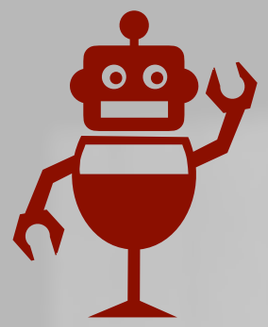


Efficient Image Search and Identification: The Making of WINE-O.AI

Michelle L. Gill, Ph.D.
Senior Data Scientist, Metis
@modernscientist

SciPy 2017

link.mlgill.co/scipy2017

Metis Data Science Training

- Data Science Bootcamp
  - 12 Week, In-Person
  - New York, San Francisco, Chicago, Seattle
- Corporate Training
  - Python for Data Science
  - Machine Learning
  - Natural Language Processing
  - Spark
- Evening Professional Development Courses
- Explore Data Science Online Training

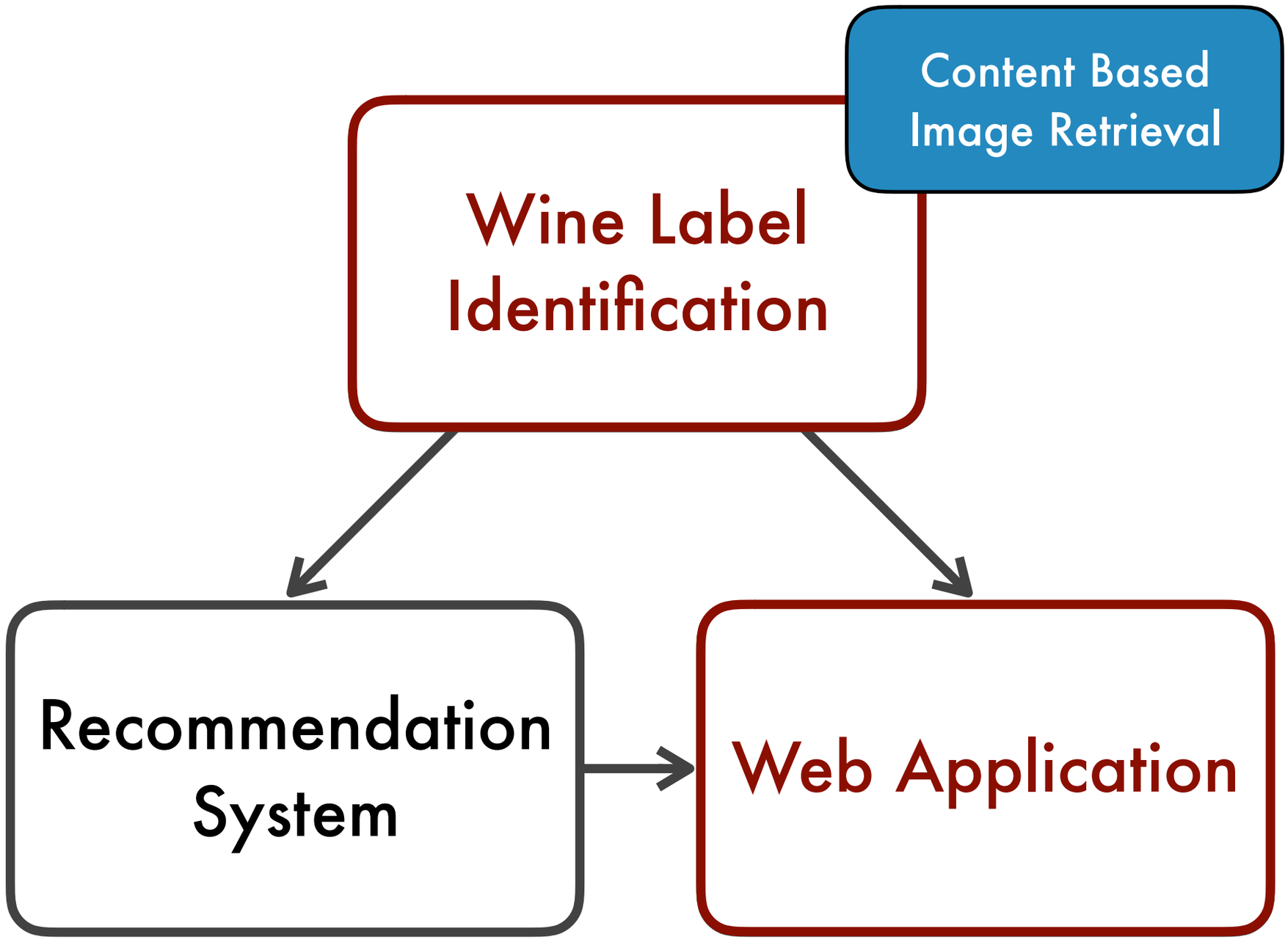
thisismetis.com

METIS

Motivation for WINE-O.AI

- Facilitate discovery and exploration of new wine
- Open source, computer vision project to share with the community

Ingredients of WINE-O.AI



Content Based Image Retrieval

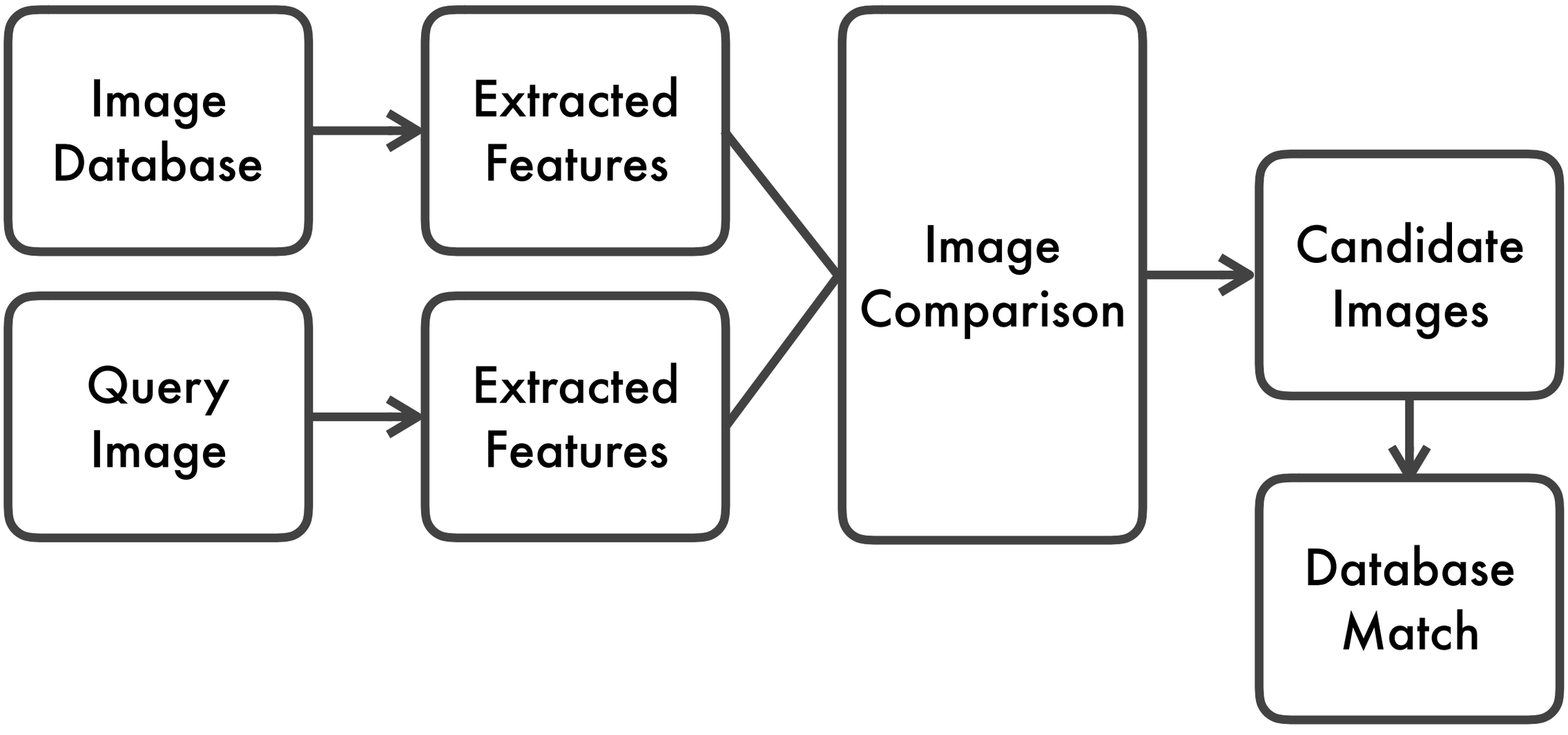


Image Comparison Challenges

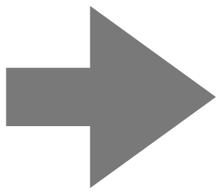
Query Image

Database Image

- Must be robust to differences in size, rotation, occlusion, and illumination
- And search must remain fast!

Content Based Image Retrieval in WINE-O.AI

1. Scraped ~50,000 Wine Labels
2. Identification of Image Features with SIFT
3. Creation of Bag of Visual Words
4. Image Selection with RANSAC



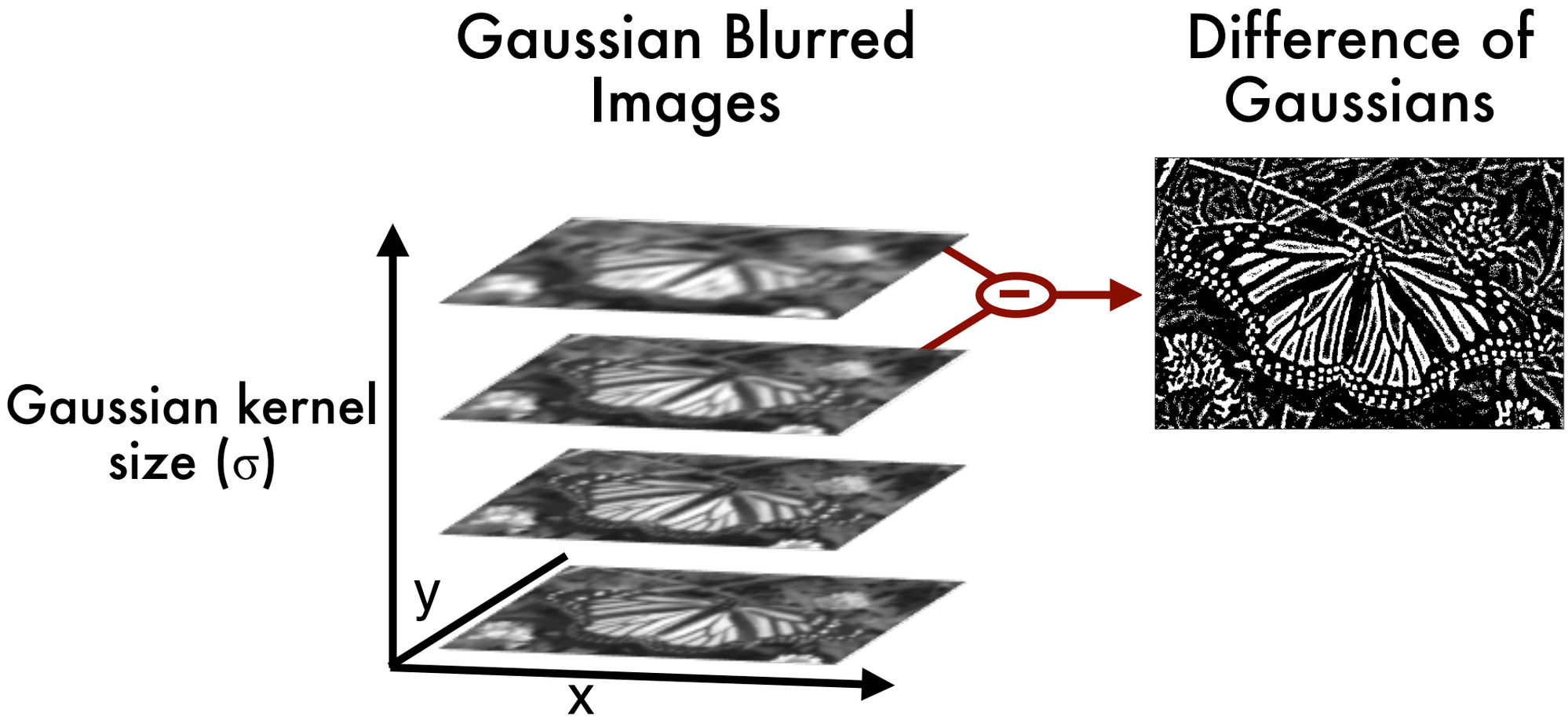
Web Application

SIFT Feature Detection

- Scale Invariant Feature Transformation (SIFT)
- Blur images using a Gaussian function of increasing width (σ)

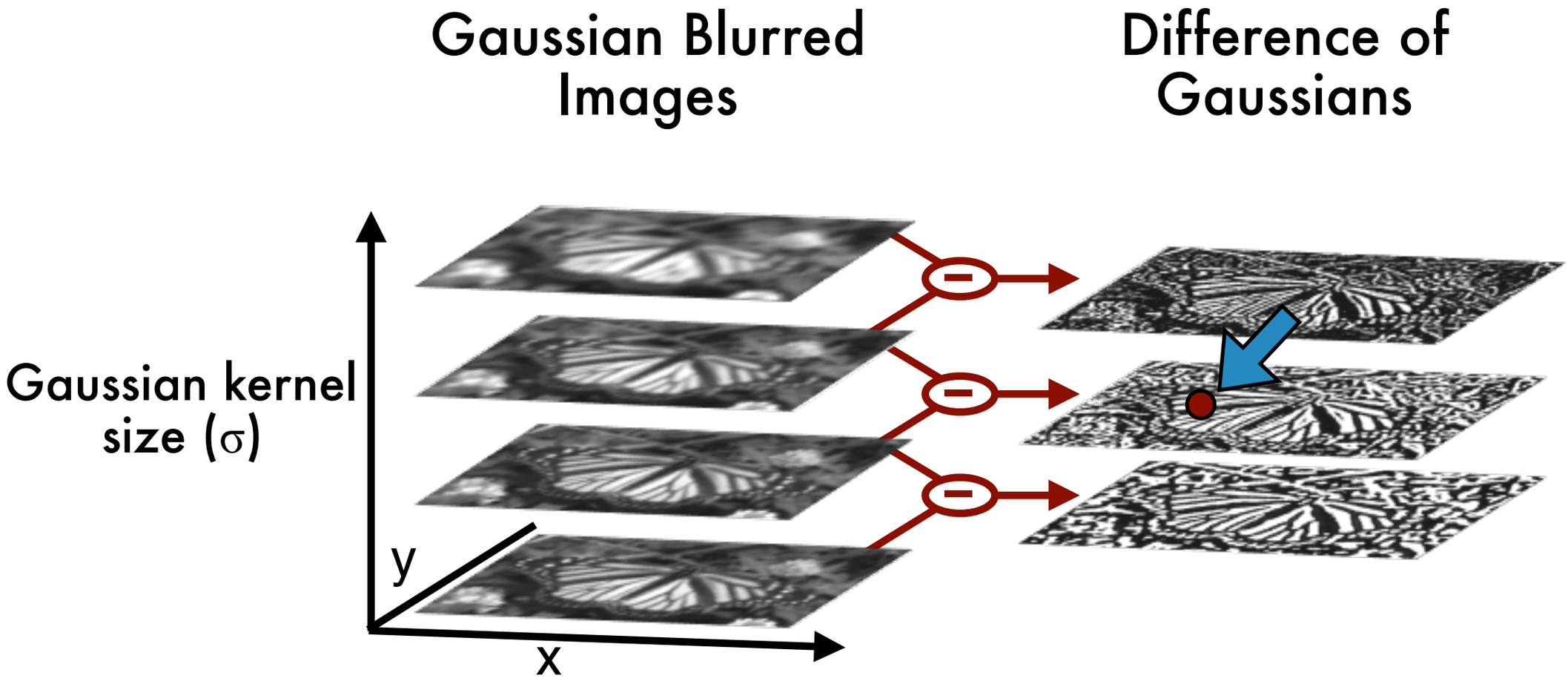
Lowe, D., *IJCV*, 2004

SIFT Feature Detection



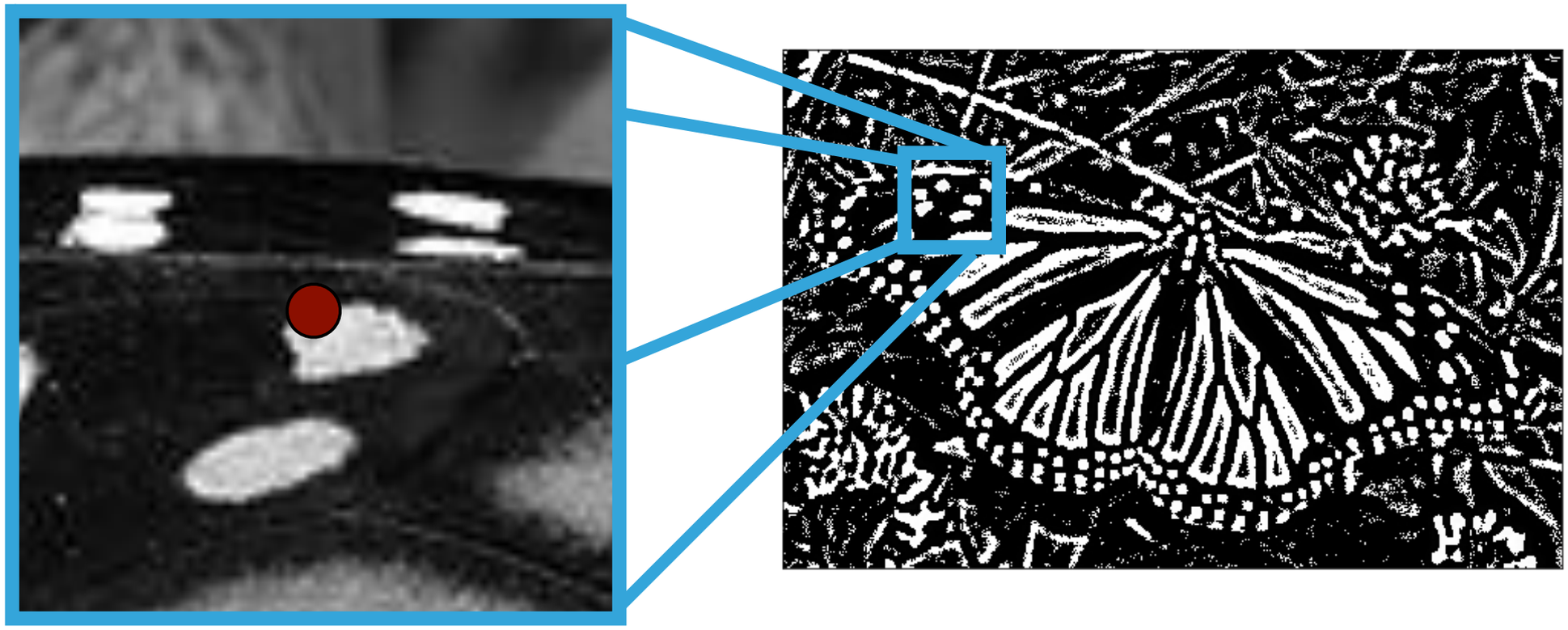
- Align images in 3D scale-space (x, y, σ)
- Subtract adjacent images

SIFT Feature Detection

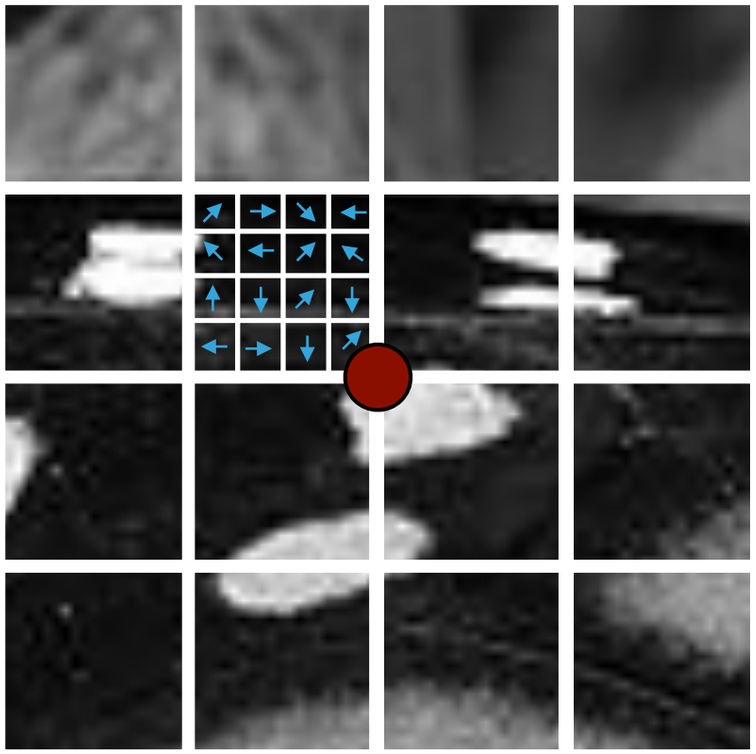


- Align images in 3D scale-space (x, y, σ)
- Subtract adjacent images
- Local extrema evaluated as potential features

SIFT Feature Descriptor

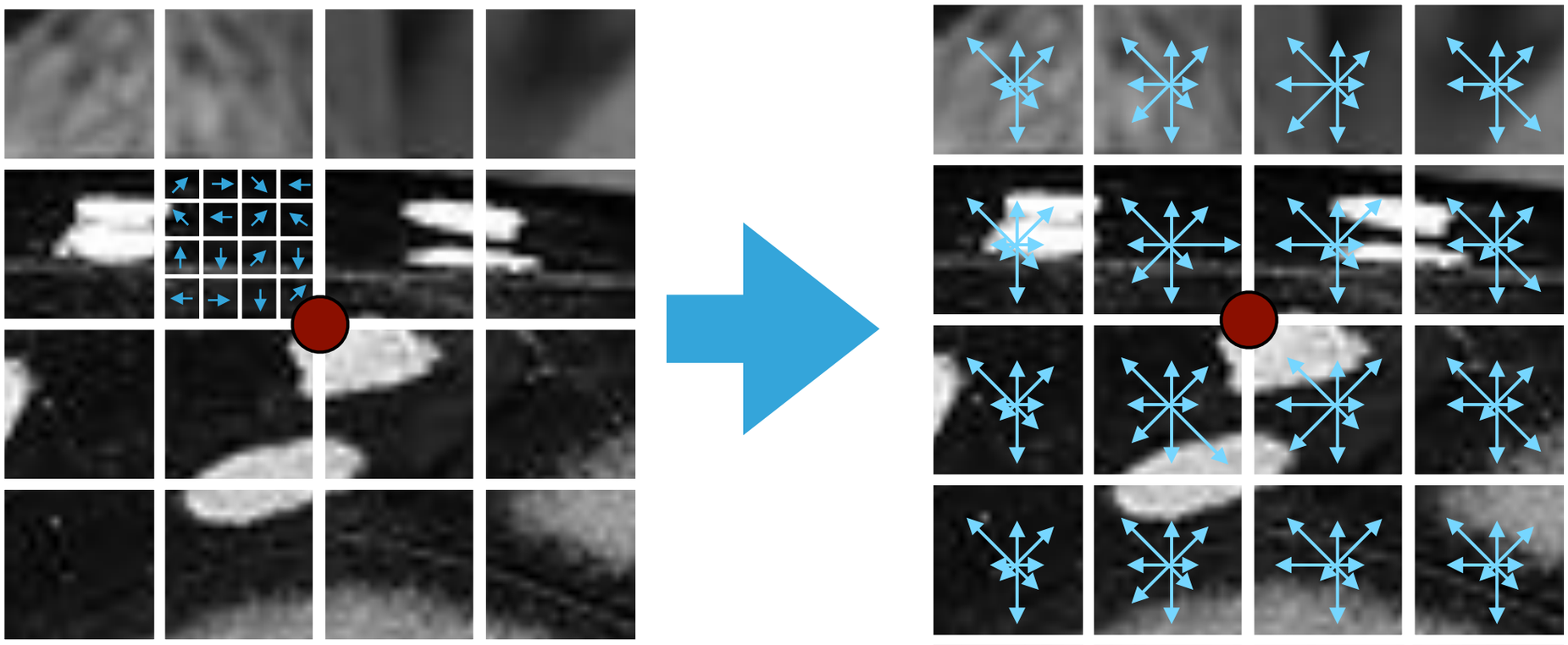


SIFT Feature Descriptor



- Descriptor calculated in 16 regions around key point
- Changes in intensity calculated and binned

SIFT Feature Descriptor



- Descriptor calculated in 16 regions around key point
- Changes in intensity calculated and binned
- Produces 128 dimension descriptor for each key point

SIFT in Practice

```
# image = array of grayscale, resized image pixels
```

Load and Process Image

```
# Detect features
detector = cv2.FeatureDetector_create('SIFT')
keypoints = detector.detect(image)
```

Feature Detection

```
# Get feature descriptors
descriptor = cv2.DescriptorExtractor_create('SIFT')
keypoints, features = descriptor.compute(image,
                                         keypoints)
```

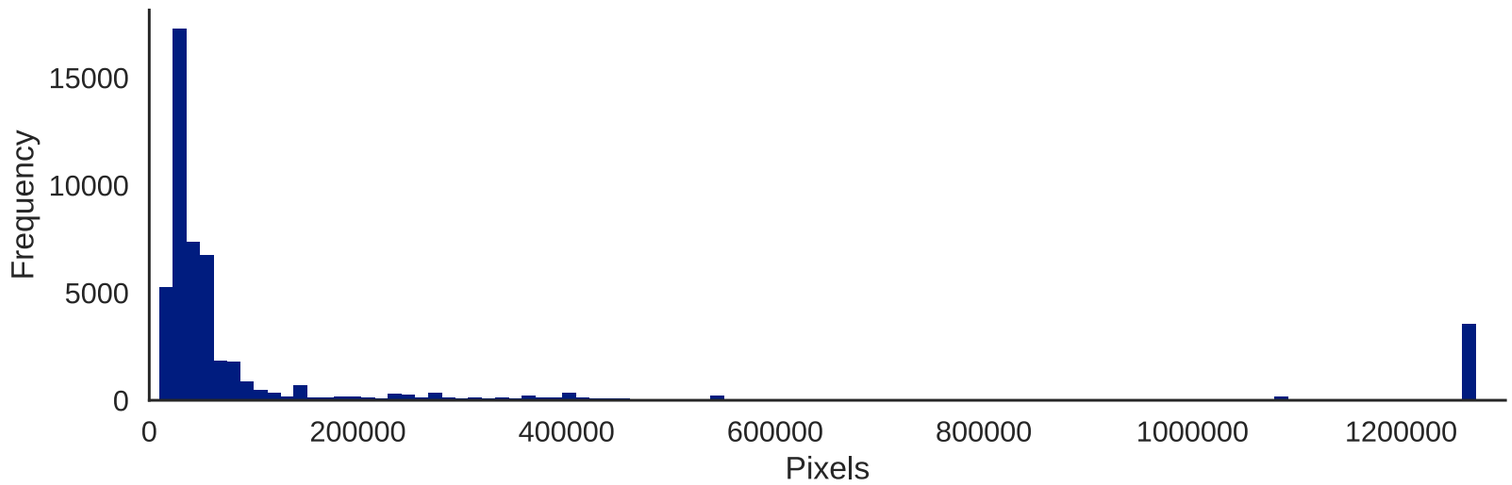
Get Feature Descriptors

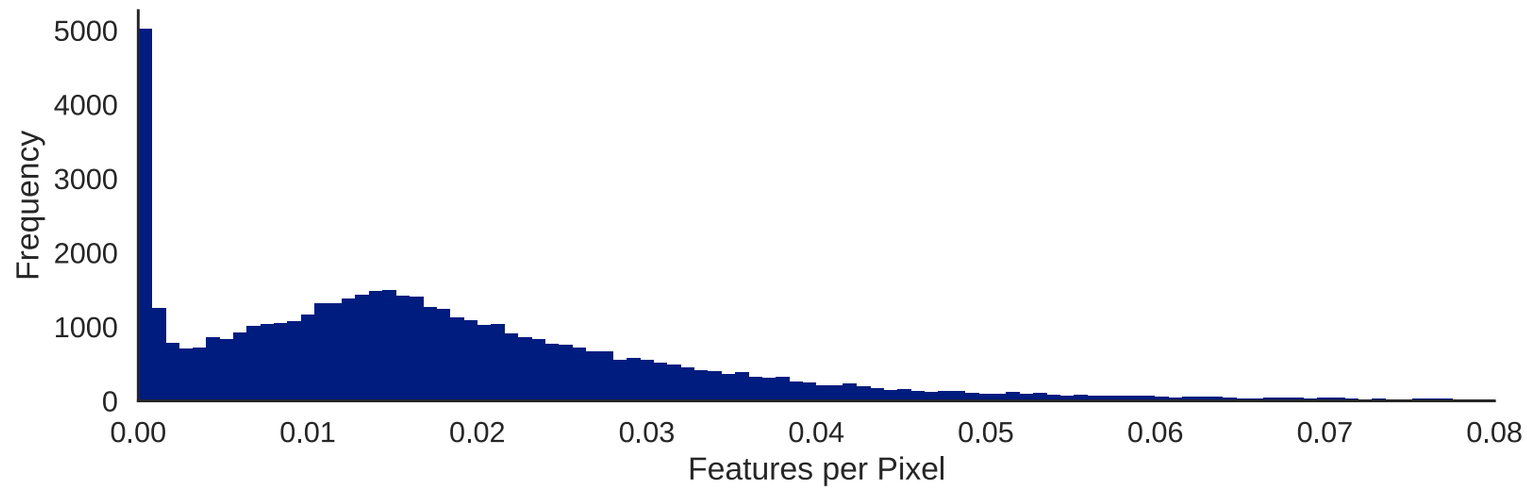
```
# RootSIFT uses L1 norm (absolute value)
descriptors /= np.sqrt(descriptors.sum())
```

Convert to RootSIFT

Arandjelovic, R. and Zisserman, A., *IEEE CCVPR*, 2012
PyImageSearch RootSIFT Discussion
OpenCV2 code has been streamlined for presentation

Detection of Wine Label Features

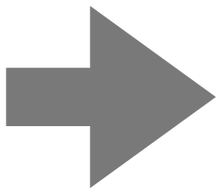




- Image sizes: 100 x 100 to 1000 x 1200
- Features from high resolution images did not encode well

Wine Label Recognition: Bag of Visual Words

1. Scraped ~50,000 Wine Labels
2. Detection of Image Features with SIFT
3. Creation of Bag of Visual Words
4. Image Selection with RANSAC

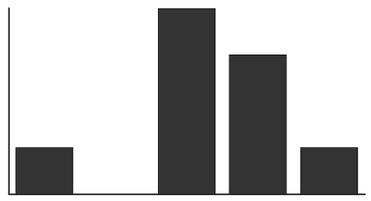


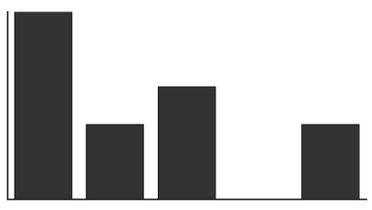
Web Application

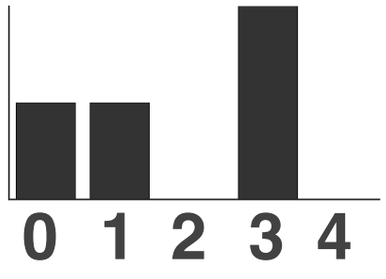
Bag of Visual Words

Wine Label

Cluster Histogram





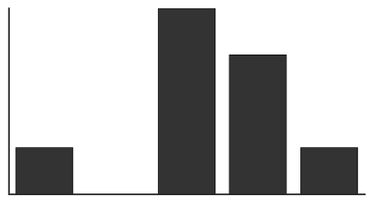


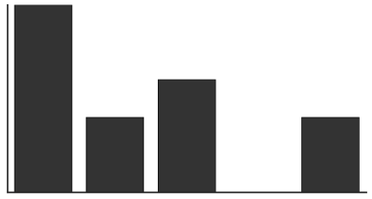
- K-means clustering on combined features from data set
- Map features for each label to nearest cluster
- Creates a histogram "fingerprint" for each label

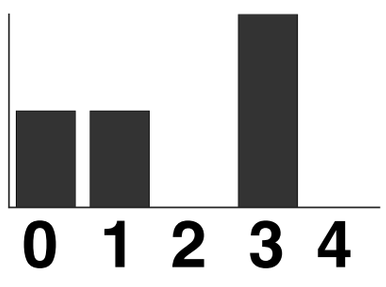
Bag of Visual Words

Wine Label

Cluster Histogram







Inverted Index

Cluster ID

0

1

2

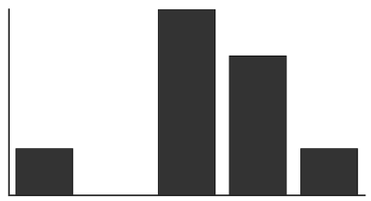
3

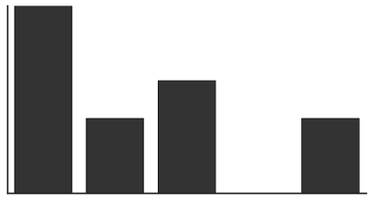
Wine Labels

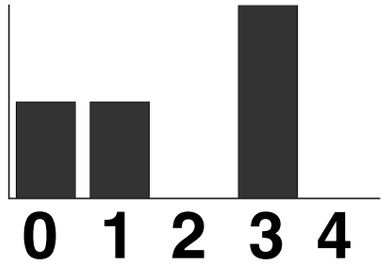
Bag of Visual Words

Wine Label

Cluster Histogram







Inverted Index

Cluster ID

Wine Labels

0

1

2

3

Wine Label Selection

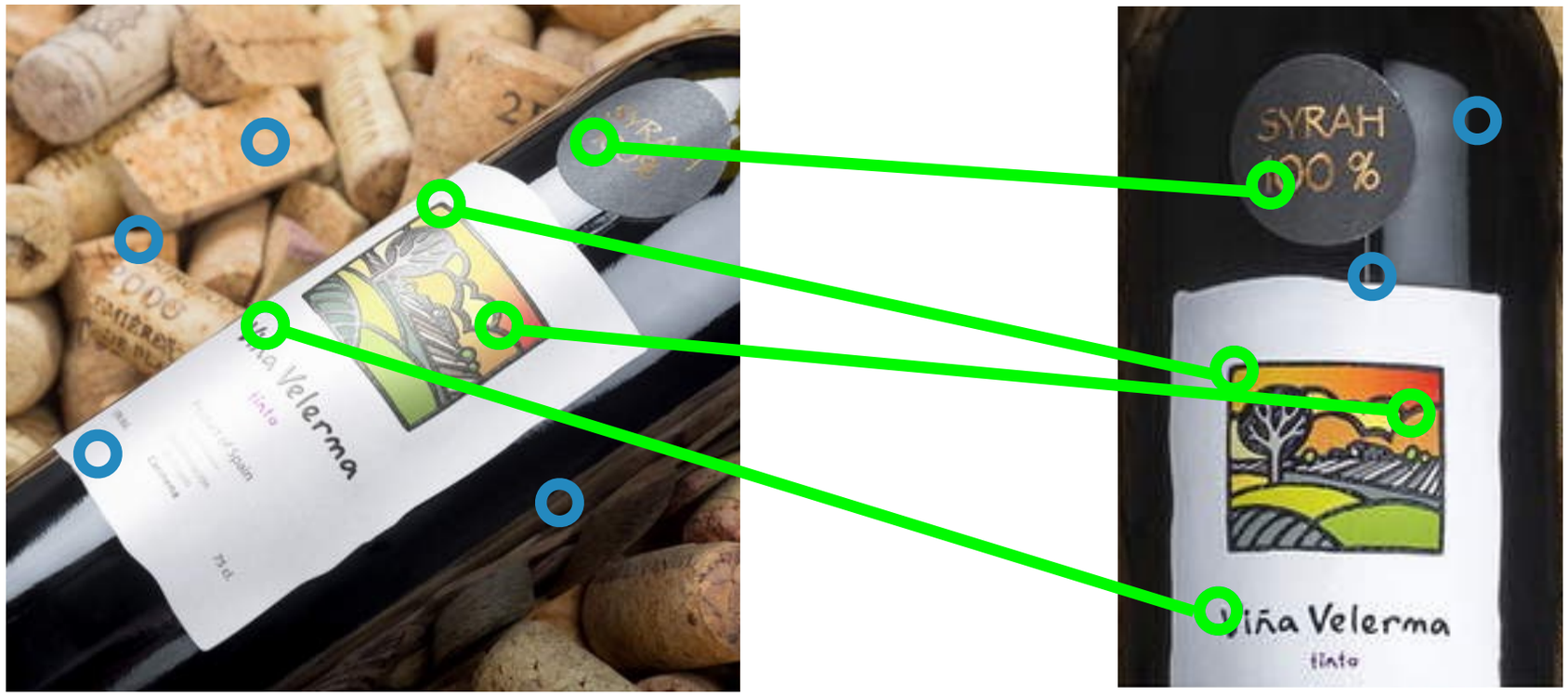
1. Scraped ~50,000 Wine Labels
2. Identification of Image Features with SIFT
3. Creation of Bag of Visual Words
4. Image Selection with RANSAC

→ Web Application

Image Retrieval with RANSAC

- Random sample consensus (RANSAC) used to choose best candidate image

Image Retrieval with RANSAC



- Random sample consensus (RANSAC) used to choose best candidate image
- Random subsets of data fit to model

Running RANSAC

```
# keypointsA, featuresA from query image
# keypointsB, featuresB from candidate image

# Pair similar features from each image
d_matcher = cv2.DescriptorMatcher_create('BruteForce')
matches = d_matcher.knnMatch(featuresB,
                             featuresA, 2)

# Select matched keypoints
match_kpsA = keypointsA[matches[0]]
match_kpsB = keypointsB[matches[1]]

# Run RANSAC to calculate transformation matrix
matrix, status = cv2.findHomography(match_kpsA,
                                    match_kpsB,
                                    cv2.RANSAC,
                                    4.0)
```

Features from Comparison Images

Brute Force Feature Matching

Find Projection

OpenCV2 code has been streamlined for presentation

Putting It All Together

```
# Run SIFT & calculate histogram on query image
keypointsA, featuresA = run_SIFT(image)
histogramA = map_features_to_clusters(featuresA)

# Get candidate images with similar histograms
candidate_images = get_similar_images(histogramA)

for cimage in candidate_images:

    # Load keypointsB, featuresB for cimage

    # Run RANSAC on candidate image
    score = get_ransac_matches(keypointsA,
                               featuresA,
                               keypointsB,
                               featuresB)

# Matched image has best score
```

Calculate Histogram from SIFT Features

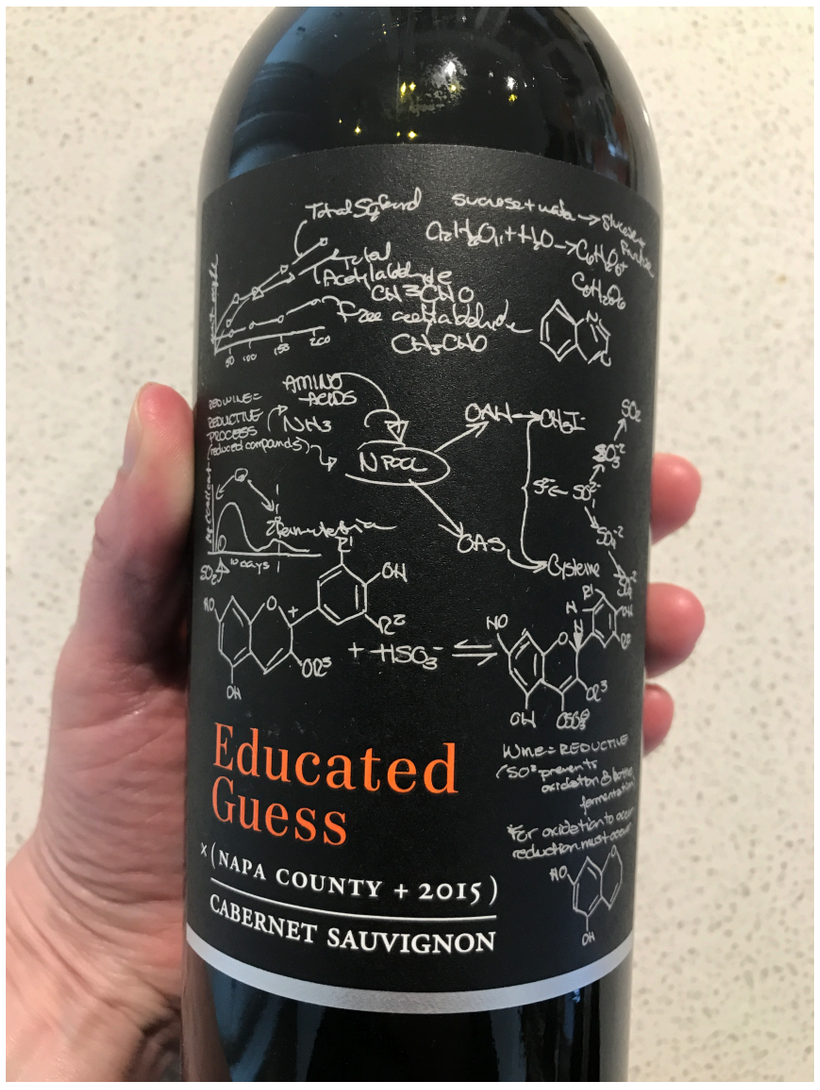
Identify Candidate Images

Use RANSAC to Choose Image Match

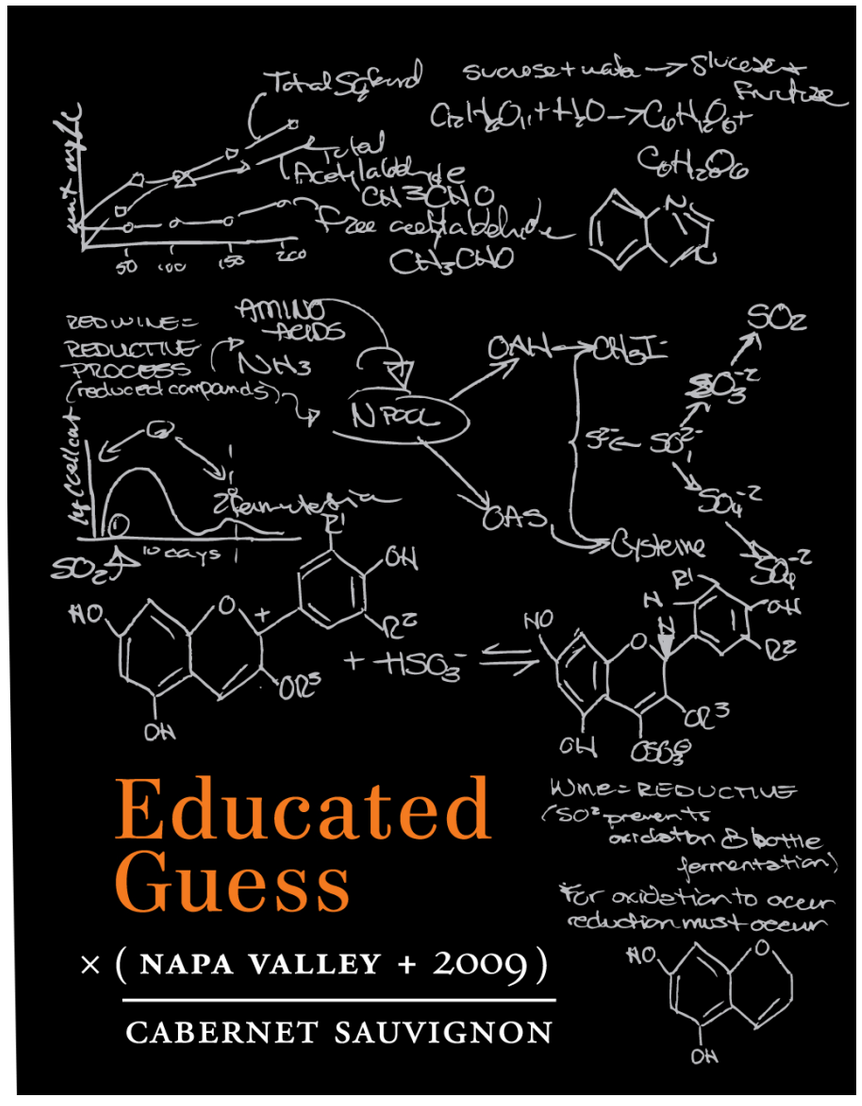
OpenCV2 code has been streamlined for presentation

WINE-O.AI Demonstration

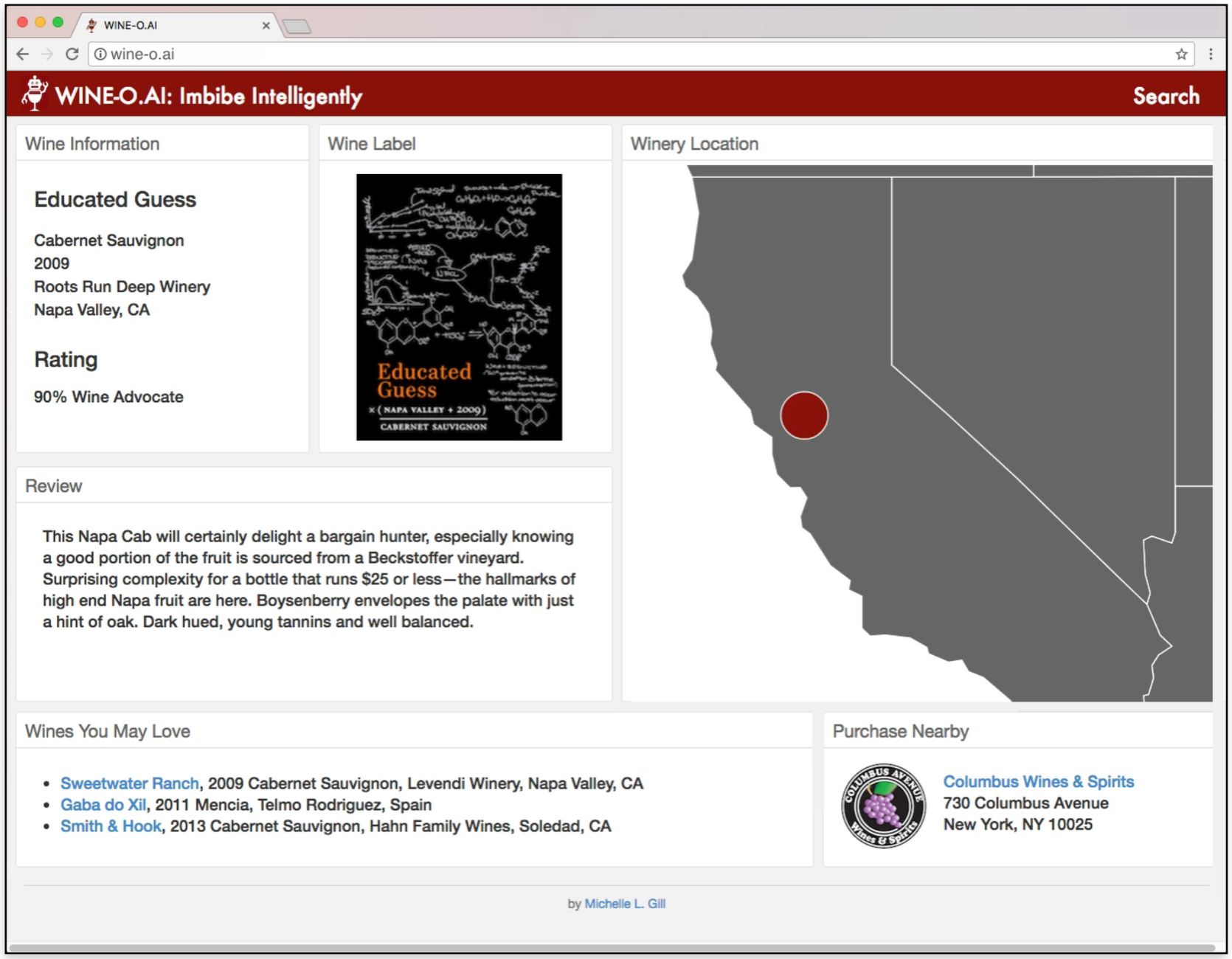
Candidate Image



Database Image



WINE-O.AI Demonstration



Future of WINE-O.AI

- Open source wine application
- Educational use
- GitHub repo: mlgill/wine-o.ai
- Website: wine-o.ai

Thank You

michelle@thisismetis.com

@modernscientist

themodernscientist

mlgill