Machine Learning for Computer Vision

[MVA 2015/2016]

Final assignment

Maha ELBAYAD

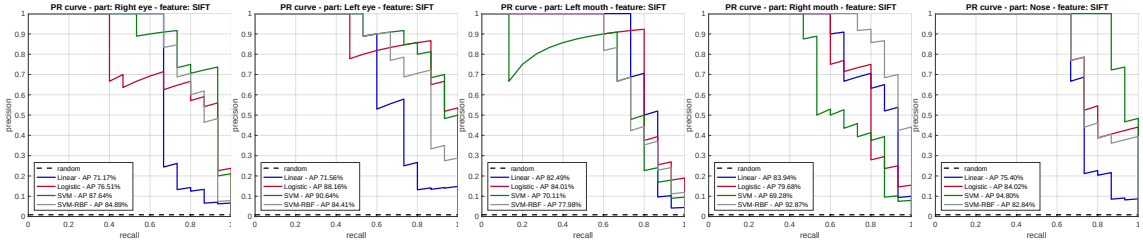
1 Detector training

We trained different classifiers: Linear regression, L2-logistic regression (c.f. assignment 1), linear SVM and RBF-SVM (c.f. assignement 2) to detect particular points of a facial image namely the left eye, right eye, left mouth, right mouth and nose. Each of these classifier is either trained on SIFT features or CNN features extracted with an Alexnet pre-trained network[1] shown in figure (1) where the features are either the outpout of the relu2 layer denoted `CNN2` or that of the relu5 layer denoted `CNN5`.

Figure 1: *Alexnet architecture*

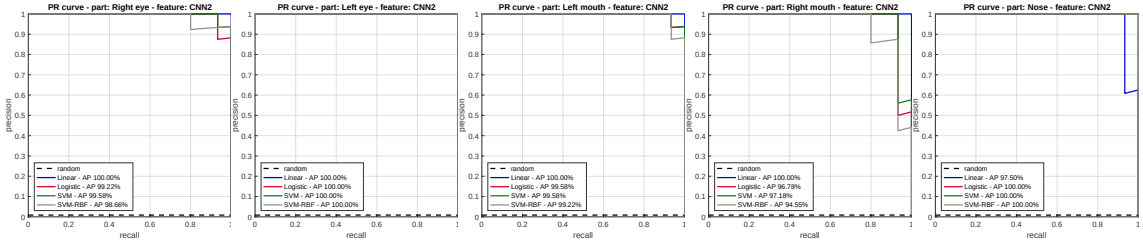
If needed we choose the classifier's optimal parameters with cross-validation (5-folds).

1.1 Precision recall curves:



(a) *Right eye* **(b)** *Left eye* **(c)** *Left mouth* **(d)** *Right mouth* **(e)** *Nose*

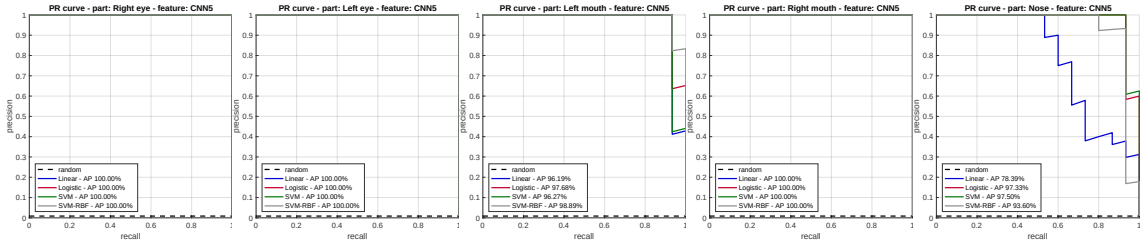
Figure 2: *SIFT features*



(a) *Right eye* **(b)** *Left eye* **(c)** *Left mouth* **(d)** *Right mouth* **(e)** *Nose*

Figure 3: *CNN2 features*

[1] http://www.vlfeat.org/matconvnet/models/beta16/imagenet-caffe-alex.mat

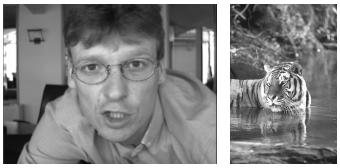


(a) *Right eye* **(b)** *Left eye* **(c)** *Left mouth* **(d)** *Right mouth* **(e)** *Nose*

Figure 4: *CNN5 features*

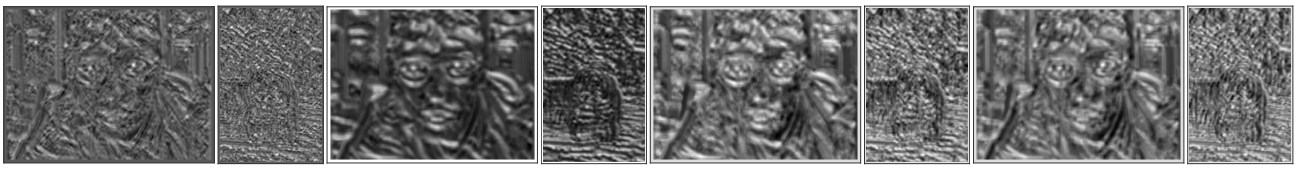
1.2 SIFT features - Dense scores

For two samples from different categories we show the dense scores of each classifier.



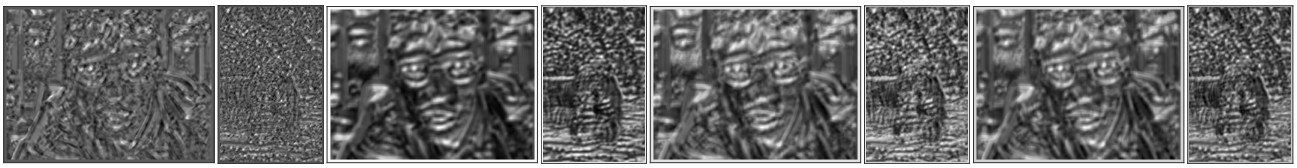
(a) *Face* **(b)** *Background*

Figure 5: *Input images*



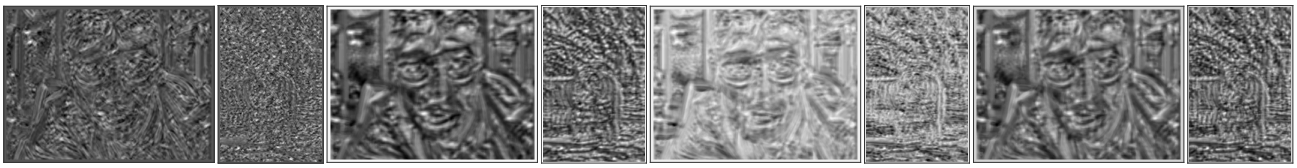
(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

Figure 6: *Right eye*



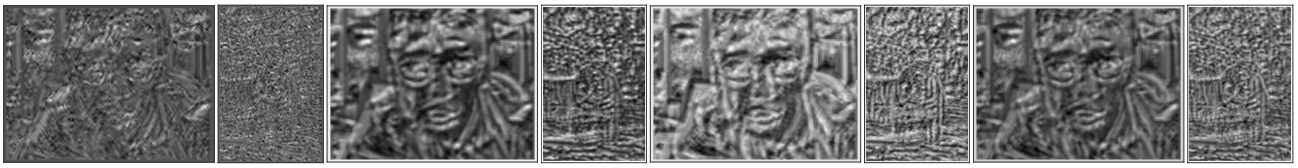
(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

Figure 7: *Left eye*



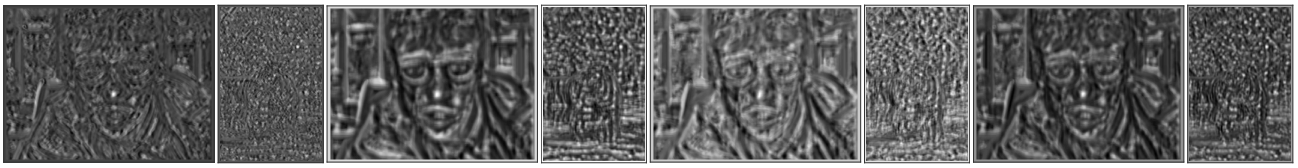
(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

Figure 8: *Left mouth*



(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

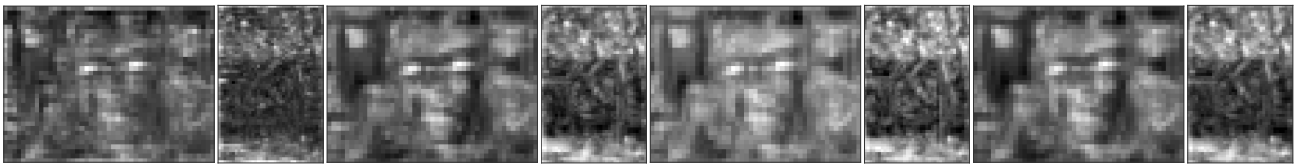
Figure 9: *Right mouth*



(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

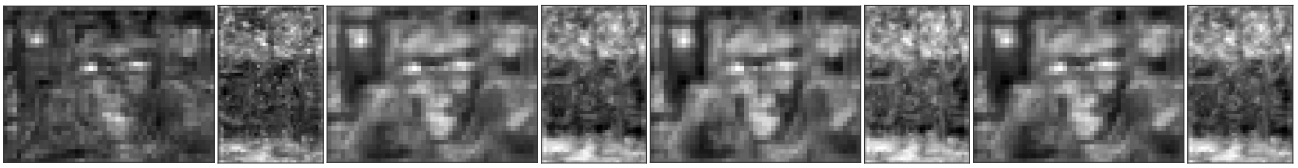
Figure 10: *Nose*

1.3 CNN2 features - Dense scores



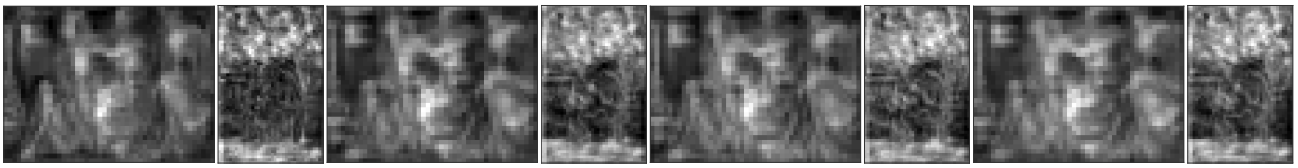
(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

Figure 11: *Right eye*



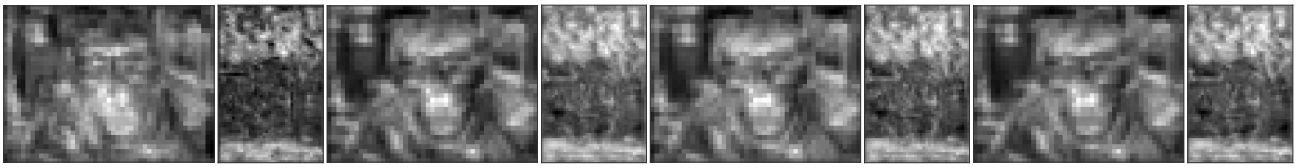
(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

Figure 12: *Left eye*



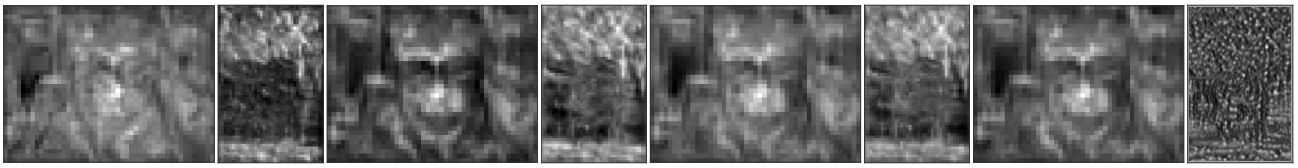
(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

Figure 13: *Left mouth*



(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

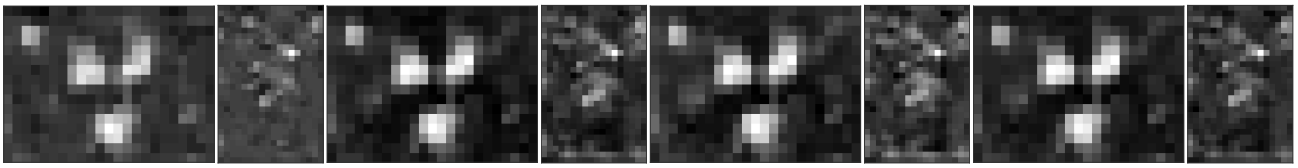
Figure 14: *Right mouth*



(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

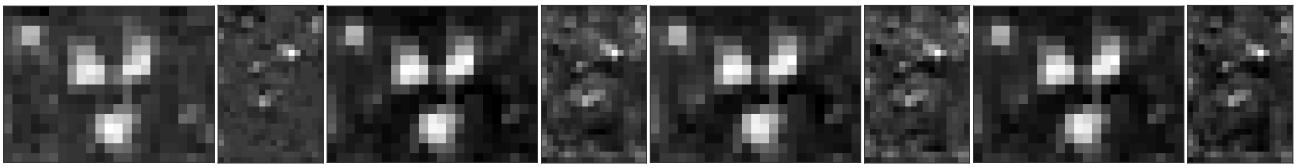
Figure 15: *Nose*

1.4 CNN5 features - Dense scores



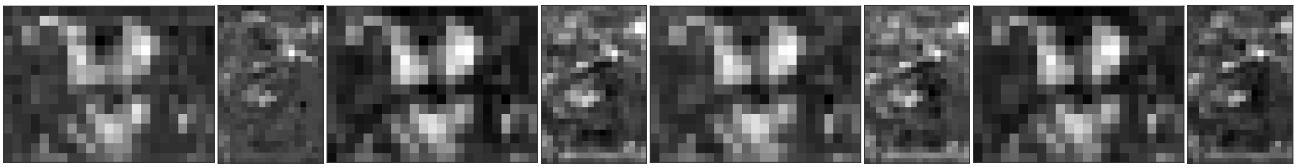
(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

Figure 16: *Right eye*



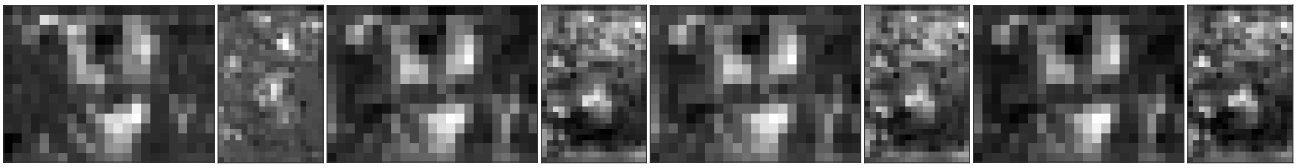
(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

Figure 17: *Left eye*



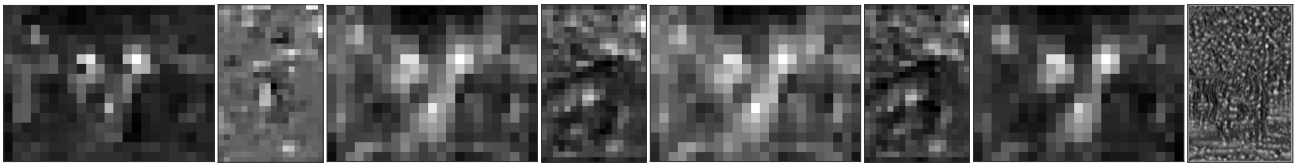
(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

Figure 18: *Left mouth*



(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

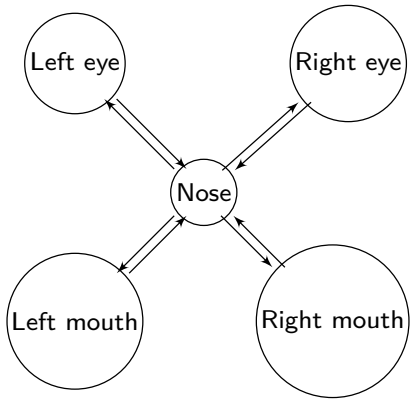
Figure 19: *Right mouth*



(a) *LR (1)* **(b)** *LR (0)* **(c)** *Logistic (1)* **(d)** *Logistic (0)* **(e)** *SVM (1)* **(f)** *SVM (0)* **(g)** *SVM-RBF (1)* **(h)** *SVM-RBF (0)*

Figure 20: *Nose*

2 Combination: Max-product for DPM detection



We consider the probabilsitic model above to which we apply the max-product algorithm. Each leaf node {Left eye, right eye, left mouth, right mouth} (index p) passes a message to the root (index r) of the form:

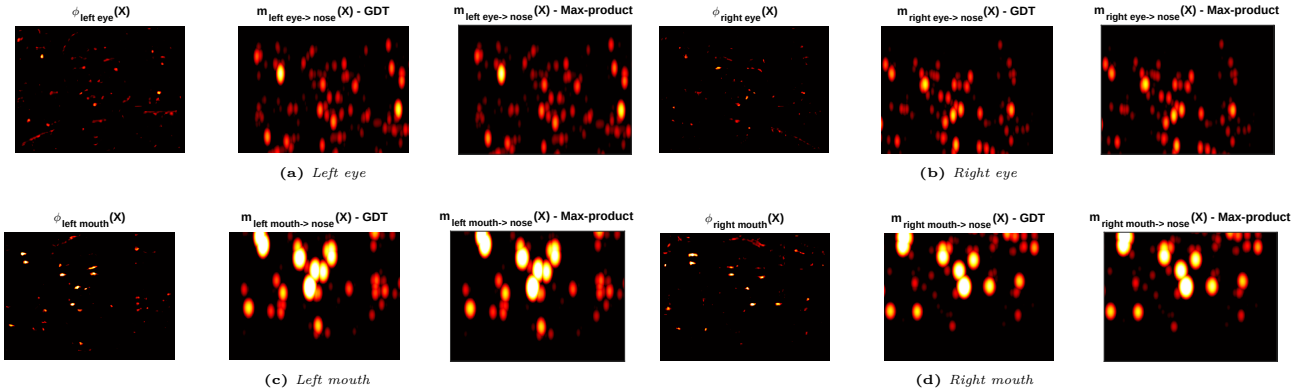
$$\begin{aligned} m_{p\to r}(X) &= \log\left(\max_{X_p} \Phi_p(X_p)\Psi_{p,r}(X_p, X_r)\right) \\ &= \max_{X_p} \phi_p(X_p) + a\left(X_p(1) - X_r(1)\right)^2 + b\left(X_p(1) - X_r(1)\right) + c\left(X_p(2) - X_r(2)\right)^2 + d\left(X_p(2) - X_r(2)\right) + e \end{aligned}$$

where $\phi = \log\Phi, \psi = \log\Psi$ and a, b, c, d and e are parameters characterizing the distribution of the displacement vector $X_P - X_r$ assumed Gaussian.
We have:

$$\begin{cases} a = -\frac{1}{2\sigma_p(1)^2} \\ b = \frac{\mu_p(1)}{\sigma_p(1)^2} \\ c = -\frac{1}{2\sigma_p(2)^2} \\ d = \frac{\mu_p(2)}{\sigma_p(2)^2} \\ e = -\frac{\mu_p(1)^2}{2\sigma_p(1)^2} - \frac{\mu_p(2)^2}{2\sigma_p(2)^2} - \log(2\pi\sigma_1\sigma_2) \end{cases}$$

The unary potentials are computed first with the provided SVM weights using SIFT features and then with an SVM learnt using the CNN2 features with the Alexnet architecture.

2.1 Provided SVM model with SIFT



(a) *Left eye* **(b)** *Right eye* **(c)** *Left mouth* **(d)** *Right mouth*

Figure 21: *Message passing : GDT vs Max-product (image 3)*

At the root we collect the messages to compute the belief function $b(X) = \phi_r(X) + \sum_{i \neq r} m_{i \to r}(X)$

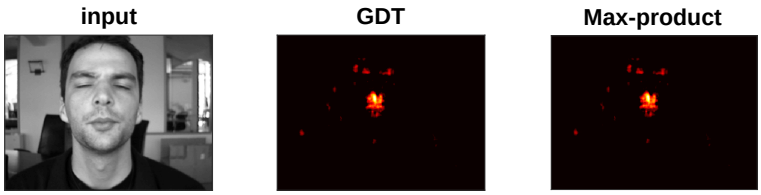
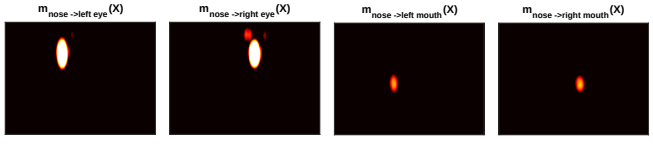


Figure 22: *Belief function at the root (≡ nose) - (image 3)*

In a similar fashion, we send messages back to the parts:

$$m_{r \to p}(X) = \log \left(\max_{X_r} \left[\Phi_r(X_r) \Psi_{r,p}(X_r, X_p) \prod_{k \in \mathcal{N}(r) \setminus p} M_{k \to r}(X_r) \right] \right)$$

$$= \max_{X_r} \left[\phi_r(X_r) + \psi_{r,p}(X_r, X_p) + \sum_{k \in \mathcal{N}(r) \setminus p} m_{k \to r}(X_r) \right]$$

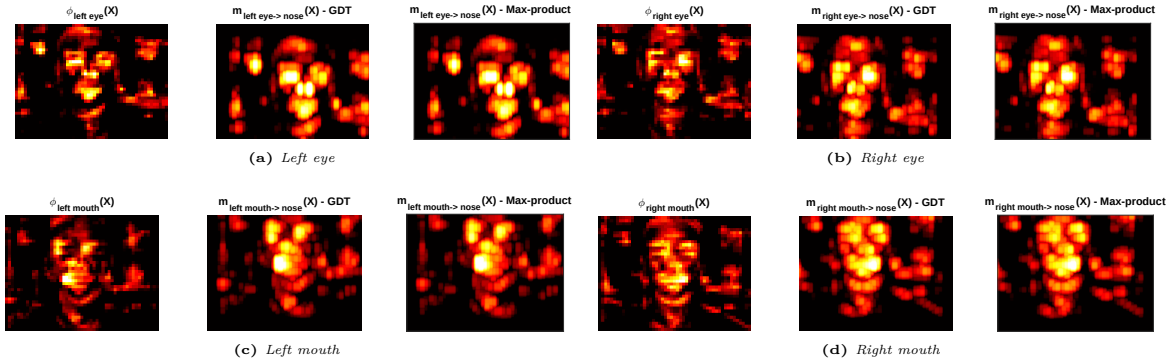


(a) *Left eye* **(b)** *Right eye* **(c)** *Left mouth* **(d)** *Right mouth*

Figure 23: *Messages to the parts (image 3)*

2.2 Part 1' SVM model with CNN2

Image 3:



(a) *Left eye* **(b)** *Right eye*

(c) *Left mouth* **(d)** *Right mouth*

Figure 24: *Message passing : GDT vs Max-product (image 3)*

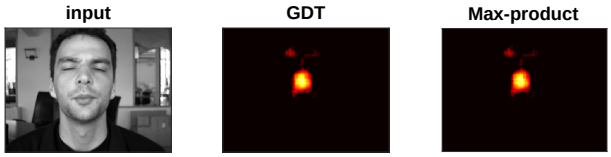
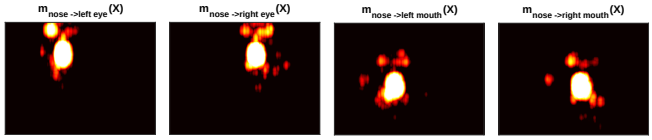


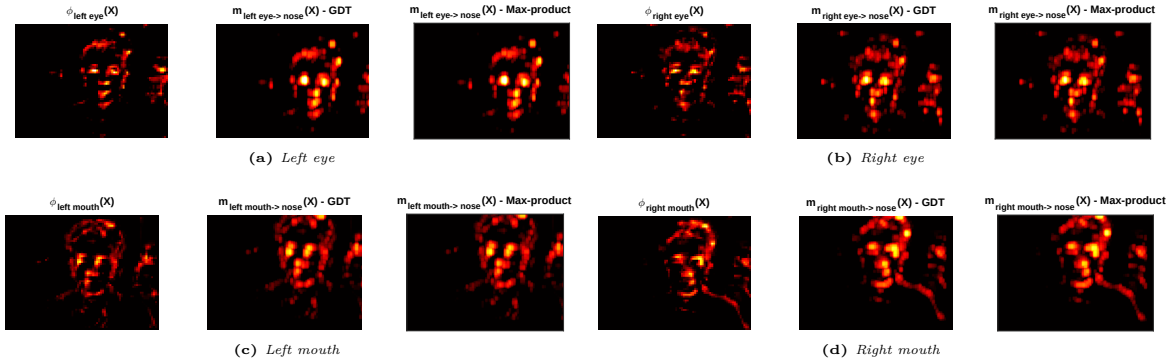
Figure 25: *Belief function at the root (≡ nose) - (image 3)*



(a) *Left eye* **(b)** *Right eye* **(c)** *Left mouth* **(d)** *Right mouth*

Figure 26: *Messages to the parts (image 3)*

Image 231:



(a) *Left eye* **(b)** *Right eye*

(c) *Left mouth* **(d)** *Right mouth*

Figure 27: *Message passing : GDT vs Max-product (image 231)*

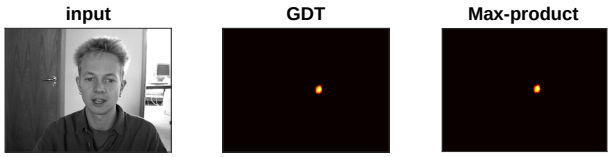
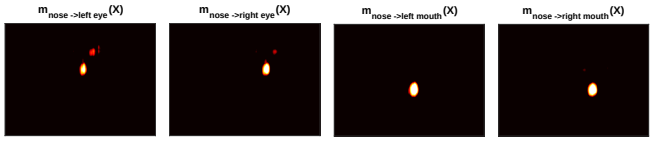


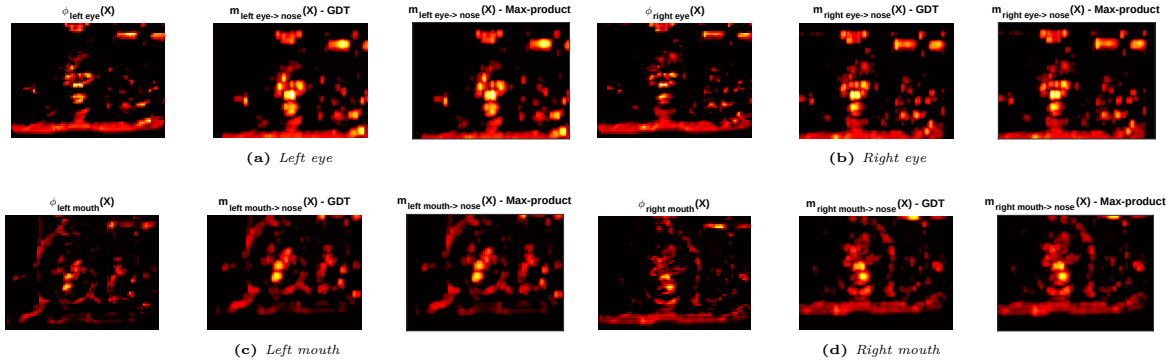
Figure 28: *Belief function at the root (≡ nose) - (image 231)*



(a) *Left eye* **(b)** *Right eye* **(c)** *Left mouth* **(d)** *Right mouth*

Figure 29: *Messages to the parts (image 231)*

Image 507:



(a) *Left eye* **(b)** *Right eye* **(c)** *Left mouth* **(d)** *Right mouth*

Figure 30: *Message passing : GDT vs Max-product (image 507)*

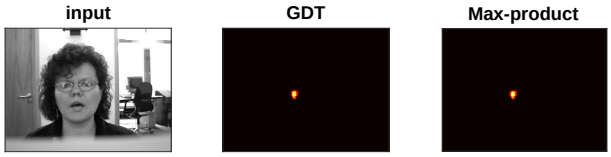
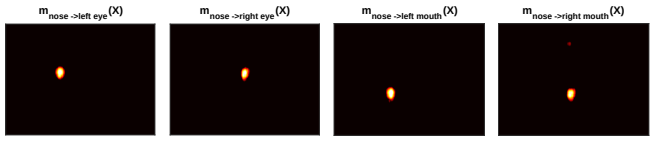


Figure 31: *Belief function at the root (≡ nose) - (image 507)*



(a) *Left eye* **(b)** *Right eye* **(c)** *Left mouth* **(d)** *Right mouth*

Figure 32: *Messages to the parts (image 507)*

3 Multi-scale detection , non-maximum suppression, benchmarking

We perform feature extraction with the previously trained SVM on the CNN2 features then compute the unary scoes and run GDT to compute the belief at the root at different scales $[2^{-1}, 2^{-0.5}, 1, 2^{0.5}, 2]$. The input image remains unnormalized i.e. we

do not correct the distance between the eyes. The candidate roots are selected when the belief is above a selection threshold and the remaining parts are located using the ix and iy maps outputted by GDT given the root location.

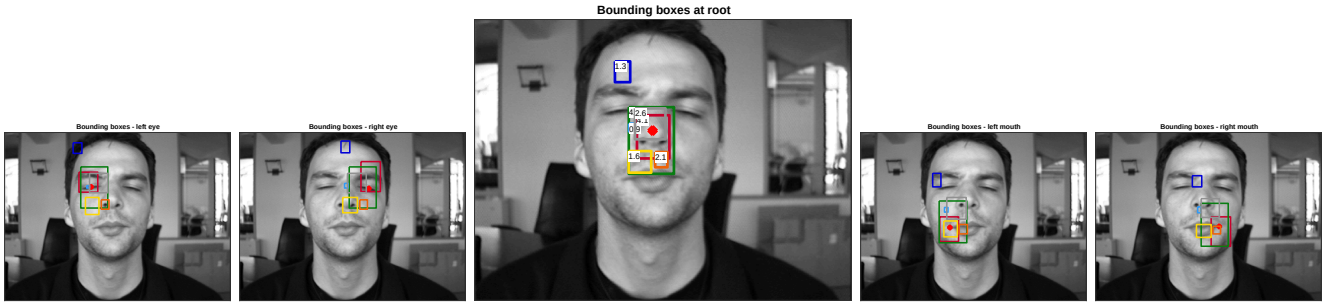


Figure 33: *Threshold = 0.5 - image 3*

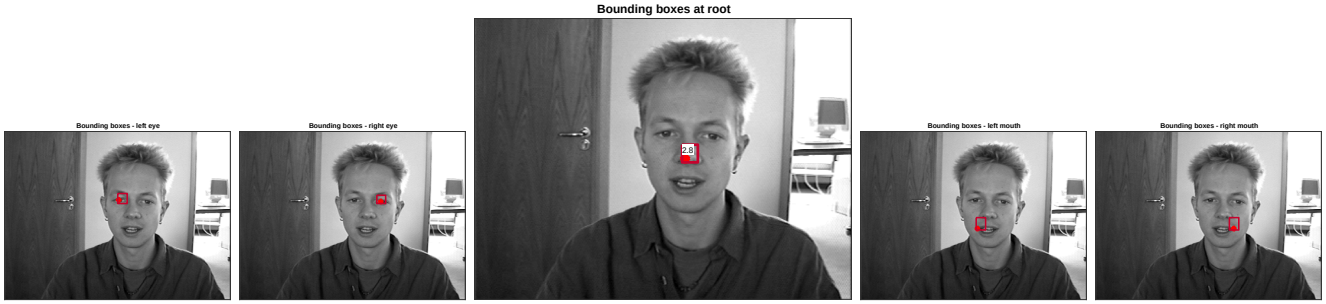


Figure 34: *Threshold = 0.5 - image 231*

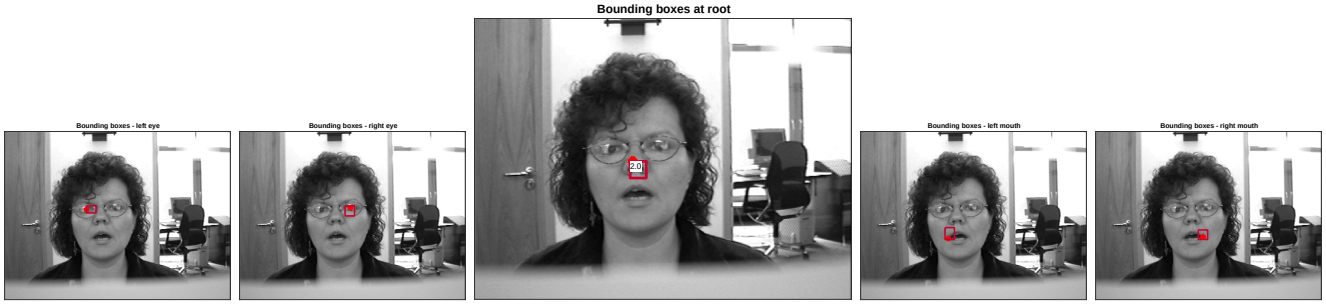
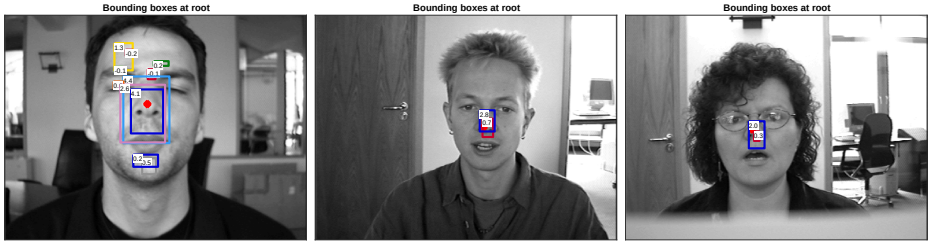
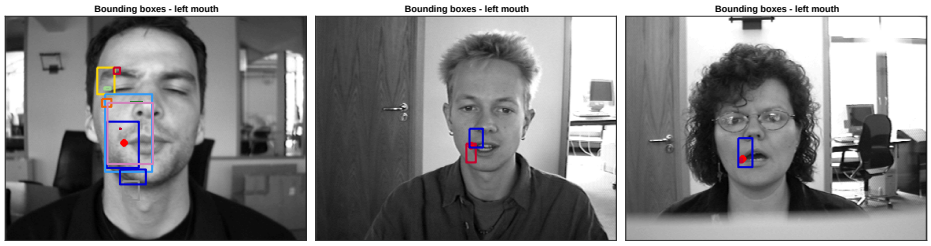


Figure 35: *Threshold = 0.5 - image 507*

With a lower threshold we can suggest more locations:



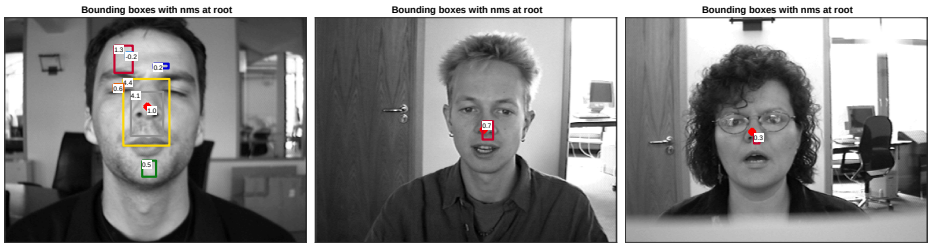
(a) *image 3* **(b)** *image 231* **(c)** *image 507*



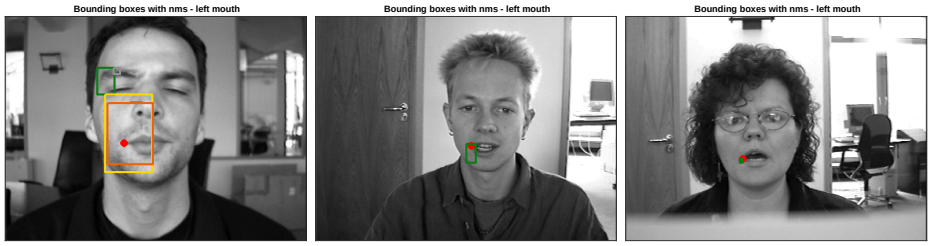
(d) *image 3* **(e)** *image 231* **(f)** *image 507*

Figure 36: *Threshold = -0.5*

To filter the suggested boxes we apply non-maximum suppression with an overlap threshold of 0.7



(a) *image 3* **(b)** *image 231* **(c)** *image 507*



(d) *image 3* **(e)** *image 231* **(f)** *image 507*

Figure 37: *NMS - box selection thresold -0.5 - box overlap threshold 0.7*