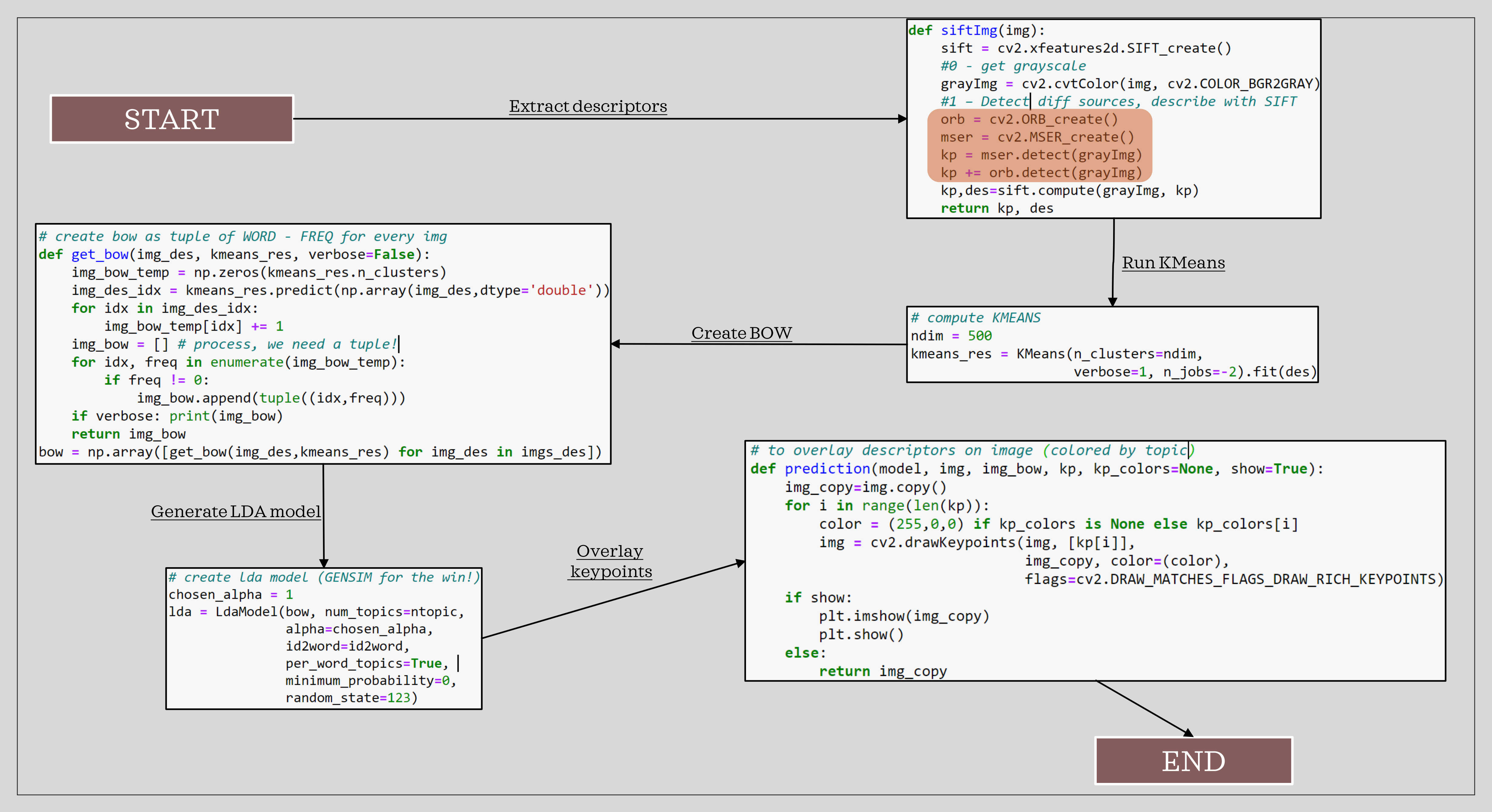
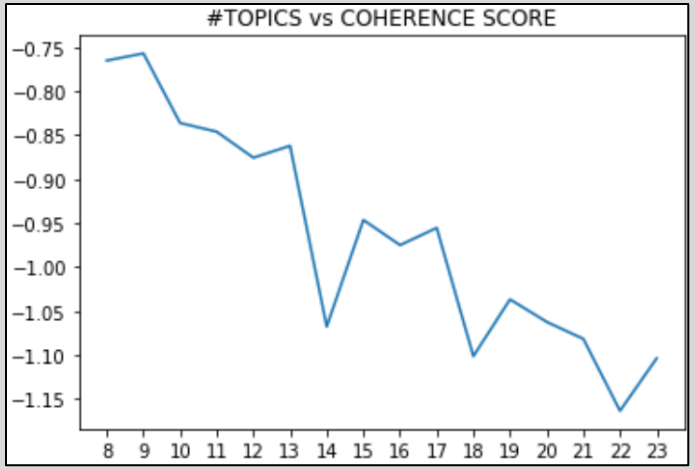
TOPIC MODELING WITH LDA

Assignment 2
Nicola Gugole
ID: 619625


START
Extract descriptors
def siftImg(img):
    sift = cv2.xfeatures2d.SIFT_create()
    #0 - get grayscale
    grayImg = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    #1 - Detect diff sources, describe with SIFT
    orb = cv2.ORB_create()
    mser = cv2.MSER_create()
    kp = mser.detect(grayImg)
    kp += orb.detect(grayImg)
    kp,des=sift.compute(grayImg, kp)
    return kp, des
Run KMeans
# compute KMEANS
ndim = 500
kmeans_res = KMeans(n_clusters=ndim,
                    verbose=1, n_jobs=-2).fit(des)
Create BOW
# create bow as tuple of WORD - FREQ for every img
def get_bow(img_des, kmeans_res, verbose=False):
    img_bow_temp = np.zeros(kmeans_res.n_clusters)
    img_des_idx = kmeans_res.predict(np.array(img_des,dtype='double'))
    for idx in img_des_idx:
        img_bow_temp[idx] += 1
    img_bow = [] # process, we need a tuple!
    for idx, freq in enumerate(img_bow_temp):
        if freq != 0:
            img_bow.append(tuple((idx,freq)))
    if verbose: print(img_bow)
    return img_bow
bow = np.array([get_bow(img_des,kmeans_res) for img_des in imgs_des])
Generate LDA model
# create lda model (GENSIM for the win!)
chosen_alpha = 1
lda = LdaModel(bow, num_topics=ntopic,
               alpha=chosen_alpha,
               id2word=id2word,
               per_word_topics=True,
               minimum_probability=0,
               random_state=123)
Overlay keypoints
# to overlay descriptors on image (colored by topic)
def prediction(model, img, img_bow, kp, kp_colors=None, show=True):
    img_copy=img.copy()
    for i in range(len(kp)):
        color = (255,0,0) if kp_colors is None else kp_colors[i]
        img = cv2.drawKeypoints(img, [kp[i]],
                                img_copy, color=(color),
                                flags=cv2.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS)
    if show:
        plt.imshow(img_copy)
        plt.show()
    else:
        return img_copy
END

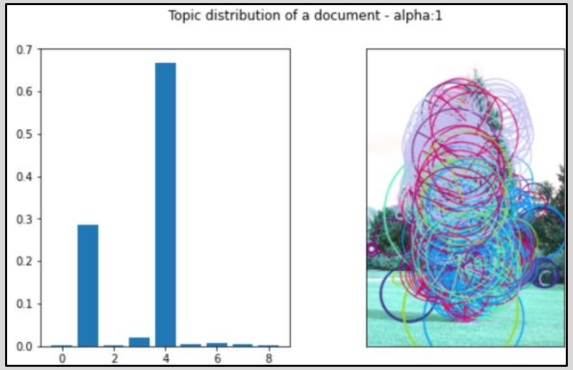
Choosing # TOPICS

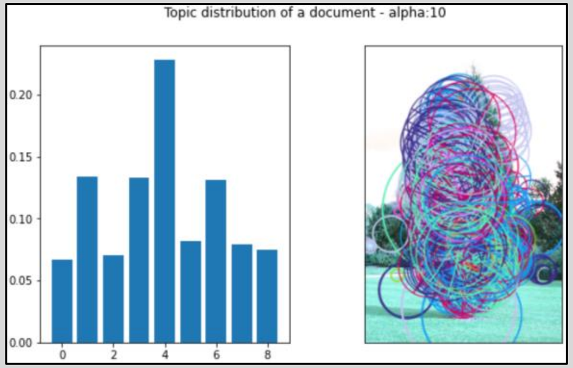


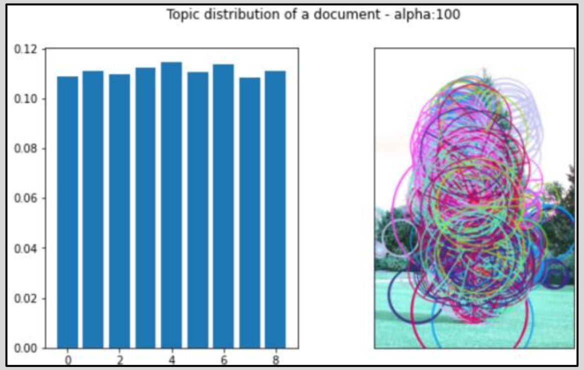
$$coherence(V) = \sum_{(v_i, v_j) \in V} score(v_i, v_j, \epsilon)$$

Choose by HIGHEST coherence score (**umass** in the case of images)

Effect of DIFFERENT ALPHAS

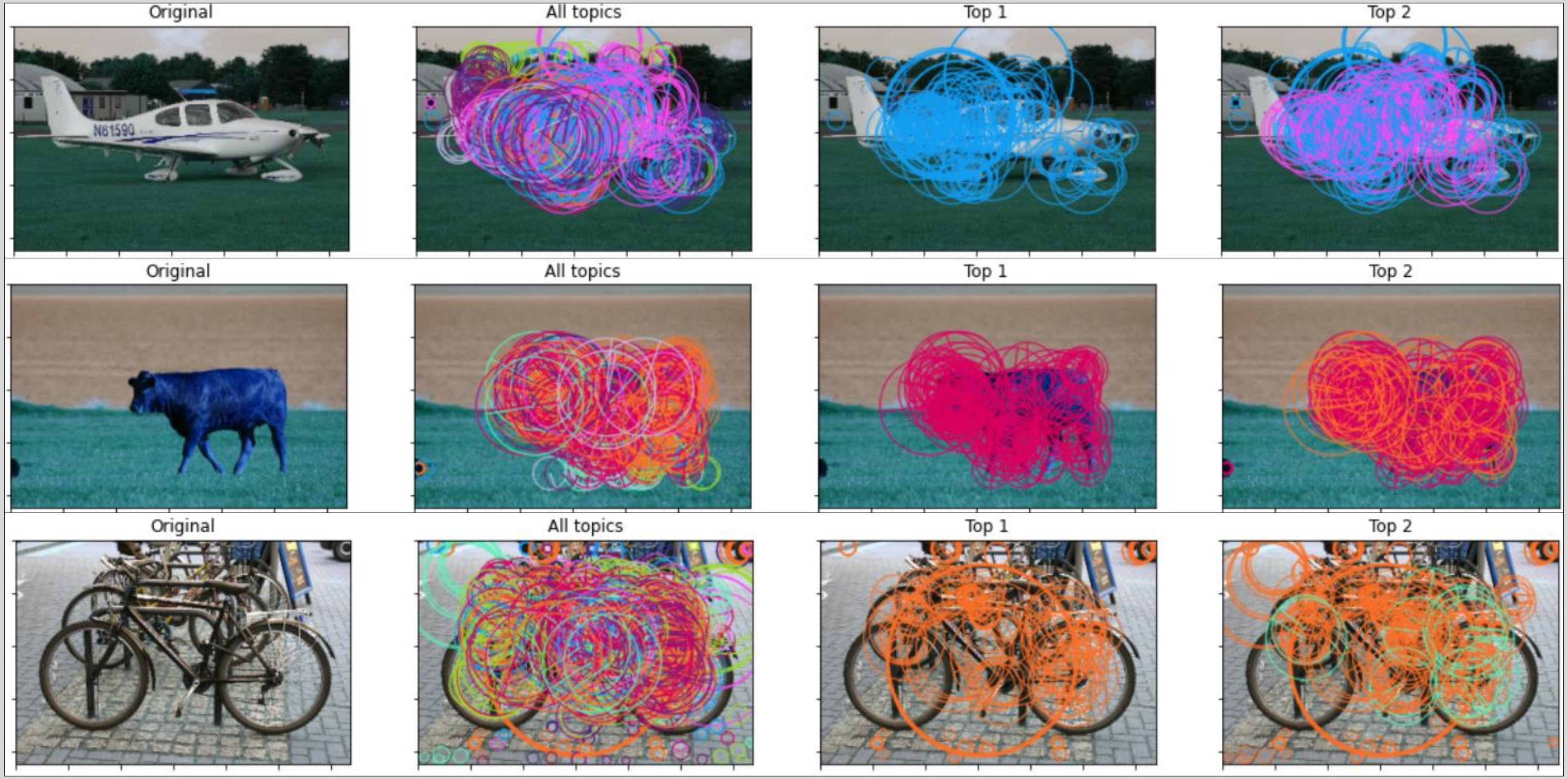






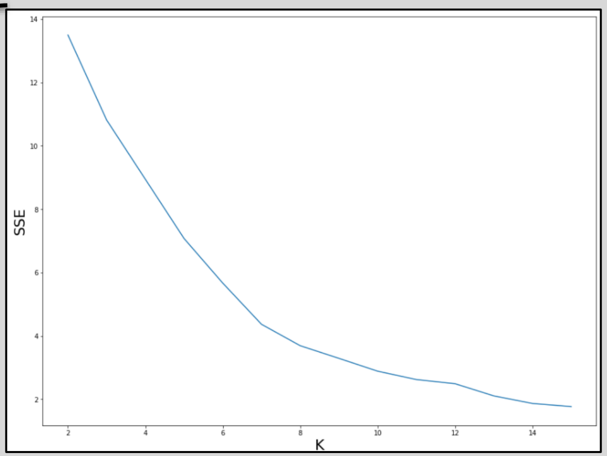
Higher **alpha** implies a more balanced topic mixture

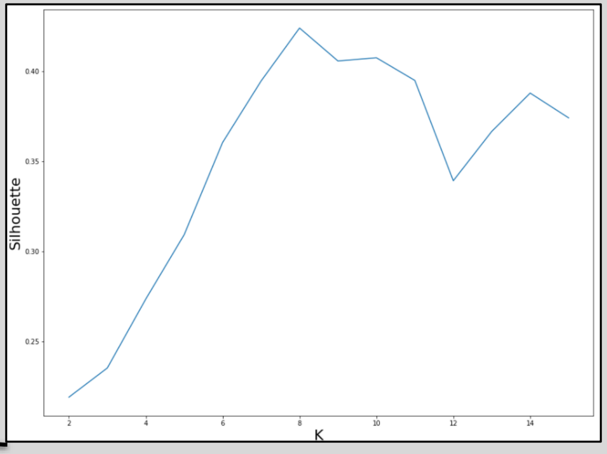
APPLYING TOPICS TO TEST IMAGES



CLUSTERING BY TOPIC DISTRIBUTION

How many? Usual way..



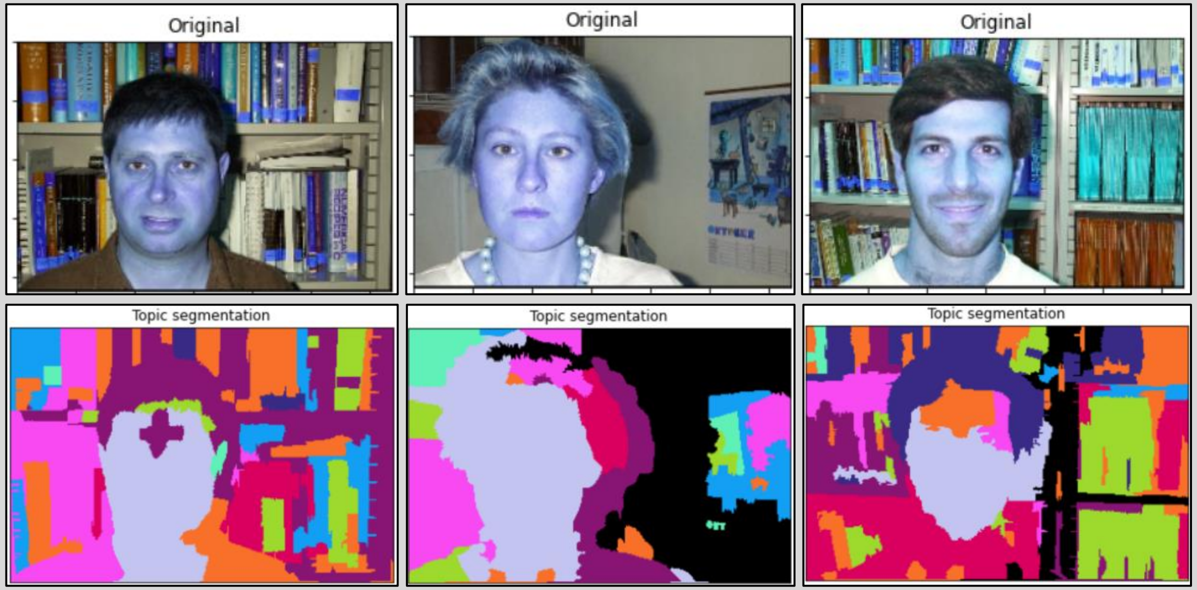


Example of members for a cluster

BETTER VISUAL: TOPIC SEGMENTATION

SKETCHY BUT SHOWS COHERENCE

- Method has potential, but not even remotely perfect
- **Dataset** is limited and dispersive
- Implemented methodology is naive
- **Spatial information** would help the model
- **Spatial LDA** and **Topic Random Fields** go in that direction



THANK YOU FOR YOUR ATTENTION

(FOR CODE CURIOSITIES, HERE)