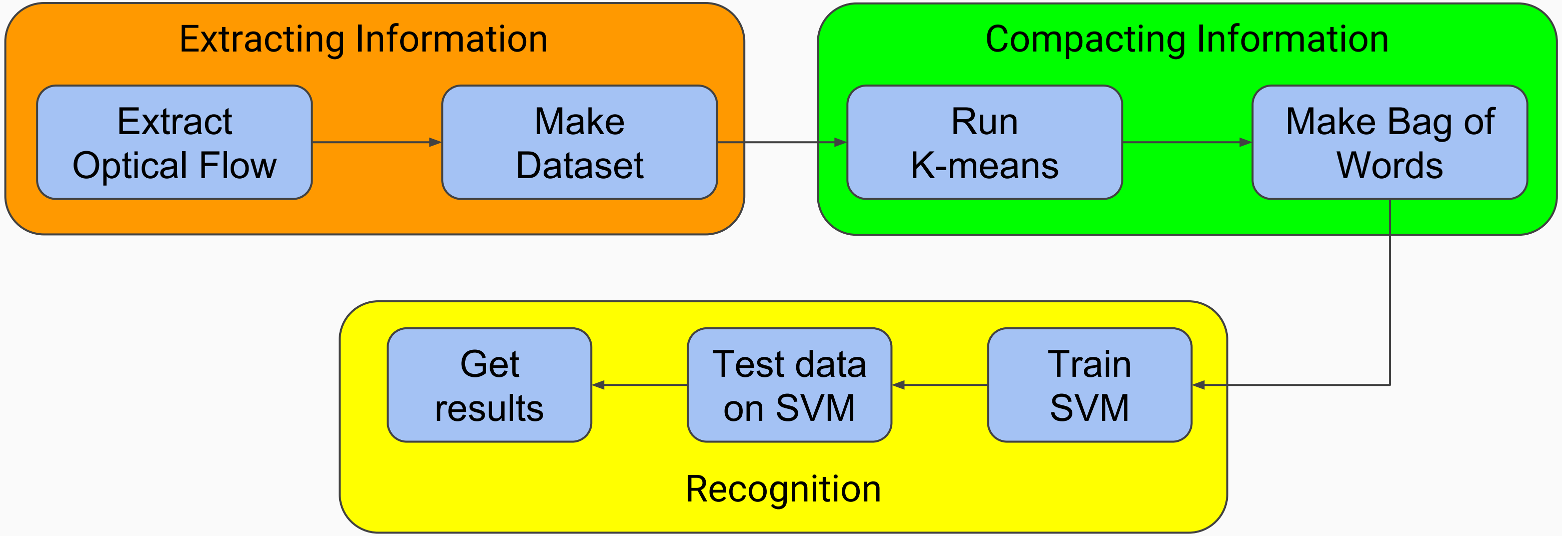
# Performing Human Action Recognition with Optical-Flow, Bag-Of-Words and SVM

Alvaro Rojas, Felipe Moreno & Israel Chaparro

# Pipeline


Extracting Information
Extract Optical Flow
Make Dataset
Compacting Information
Run K-means
Make Bag of Words
Train SVM
Test data on SVM
Get results
Recognition

# Optical flow

Given a current frame and its previous frame, we can compute its optical flow feature using the built-in dense optical flow **Gunnar Farneback's algorithm of OpenCV**. Thus, given a video with N frames, we can compute a set of N-1 optical flow feature descriptors.

As the videos' resolution are 160x120 and we want to save memory, we only sample the optical flow values on the rows and columns whose indices are multiples of 10 (i.e. row and columnn 0, 10, 20, ...). The optical flow descriptor for a frame will have size 16x12x2 = 384 (2 comes from the horizontal and vertical direction).

# Make dataset

Split the computed optical flow figures **using an identifier** of the person in the record generating train and test sets. This generated “train_keypoints.p” (features)

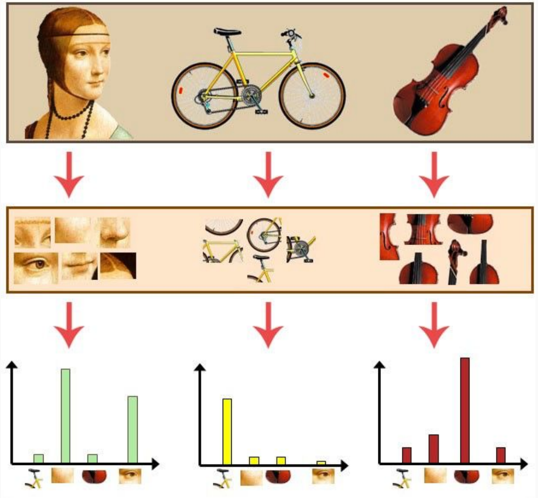
TRAIN_PEOPLE_ID=[11,12,13,14,15,16,17,18,19,20,21,23,24,25,1,4] = 383 videos

TEST_PEOPLE_ID=[22, 2, 3, 5, 6, 7, 8, 9, 10] = 216 videos

# Bag of visual words

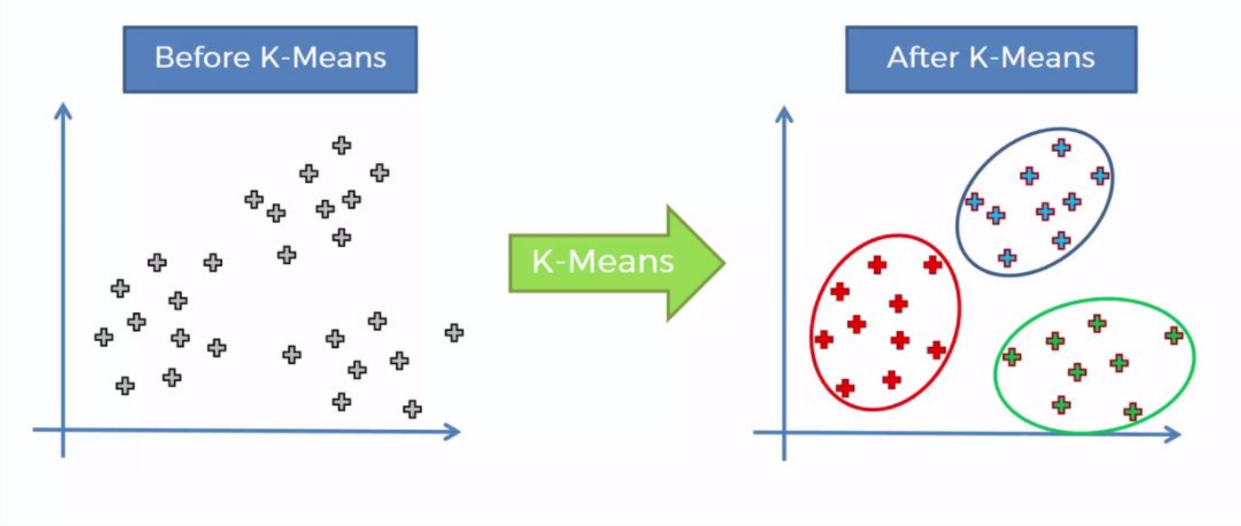
Use a descriptor based on the occurence rate of some particular regions defined by clustering.

This algorithm applies a detector and descriptor to get a list of vectors describing regions and using k-means to define a specific number of regions.



# Running K-means

Run K-means on "train_keypoints.p" (features) with 200 as the number of clusters and produce the codebook.



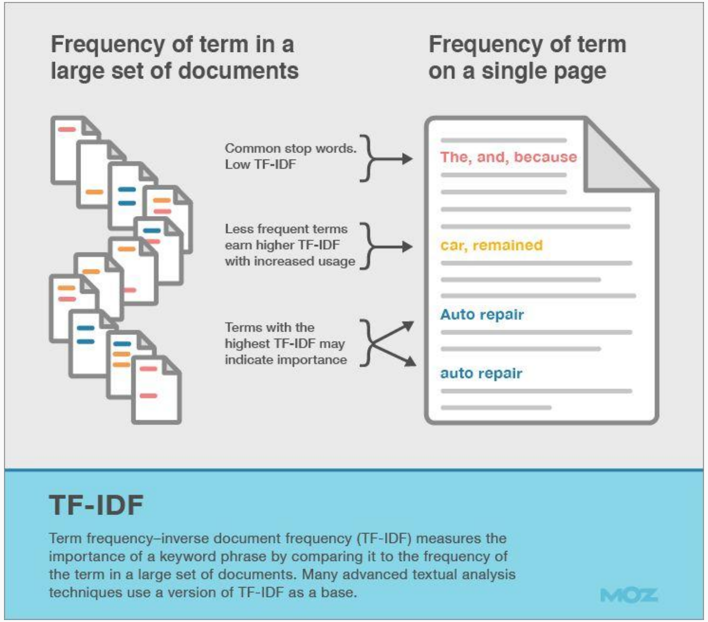
# Build bag of words

Make Bag-Of-Words for every **video** (not frame) in the training and test set, using the computed clusters (K-means) using Term frequency – Inverse document frequency (TF-IDF).

A BoW vector is like a **histogram** that counts the frequency of optical flow descriptors which appear in a video.

TF-IDF: Term frequency – Inverse document frequency.

# Build bag of words




| | Document 1 | Document 2 | Document 3 | Document 4 | Document 5 | Document 6 | Document 7 | Document 8 |
|---|---|---|---|---|---|---|---|---|
| Term(s) 1 | 10 | 0 | 1 | 0 | 0 | 0 | 0 | 2 |
| Term(s) 2 | 0 | 2 | 0 | 0 | 0 | 18 | 0 | 2 |
| Term(s) 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| Term(s) 4 | 6 | 0 | 0 | 4 | 6 | 0 | 0 | 0 |
| Term(s) 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 |
| Term(s) 6 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| Term(s) 7 | 0 | 1 | 8 | 0 | 0 | 0 | 0 | 0 |
| Term(s) 8 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 |

Word Vector (Passage Vector)

Document Vector

# Train SVM

We then train a linear SVM classifier on BoW vectors of training set. As the number of videos in our training set is only 383, the training process takes less than a second, which is a lot faster than the method of considering each individual frame as an instance.

C=1

kernel=”linear”

# Evaluate SVM

Use computed SVM classifier to classify videos in train and test set and get the accuracy result.

# RESULTS

171/216 Correct

Accuracy = 0.79166

Confusion Matrix:

| | BOX | CLAP | WAVE | JOG | RUN | WALK |
|---|---|---|---|---|---|---|
| BOX | 32 | 5 | 3 | 0 | 0 | 0 |
| CLAP | 2 | 31 | 4 | 0 | 0 | 1 |
| WAVE | 0 | 0 | 29 | 0 | 0 | 0 |
| JOG | 0 | 0 | 0 | 22 | 5 | 7 |
| RUN | 1 | 0 | 0 | 9 | 29 | 0 |
| WALK | 1 | 0 | 0 | 5 | 2 | 28 |
| ACCU | 88.8% | 86.1% | 80.5% | 61.1% | 80.5% | 77.7% |