Modeling Perceptual Color Differences by Local Metric Learning

A. Habrard
Joint work with M. Perrot, D. Muselet and M. Sebban

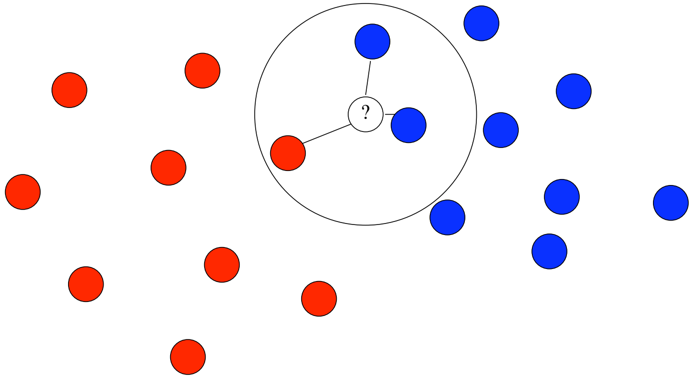
Hubert Curien lab, UMR CNRS 5516
University of Jean Monnet Saint-Étienne (France)

Workshop on machine learning-assisted image formation, Nice, 2019

Importance of metrics

Pairwise metric

The notion of metric plays an important role in many domains such as classification, regression, clustering, ranking, etc.

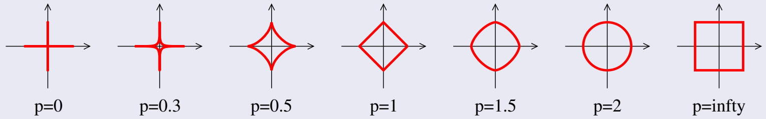


Minkowski distances: family of distances induced by ℓ_p norms over $\mathbb{R}^d$

$$d_p(\mathbf{x},\mathbf{x}') = \|\mathbf{x}-\mathbf{x}'\|_p = \left(\sum_{i=1}^{d} |x_i - x'_i|^p\right)^{1/p}.$$

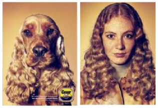
- $p = 1$: **Manhattan distance** $d_1(\mathbf{x},\mathbf{x}') = \sum_{i=1}^{d} |x_i - x'_i|$.
- $p = 2$: **Euclidean distance**

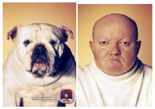
  $$d_2(\mathbf{x},\mathbf{x}') = \left(\sum_{i=1}^{d} |x_i - x'_i|^2\right)^{1/2} = \sqrt{(\mathbf{x}-\mathbf{x}')^T(\mathbf{x}-\mathbf{x}')}.$$
- For $p = \infty$, The Chebyshev distance $d_\infty(\mathbf{x},\mathbf{x}') = \max_i |x_i - x'_i|$
- Minkowski distances - unit circles for various values of p :



Choosing the right metric?

Choosing the right metric?



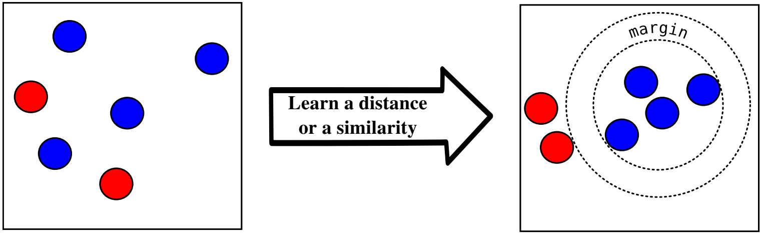


Each problem has its own notion of similarity, which is often badly captured by standard metrics.

Metric Learning: Adapt the metric to the problem of interest

Learning the metric from data

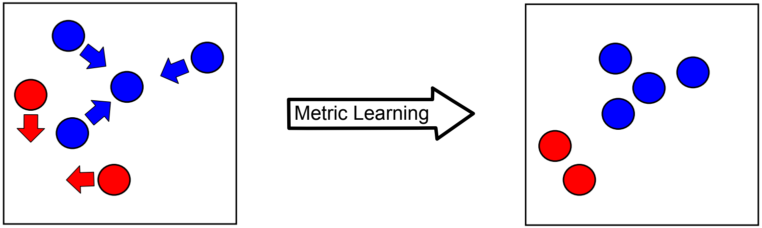
Basic idea: learn a metric that assigns small (resp. large) distance to pairs of examples that are semantically similar (resp. dissimilar).



It typically **induces a change of representation space** which satisfies the semantic constraints.

From a classification standpoint, we aim at moving closer examples of the same class while putting far away examples of different classes.

Metric Learning - Intuition



We need to define some contraints to satisfy between examples:

- Must-link constraints: examples must be close
- Cannot-link constraints: examples must be far away
- Relative: $\mathbf{x}$ is closer to $\mathbf{y}$ than $\mathbf{z}$

"Learnable" metric - "Mahalanobis" distance

Mahalanobis (pseudo) distance

- For any $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^d$, it is defined as:

$$d_{\mathbf{M}}(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{x} - \mathbf{x}')^T \mathbf{M} (\mathbf{x} - \mathbf{x}')}$$

  where $\mathbf{M} \in \mathbb{S}_+^d$ is symmetric Positive Semi-Definite $d \times d$ matrix
  This implies that $\mathbf{M} = \mathbf{L}^T \mathbf{L}$ for some matrix $\mathbf{L}$.
  When $\mathbf{M} = \mathbf{I}$ you recover the Euclidean Distance
- Equivalent to Euclidean distance after linear projection wrt $\mathbf{L}$

$$d_{\mathbf{M}}(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{x} - \mathbf{x}')^T \mathbf{L}^T \mathbf{L} (\mathbf{x} - \mathbf{x}')} = \sqrt{(\mathbf{L}\mathbf{x} - \mathbf{L}\mathbf{x}')^T (\mathbf{L}\mathbf{x} - \mathbf{L}\mathbf{x}')}$$

- If $\mathbf{M}$ has rank $k \leq d$, then $\mathbf{L} \in \mathbb{R}^{k \times d}$ reduces data dimension
- Work with the squared distance for convenience

Metric Learning - Basic recipe

1. Choose a parameterized distance or similarity function
   The function is parameterized by a set of parameters $\mathbf{M}$
2. Collect semantic constraints from data pairs/triplets
   - $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{x}_j) : \mathbf{x}_i$ and $\mathbf{x}_j$ are similar$\}$
   - $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{x}_j) : \mathbf{x}_i$ and $\mathbf{x}_j$ are dissimilar$\}$
   - $\mathcal{R} = \{(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) : \mathbf{x}_i$ is more similar to $\mathbf{x}_j$ than to $\mathbf{x}_k\}$
3. Estimate the parameters $\mathbf{M}$ s.t. the metric best fulfills the semantic constraints
   $\rightarrow$ Solve an optimization problem of the form

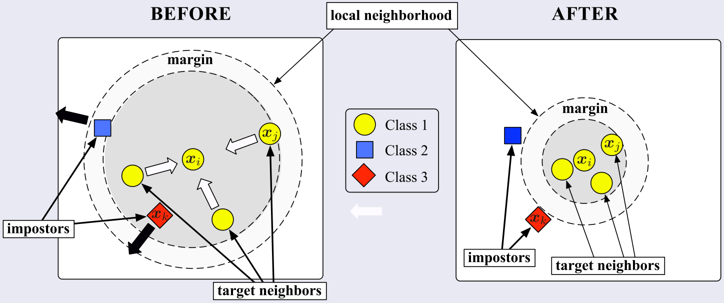
$$\hat{\mathbf{M}} = \arg\min_{\mathbf{M}} \left[\underbrace{\ell(\mathbf{M}, \mathcal{S}, \mathcal{D}, \mathcal{R})}_{\text{loss function}} + \underbrace{\lambda reg(\mathbf{M})}_{\text{regularization}} \right]$$

where ℓ is a loss function and *reg* a regularization over $\mathbf{M}$.

One of the most famous algorithm: LMNN

Large Margin Nearest Neighbor [Weinberger et al.,09]

- For k-Nearest-Neighbor classification
  $LS = \{(\mathbf{x}_i, y_i\}_{i=1}^n, \mathbf{x}_i \in \mathbb{R}^d, y_i \in Y$ discrete label set$\}$
- Constraints derived from labeled data
  - $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{x}_j) : y_i = y_j, \mathbf{x}_j$ belongs to the k-neighborhood of $\mathbf{x}_i\}$
  - $\mathcal{R} = \{(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) : (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{S}, y_i \neq y_k\}$



One of the most famous algorithm: LMNN

Formulation

$$\min_{\mathbf{M}\in\mathbb{S}_+^d,\boldsymbol{\xi}\geq 0} \quad (1-\mu) \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}_{lmnn}} d^2_{\mathbf{M}}(\mathbf{x}_i,\mathbf{x}_j) \quad + \quad \mu \sum_{i,j,k} \xi_{ijk}$$

$$\text{s.t.} \quad d^2_{\mathbf{M}}(\mathbf{x}_i,\mathbf{x}_k) - d^2_{\mathbf{M}}(\mathbf{x}_i,\mathbf{x}_j) \geq 1 - \xi_{ijk} \quad \forall(\mathbf{x}_i,\mathbf{x}_j,\mathbf{x}_k) \in \mathcal{R},$$

where $\mu \in [0,1]$ controls the trade-off between pulling target neighbors closer together and pushing away impostors.

- Convex formulation
- Number of constraints in the order of kn^2
  - Efficient solver based on projected gradient descent
  - Other alternative: only consider closest "impostors"
- Which metric to build the constraints?
- Possible overfitting in high dimensions (lack of regularization)

Importance of the regularization

Adding a regularization term to prevent overfitting

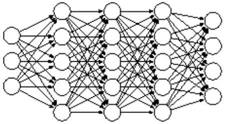
- Simple and classic choice: $\|\mathbf{M}\|_{\mathcal{F}}^2 = \sum_{i,j}^{d} \mathbf{M}_{ij}^2$ (Frobenius norm)
- Using some prior metrics : $\|\mathbf{M} - \mathbf{I}\|_{\mathcal{F}}^2$
- LogDet divergence $tr(\mathbf{M}\mathbf{M}_0^{-1}) - logdet(\mathbf{M}\mathbf{M}_0^{-1})$ (used in ITML [Davis et al.,07]).
  Remain close to a good prior metric, implicitly ensures that **M** is PSD, convex in **M**
- Feature selection with Mixed $L_{2,1}$ norm: $\|\mathbf{M}\|_{2,1} = \sum_i^d \|\mathbf{M}_i\|$ L_2 regularization over the columns, convex but non smooth
- Favoring low rank matrices for dimensionality reduction with trace (or nuclear) norm $\|\mathbf{M}\|_* = \sum_{i=1}^d \sigma_i(\mathbf{M})$ (convex but non smooth, use efficient Frank-Wolfe algorithms)

Non-Linear Extensions

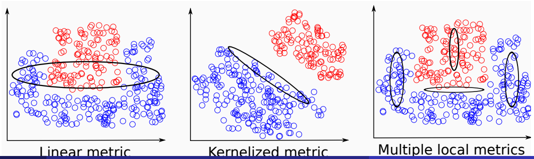
3 main types off approaches

Linear formulations are have the advantage to be convex and robust to overfitting **but** are unable to capture nonlinear structure

- Kernelized metrics: learn the metric from a feature space induced by a kernel - often very technical/ Other solution: use KPCA as a pre-process.
- Learn non linear mappings: Deep Metric Learning (siamese networks)



- Local metrics: learn many metrics for different part of the space. Pb: space splitting, blow-up of parameters, comparisons



Application: RGB color model

RGB color model

The RGB color model is an additive color model in which red, green, and blue light are added together in various ways to reproduce a broad array of colors.

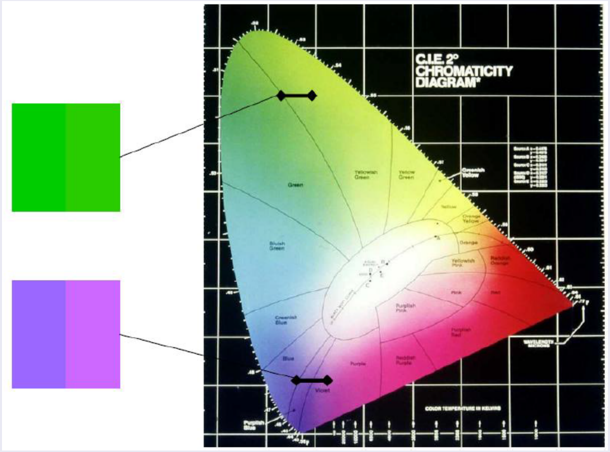
Color Differences

Required Features for Color Differences

In computer vision, the evaluation of color differences is required for many applications (image segmentation, visual salient region detection, edge and corner detection, etc.). For most of these applications, the color difference has to be:

- **robust** to acquisition condition variations,
- **discriminative**,
- and above all, **perceptual**, i.e. proportional to the color difference **perceived by human observers**.

Limitations of the of Non Uniform Spaces



The RGB space is known to be **non uniform**

From non uniform to uniform spaces

Solution

A classical strategy consists in using a **default transformation** (assuming standard viewing conditions) **from the RGB components to uniform spaces**, like $L^*a^*b^*$ or $L^*u^*v^*$ spaces and then computing:

- either the Euclidean Distance
- or the $\Delta E2000$

How to pass from RGB space to $\Delta E2000$?

$$\begin{tikzcd} \text{RGB} \arrow[r] & \text{XYZ} \arrow[r] & \text{L*a*B*} \arrow[r] & \Delta E2000 \end{tikzcd}$$

How to pass from RGB space to $\Delta E2000$?

$$\begin{tikzcd} \text{RGB} \arrow[r, red] & \text{XYZ} \arrow[r] & \text{L*a*B*} \arrow[r] & \Delta E2000 \end{tikzcd}$$

$$\text{sRGB} = \left(\frac{\frac{\text{RGB}}{255} + 0.055}{1.055} \right)^{2.4}$$

Then

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \mathbf{T} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

with

$$\mathbf{T} = \begin{pmatrix} 0.4125 & 0.3576 & 0.1804 \\ 0.2127 & 0.7152 & 0.0722 \\ 0.0193 & 0.1192 & 0.9503 \end{pmatrix}$$

How to pass from RGB space to $\Delta E2000$?

$$\text{RGB} \longrightarrow \text{XYZ} \longrightarrow \text{L*a*B*} \longrightarrow \Delta E2000$$

$$L^* = 116 f\left(\frac{Y}{Y_w}\right) - 16$$

$$a^* = 500\left(f\left(\frac{X}{X_w}\right) - f\left(\frac{Y}{Y_w}\right)\right)$$

$$b^* = 200\left(f\left(\frac{Y}{Y_w}\right) - f\left(\frac{Z}{Z_w}\right)\right)$$

with

$$f(t) = \begin{cases} t^{\frac{1}{3}} & \text{if} \quad t > 0.008856 \\ \frac{903.3t+16}{116} & \text{if} \quad t \leq 0.008856 \end{cases}$$

How to pass from RGB space to $\Delta E2000$?

$$\begin{tikzcd}\text{RGB} \arrow[r] & \text{XYZ} \arrow[r] & \text{L*a*B*} \arrow[r] & \Delta E2000\end{tikzcd}$$

$$C^*_{i,ab} = \sqrt{(a^*_i)^2 + (b^*_i)^2} \qquad i=1,2$$

$$\bar{C}^*_{ab} = \frac{C^*_{1,ab} + C^*_{2,ab}}{2}$$

$$G = 0.5\left(1 - \sqrt{\frac{\bar{C}^{*\,7}_{ab}}{\bar{C}^{*\,7}_{ab} + 25^7}}\right)$$

$$a'_i = (1+G)a^*_i \qquad i=1,2$$

$$C'_i = \sqrt{(a'_i)^2 + (b^*_i)^2} \qquad i=1,2$$

$$h'_i = \begin{cases} 0 & b^*_i = a'_i = 0 \\ \tan^{-1}(b^*_i, a'_i) & \text{otherwise} \end{cases} \qquad i=1,2$$

$$\Delta L' = L^*_2 - L^*_1$$

$$\Delta C' = C'_2 - C'_1$$

$$\Delta h' = \begin{cases} 0 & C'_1C'_2 = 0 \\ h'_2 - h'_1 & C'_1C'_2 \neq 0;\ |h'_2 - h'_1| \leq 180^\circ \\ (h'_2 - h'_1) - 360 & C'_1C'_2 \neq 0;\ (h'_2 - h'_1) > 180^\circ \\ (h'_2 - h'_1) + 360 & C'_1C'_2 \neq 0;\ (h'_2 - h'_1) < -180^\circ \end{cases}$$

$$\Delta H' = 2\sqrt{C'_1C'_2}\,\sin\left(\frac{\Delta h'}{2}\right)$$

$$\bar{L}' = (L^*_1 + L^*_2)/2$$

$$\bar{C}' = (C'_1 + C'_2)/2$$

$$\bar{h}' = \begin{cases} \dfrac{h'_1 + h'_2}{2} & |h'_1 - h'_2| \leq 180^\circ;\ C'_1C'_2 \neq 0 \\ \dfrac{h'_1 + h'_2 + 360^\circ}{2} & |h'_1 - h'_2| > 180^\circ;\ (h'_1 + h'_2) < 360^\circ;\ C'_1C'_2 \neq 0 \\ \dfrac{h'_1 + h'_2 - 360^\circ}{2} & |h'_1 - h'_2| > 180^\circ;\ (h'_1 + h'_2) \geq 360^\circ;\ C'_1C'_2 \neq 0 \\ (h'_1 + h'_2) & C'_1C'_2 = 0 \end{cases}$$

$$\boxed{\begin{aligned} \Delta E^{12}_{00} &= \Delta E_{00}(L^*_1, a^*_1, b^*_1;\ L^*_2, a^*_2, b^*_2) \\ &= \sqrt{\left(\frac{\Delta L'}{k_L S_L}\right)^2 + \left(\frac{\Delta C'}{k_C S_C}\right)^2 + \left(\frac{\Delta H'}{k_H S_H}\right)^2 + R_T\left(\frac{\Delta C'}{k_C S_C}\right)\left(\frac{\Delta H'}{k_H S_H}\right)}. \end{aligned}}$$

$$T = 1 - 0.17\cos(\bar{h}' - 30^\circ) + 0.24\cos(2\bar{h}') + 0.32\cos(3\bar{h}' + 6^\circ) - 0.20\cos(4\bar{h}' - 63^\circ)$$

$$\Delta\theta = 30\exp\left\{-\left[\frac{\bar{h}' - 275^\circ}{25}\right]^2\right\}$$

$$R_C = 2\sqrt{\frac{\bar{C}'^7}{\bar{C}'^7 + 25^7}}$$

$$S_L = 1 + \frac{0.015(\bar{L}' - 50)^2}{\sqrt{20 + (\bar{L}' - 50)^2}}$$

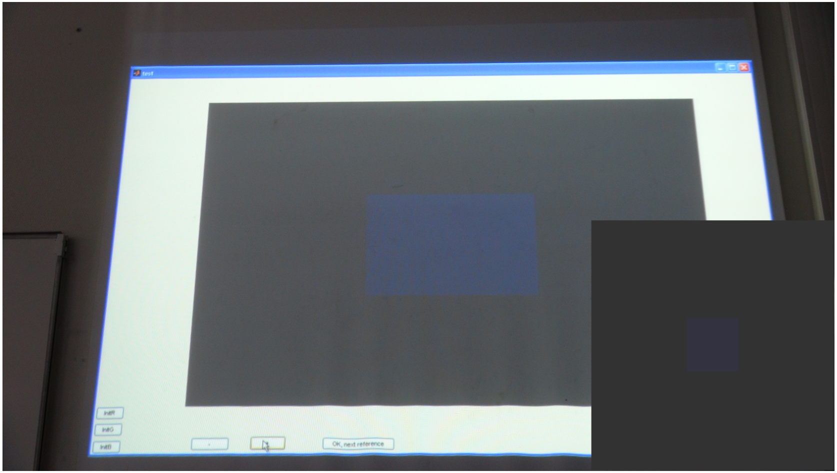
$$S_C = 1 + 0.045\bar{C}'$$

$$S_H = 1 + 0.015\bar{C}'T$$

$$R_T = -\sin(2\Delta\theta)\,R_C$$

Effect of the acquisition conditions on the perception of the color

However, the acquisition device settings (color filtering, white balance, gamma correction, demosaicing, etc.) or the illumination conditions are not taken into acccount in the default transformations.



Contributions - [Perrot et al., ECCV'14]

Our strategy

$$\begin{tikzcd} \text{RGB} \arrow[r] \arrow[rrr, bend right, red, "\text{Metric Learning}"'] & \text{XYZ} \arrow[r] & \text{L*a*B*} \arrow[r] & \Delta E2000 \end{tikzcd}$$

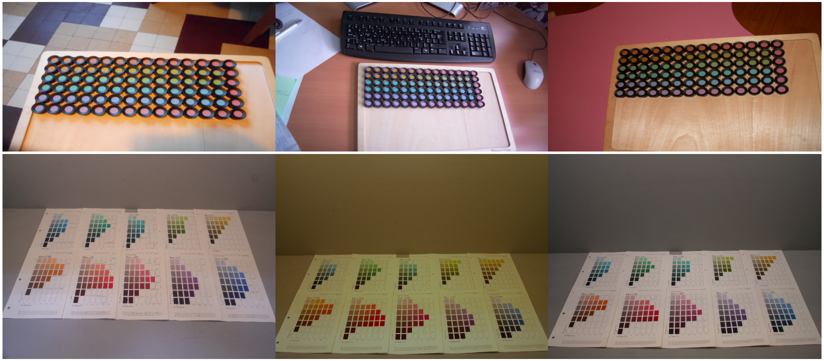
Contributions

1. Creation of a database of similar pairs of color patches (used as **must-link constraints**) $\rightarrow$ **benchmark for the community**.
2. Optimization of local metrics than correctly estimate $\Delta E2000$ in the projected space $\rightarrow$ one possible way to **capture non linearity**.
3. Experiments in **image segmentation**.

Dataset of color patches - Requirements

1. Patches of color must be **well distributed** in the RGB cube to allow the metrics to generalize well.
2. Both the **RGB components and the true** $\Delta E2000$ **(from a colorimetric point of view) have to be known** for each patch.
3. Since *hue, chroma* and *luminance* impact the perceptual color difference, the **dataset has to cover variations** w.r.t. these 3 features.
4. To get reference distances whatever the acquisition conditions, we used **four different cameras**:
   - *Kodak* DCS Pro 14n
   - *Konica Minolta* Dimage Z3
   - *Nikon* Coolpix S6150
   - *Sony* DCR-SR32

$\boxed{\text{We get 697200 Pairs of colors}}$



We used two well-known sets of patches:
1/ Farnsworth-Munsell 100 Hue Color
2/ Munshell atlas

We used a sprectroradiometer (Minolta CS 1000) to measure the spectra of each patch to get the true $L^*a^*b^*$ coordinates under D65 illuminant.

Metric learning setting

Setting

- Input $X \times X \times \mathbb{R} \subseteq [0,1]^3 \times [0,1]^3 \times \Delta E_{2000}$
- Mahalanobis $d^2(\mathbf{x} - \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^T \mathbf{M} (\mathbf{x} - \mathbf{x}')$
- $\mathbf{M}$ symmetric PSD matrix ($\mathbf{x}^T \mathbf{M} \mathbf{x} \geq 0$)
  - $\mathbf{M} = \mathbf{L}^T \mathbf{L}$
  - $d^2(\mathbf{x} - \mathbf{x}') = (\mathbf{L}\mathbf{x} - \mathbf{L}\mathbf{x}')^T (\mathbf{L}\mathbf{x} - \mathbf{L}\mathbf{x}')$
- Learning sample $\{\mathbf{x}_i, \mathbf{x}'_i, \Delta E_{2000}(\mathbf{x}_i, \mathbf{x}'_i)\}_i^m$, $\Delta E_{2000}(\mathbf{x}_i, \mathbf{x}'_i) \leq\sim 5$

$\Rightarrow$ Find $\mathbf{M}$ to obtain a good approximation of ΔE_{2000}

Learning local metrics

Algorithm 1: Local metric learning

```
input  : A training set $S$ of patches; a parameter $K \geq 2$
output: $K$ local Mahalanobis distances and one global metric
begin
    Run $K$-means on $S$ and deduce $K+1$ training subsets $T_j$ ($j = 0, 1 \ldots, K$) of triplets
      $T_j = \{(\mathbf{x_i}, \mathbf{x_i'}, \Delta E_{00})\}_{i=1}^{n_j}$ (where $(\mathbf{x_i}, \mathbf{x_i'})$ are similar patches in region $C_j$ with a
      $\Delta E_{2000} < 5$)
    for $j = 0 \to K$ do
        Learn $\mathbf{M_j}$ by solving the convex optimization Problem (1) using $T_j$
```

$$\underset{\mathbf{M_j} \succeq 0}{\arg\min} \ \hat{\varepsilon}_{T_j}(\mathbf{M_j}) + \lambda_j \|\mathbf{M_j}\|_{\mathcal{F}}^2, \qquad (1)$$

where:

- $\hat{\varepsilon}_{T_j}(\mathbf{M_j}) = \frac{1}{n_j} \sum_{(\mathbf{x}, \mathbf{x}', \Delta E_{00}) \in T_j} \left| (\mathbf{x} - \mathbf{x}')^T \mathbf{M_j} (\mathbf{x} - \mathbf{x}') - \Delta E_{00}(\mathbf{x}, \mathbf{x}')^2 \right|$,
- $\lambda_j > 0$ is a regularization parameter,
- $\| \cdot \|_{\mathcal{F}}$ denotes the Frobenius norm.

Theoretical Analysis

Two types of possible theoretical results on a learned metric:

1. Consistency guarantee of the metric learning algorithm $\rightarrow$ **Uniform stability**
2. Generalization guarantee of the algorithm making use of the learned metric.

$$R(\mathbf{M}) \leq R_n(\mathbf{M}) + \mathcal{O}\left(\frac{\text{complexity}(\mathbf{M})}{\sqrt{n}}\right)$$

Theoretical bound on the learned metrics

Intuitively, an algorithm is said **stable** if it is **robust to small changes** in the training sample, i.e., the variation in its output h is small.

Definition (Uniform stability)

An algorithm A has uniform stability $\frac{\kappa}{n}$ with respect to a loss function l if the following holds:

$$\forall S, \forall i \in \{1, ..., n\}, sup_z |l(h_S, z) - l(h_{S_i}, z)| \leq \frac{\kappa}{n},$$

where κ is a positive constant, S_i is obtained from the training sample S by replacing the ith example $z_i \in S$ by another example z_i' drawn i.i.d. from $\mathcal{D}_{\mathcal{X}}$, h_S and h_{S_i} are the hypothesis learned by A from S and S_i respectively.

Uniform stability

When the previous Definition is fulfilled, the following generalization bound in $\mathcal{O}(1/\sqrt{n})$ holds.

Theorem (Uniform stability)

Let S be a training sample of size n and $\delta > 0$. For any algorithm A with uniform stability $\frac{\kappa}{m}$ with respect to a loss function l bounded by 1, with probability $1 - \delta$, we have:

$$\mathcal{R}_{h_S} \leq \hat{\mathcal{R}}_{h_S} + \frac{\kappa}{n} + (2\kappa + 1)\sqrt{\frac{ln\frac{1}{\delta}}{2n}}$$

Uniform stability can be used to derive generalization guarantees for hypothesis classes that are difficult to analyze with classic complexity arguments, such as **k-nearest neighbors** or **support vector machines**.

Theoretical bound on the learned metrics

Theorem

Let $C_0, C_1, \dots, C_k$ be the regions considered, then for any set of metrics $\mathbf{M} = \{\mathbf{M_0}, \dots, \mathbf{M}_K\}$ learned by Algorithm 1, each region T_j has the uniform stability in $\frac{\mathcal{K}}{n_j}$ with $\mathcal{K} = \frac{2D_j^4}{\lambda_j}$, then we have with probability at least $1-\delta$ over the global loss:

$$\begin{aligned}\varepsilon(\mathbf{M}) \quad \leq \quad & \hat{\varepsilon}_T(\mathbf{M}) + L_B\sqrt{\frac{2(K+1)\ln 2 + 2\ln 2/\delta}{n}} + \frac{2(KD^4+1)}{\lambda n} \\ & + \left(\frac{2(KD^4+1)}{\lambda} + \Delta_{\max}(\frac{2(KD^2+1)}{\sqrt{\lambda}} + 2(K+1)\Delta_{\max}\right)\sqrt{\frac{\ln(\frac{4(K+1)}{\delta})}{2n}},\end{aligned}$$

where $D = \max_{1\leq j\leq K} D_j$, $L_B = \max\{\frac{\Delta_{max}}{\sqrt{\lambda}}, \Delta_{max}^2\}$ is the bound on the loss function and $\lambda = \min_{0\leq j\leq K}\lambda_j$ is the minimum regularization parameter among the $K+1$ learning problems used in Algorithm 1.

Quality Assessment of the Metrics

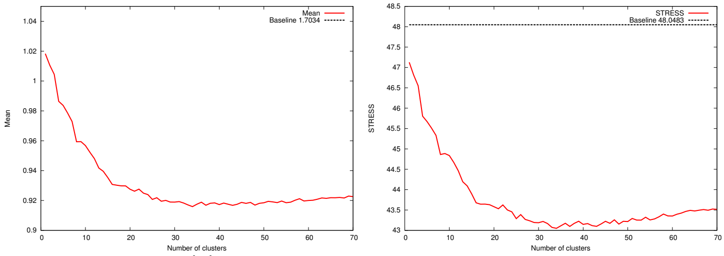
2 criteria:

- Mean Absolute Difference
- STRESS

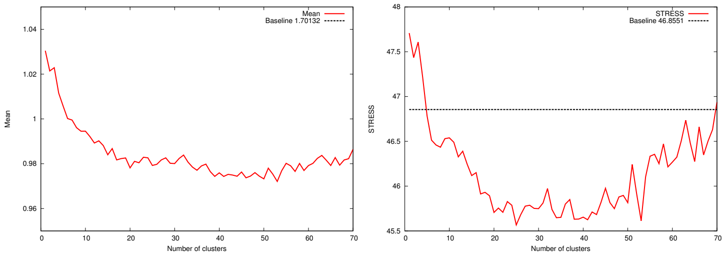
STRESS

$$\text{STRESS} = 100\sqrt{\left(\frac{\sum(d_{\mathbf{M}}(\mathbf{x},\mathbf{x}') - F\Delta E2000_{ref}(\mathbf{x},\mathbf{x}'))^2}{\sum F^2 \Delta E2000_{ref}(\mathbf{x},\mathbf{x}')^2}\right)}$$

$$F = \frac{\sum d_{\mathbf{M}}(\mathbf{x},\mathbf{x}')^2}{\sum d_{\mathbf{M}}(\mathbf{x},\mathbf{x}')\Delta E2000_{ref}(\mathbf{x},\mathbf{x}')}$$



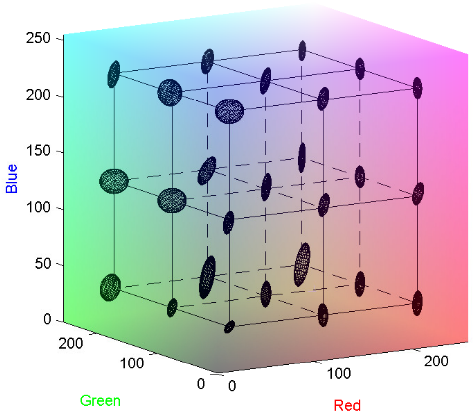
(a) Generalization to new colors.



(b) Generalization to new cameras.

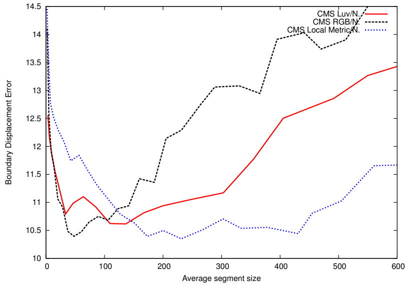
Baseline: Sharma et al., Color Research Applications, 2005.

Interest of learning local metrics

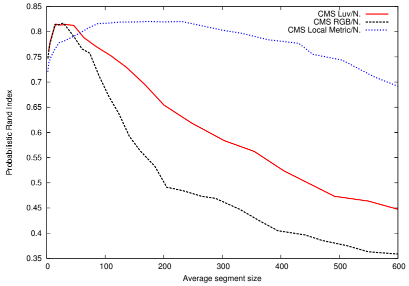


The RGB cube has been splitted into 20 regular regions, where the surface of each represented ellipsoid corresponds to the RGB colors lying at the corresponding learned local perceptual distance of 1 from the center of the ellipsoid.

Image segmentation results



(c)



(d)

Figure: (a) Boundary Displacement Error (lower is better) versus the average segment size.; (b) Probabilistic Rand Index (higher is better) versus the average segment size.

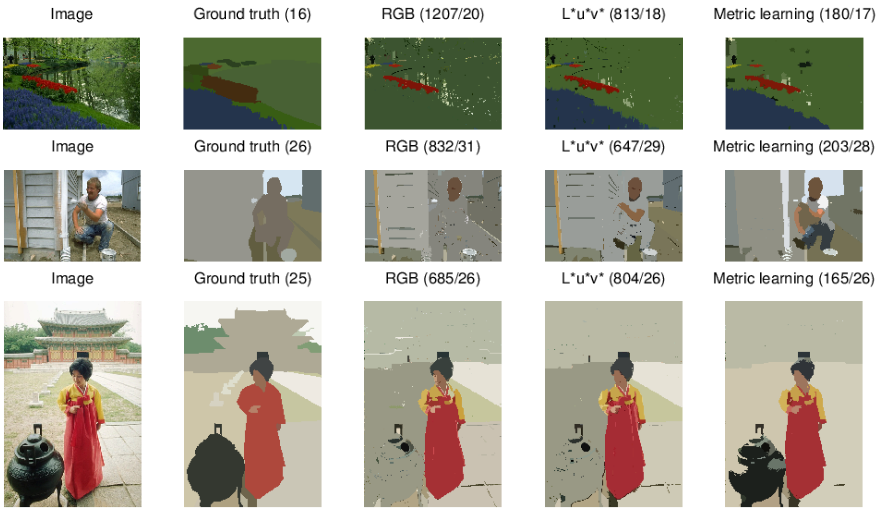
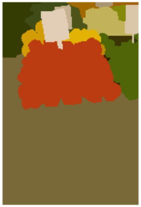


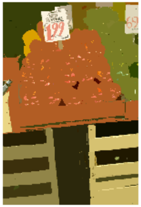
Image
Ground truth (16)
RGB (1207/20)
L*u*v* (813/18)
Metric learning (180/17)
Image
Ground truth (26)
RGB (832/31)
L*u*v* (647/29)
Metric learning (203/28)
Image
Ground truth (25)
RGB (685/26)
L*u*v* (804/26)
Metric learning (165/26)

Image

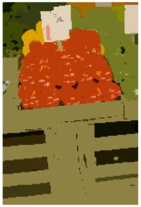
Ground truth (24)



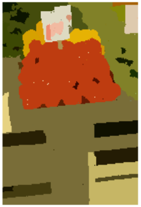
RGB (1022/37)



L*u*v* (931/33)

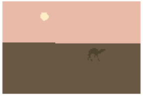


Metric learning (172/33)

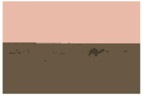


Image

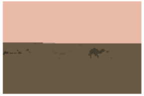
Ground truth (4)



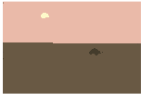
RGB (99/4)



L*u*v* (88/4)

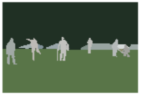


Metric learning (16/4)

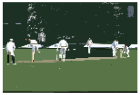


Image

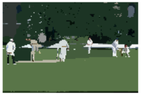
Ground truth (27)



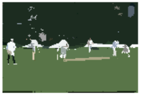
RGB (1343/31)



L*u*v* (1032/34)

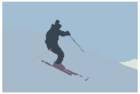


Metric learning (254/31)

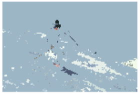


Image

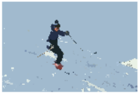
Ground truth (12)



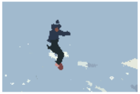
RGB (748/13)



L*u*v* (495/12)



Metric learning (110/12)



Image

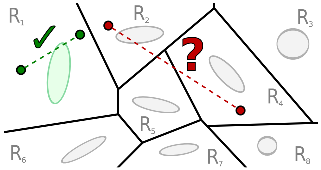
Ground truth (42)

RGB (2080/49)

L*u*v* (1350/46)

Metric learning (438/46)

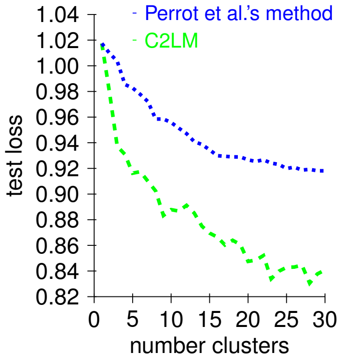
Improvement: Weighting local models

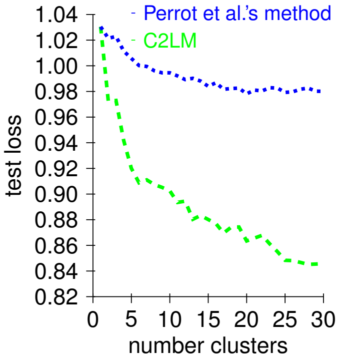


Solution: Combining weighted models [Zantedeschi et al.,CVPR'16]

- $d_{ij}(\mathbf{x}, \mathbf{x}') = \sum_{k=1}^{K} W_{ij} d_{\mathbf{M}_k}(\mathbf{x}, \mathbf{x}')$
  A different (convex) weighting for each pairs of regions
- Loss: $\frac{1}{n} \sum_{i=1}^{K} \sum_{j=1}^{i} \sum_{\mathbf{x},\mathbf{x}' \in R_{ij}} |d_{ij}(\mathbf{x}, \mathbf{x}') - \Delta E_{2000}(\mathbf{x}, \mathbf{x}')|$
- Double regularization: $D(W) = \sum_{k=1}^{K} \sum_{j=1}^{i} \|E_{ij}^T W_{ij}\|$
  $S(W) = \sum_{i=1}^{K} \sum_{j=1}^{i} \sum_{i'=1}^{K} \sum_{j'=1}^{i'} K_{iji'j'} \|W_{ij} - W_{i'j'}\|$
  E and K embed prior and relative influence of the different metrics

Experimental improvement





Conclusion

- Local metric
- Simple model local models (3×3 matrices)
- Generalization bound
- Dataset of color patches

Perspectives

- More complex models - Deep metric learning - regularizers
- Incorporation of constraints from physics
- Transfer learning