pre-trained architectures in Google, Amazon, etc.
 - Creating training data: `imglab`
- Pre-canned image detection with API access
 - **GoogleVision:** <https://cloud.google.com/vision/>, Amazon, Microsoft
 - Labels found in each picture
 - Face detection
 - Emotions
 - Sensitive content

OBJECT DETECTION: COVERS OF NEWSPAPERS

Full set of images



Only Women's March images



FACE DETECTION AND EMOTIONAL CONTENT

Good results with little effort...



FACE DETECTION AND EMOTIONAL CONTENT

Good results with little effort...



...but also tons of errors



UNINTENDED BUT DANGEROUS CONSEQUENCES



YouTube

Search



LIVE

MARC MIMRAM, Architect, DPLO
NOTRE DAME FIRE
MAYOR LAMENTS 'TERRIBLE FIRE' AT PARIS CATHEDRAL

BREAKING FRANCE • Paris mayor laments 'terrible fire' at Notre-Dame cathedral

FRANCE 24

SUBSCRIBE



September 11 attacks

September 11 attacks, also called 9/11 attacks, series of airline hijackings and suicide attacks committed in 2001 by 19 militants associated with the Islamic extremist group al-Qaeda against targets in the United States, the deadliest terrorist attacks on American soil in U.S. history. The attacks against New York City and Washington, D.C., caused extensive death and destruction and triggered an enormous U.S. effort to...

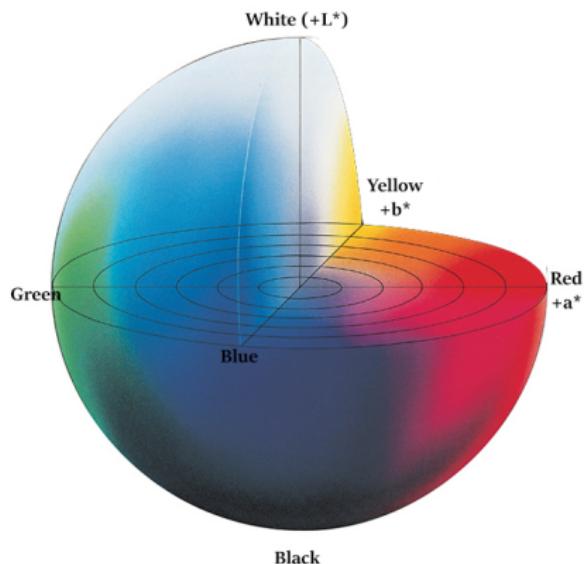
Encyclopedia Britannica

OTHER CONSIDERATIONS

- Pay attention to your training data
- Consider using transfer learning or pre-trained architectures
- Take advantages of APIs
- Understand the limitations and scope of the method

L^{*}A^{*}B COLOR SPACE

- L^{*} = lightness
- a^{*} = chromaticity coordinate (red axis)
- b^{*} = chromaticity coordinate (blue axis)



BACKPROPAGATION

- Neuron j 's classification outcome: y_j .
- Target outcome: t_j .
- Sum of prediction errors $E = \frac{1}{2} \sum_{j \in 10} (t_j - y_j)^2$
- Error derivative function of last layer: $\frac{\partial E}{\partial y_j} = -(t_j - y_j)$

Similarly, we can express the error derivatives in terms of the logit of the neuron, z_j :

$$\frac{\partial E}{\partial z_j} = \frac{\partial E}{\partial y_j} \frac{\partial y_j}{\partial z_j} = y_j(1 - y_j) \frac{\partial E}{\partial y_j}$$

BACKPROPAGATION, CONT.

- Estimate how the neuron outcomes in layer i affect the outputs of layer j given the weighted connection between both layers, w_{ij}

$$\frac{\partial E}{\partial y_i} = \sum_j \frac{\partial E}{\partial z_j} \frac{\partial z_j}{\partial y_i} = \sum_j w_{ij} \frac{\partial E}{\partial z_j} = \sum_j w_{ij} y_j (1 - y_j) \frac{\partial E}{\partial y_j}$$

- These partial derivatives allow us to estimate the contribution of a specific weight to the error term:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial z_j}{\partial w_{ij}} \frac{\partial E}{\partial z_j} = y_j (1 - y_j) \frac{\partial E}{\partial y_i}$$

$$-\Delta w_{ij} = - \sum_{k \in K} y_i^{(k)} y_j^{(k)} (1 - y_j^{(k)}) \frac{\partial E^{(k)}}{\partial y_i}$$