Image-based Classification of Swiss Traditional Costumes using Contextual Features

Artem Khatchatourov and Christoph Stamm
Institute of Mobile and Distributed Systems, FHNW University of Applied Sciences and Arts Northwestern Switzerland, Bahnhofstrasse 6, 5210 Windisch, Switzerland

Keywords: Feature-based Object Recognition, Pattern Matching, Clothing Recognition, Machine Learning.

Abstract: In this work we propose a method for feature-based clothing recognition, and prove its applicability by performing image-based recognition of Swiss traditional costumes. We employ an estimation of a simplified human skeleton (a poselet) to extract visually indistinguishable but reproducible features. The descriptors of those features are constructed, while accounting for possible displacement of clothes along human body. The similarity metrics mean squared error and correlation coefficient are surveyed, and color spaces YIQ and CIELAB are investigated for their ability to isolate scene brightness in a separate channel. We show that the model trained with mean squared error performs best in the CIELAB color space and achieves an $F_{0.5}$-score of 0.77. Furthermore, we show that omission of the brightness channel produces less biased, but overall poorer descriptors.

1 INTRODUCTION

Traditionally, feature-based object recognition is accomplished by learning features of an object and matching them to those extracted from a new image. The most well-known example of this method is the work of Lowe *et al.* (Lowe, 1999), which describes the Scale-Invariant Feature Transform method (SIFT). In general, this approach only works on mass-produced and rigid objects that look exactly alike in every picture. Feature-based object recognition will fail to recognize flexible objects, as they produce visual features that are irreproducible in other images. Therefore, in this work, instead of visual keypoints we use contextual ones, defined by a simplified approximation of human skeleton, called poselet. Even though human features are concealed under clothes, newly developed methods for pose estimation allow to extract them from the image.

Another drawback of established feature-based object recognition methods is their color-blindness. Most of them are constrained to work only with gray-scale images in order to maximize invariance to changing lighting conditions between the scenes. For clothing recognition, however, color is an essential feature that must be included in the feature description. The influence of illumination in the image can be mitigated by isolating the brightness information to a luminance channel in the color spaces YIQ and CIELAB.

In the next section of this work we explore state-of-the-art methods for object and clothing recognition. In Section 3 we introduce the Swiss traditional costume dataset and describe our approach for feature extraction. We propose a matching method and construct feature descriptors in Section 4. In Section 5 we discuss the conducted experiments and determine optimal parameters for maximizing prediction precision. Finally, conclusions are drawn in Section 6.

2 RELATED WORK

The task of garment recognition in images was tackled by Yamaguchi *et al.* (Yamaguchi et al., 2012) and by Kalantidis *et al.* (Kalantidis et al., 2013). Both works use image segmentation and pose estimation to identify spacial location of a certain region on the human body. Combining the position and shape of this region with information of the neighboring regions enables its classification as a specific clothing item. Additionally, color histograms were introduced to determine the color of a garment. Both of these works use the Fashionista dataset introduced previously (Yamaguchi et al., 2012) containing 158,235 images of people wearing common, every-day clothes. Garment

Khatchatourov, A. and Stamm, C.
Image-based Classification of Swiss Traditional Costumes using Contextual Features.
DOI: 10.5220/0009102404120420
In *Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2020) - Volume 4: VISAPP*, pages 412-420
ISBN: 978-989-758-402-2; ISSN: 2184-4321
Copyright © 2022 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

items in each image are marked with a polygon and annotated. Another clothing dataset, DeepFashion, introduced in (Liu et al., 2016) contains over 800,000 images. In contrast, our Swiss traditional costume dataset contains 247 classes and merely 1540 samples overall.

In cases where no large dataset is available, we use feature recognition to find objects from a training set as small as a single sample. Generally, it only works well with non-deformable objects that look exactly alike on all images, which is almost impossible to achieve with clothing. Nevertheless, feature based garment recognition has been attempted in numerous works. Most notably, Chen *et al.* (Chen et al., 2012) use the work in (Eichner et al., 2012) to locate body parts, Maximum Response Filters (Varma and Zisserman, 2005) to describe texture and color in LAB color space. Additionally, dense SIFT features along the body are used to describe the garment on each body part (Lowe, 1999).

3 TRADITIONAL COSTUMES

Traditional costumes are a special kind of clothing used to represent culture and customs of a community. The costume is bound to a specific region. In Switzerland, the sizes of such regions vary from a whole canton (administrative subdivision of Switzerland) to an area as small as single village. People from every region incorporate their traditions, values and history into costumes complementing them with distinctive accessories and details. Such celebratory costumes are often worn at inter-regional festivities to represent heritage of a particular locality.

Additionally to celebratory costumes, regions or groups of regions define their own costumes for other purposes. Work wear is usually common for each canton spanning many regions. Also, most protestant regions often define special clothing for church visits. Some regions and cantons have their own costumes for certain traditional professions like wine or cheese makers. The aim of this project is to help experts and amateurs to identify the costumes, which is usually not possible without proper reference material.

A special committee in every canton defines regulations for a costume's appearance in a written form, and sets up guidelines that every tailor must follow. Once set, these guidelines do not change. Many costumes, however, allow for some variations: e.g. a different set of colors for certain clothing parts or a winter version of a costume for cold weather.

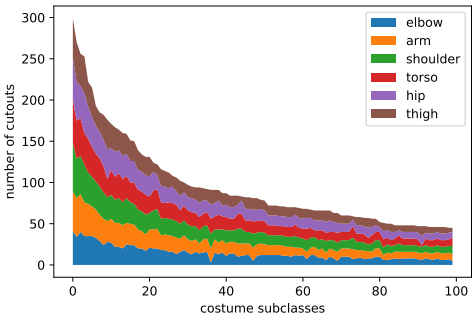


Figure 1: Distribution of samples among the 100 largest subtypes. The amount of samples of each body part is marked with color.

3.1 Dataset

Costume descriptions are written in the official language of a given canton: German, French, Italian or even Romansh; and contain many specific names for clothing parts. The task of teaching a computer to deduce a visual image from these descriptions seems to be infeasible. Thus, it was decided to collect a dataset of images. Each image in this dataset depicts one person wearing a traditional costume. In cases where variations of a costume are defined, we create subtypes of this costume. We allow only one level of subtype for a costume supertype. Subtypes are used only to produce consistent descriptors, the classification goal, however, remains supertype classification. Ultimately, our dataset contains 1540 images from 274 supertypes and 427 subtypes. The small size of the dataset correlates with the small number of unique costume samples available. The national costume community is very small and each costume is tailored to its owner. To our knowledge, this the first attempt to apply machine learning to recognize traditional clothing.

We have decided against dataset augmentation because of the several reasons: 1. linear transformation is reverted in the preprocessing stage; 2. non-linear transformation distorts important patterns on the costume; 3. altering color and adding noise to images interferes with pattern alignment at learning stage. This dataset defines the ground truth of costume appearance in our work. The images are distributed non-uniformly among classes as shown in Figure 1. The amount of samples per costume type is very unbalanced. Usually, the prevalence of a costume in the dataset is proportional to the population of its region of origin, but some of the smaller classes result from fracturing the costume supertype.

3.2 Preprocessing

As we have seen in the works in Section 2, it is important to determine the pose of a person in the image. We accomplish this using the open-source OpenPose library (Cao et al., 2018). It is based on the work of Wei *et al.* (Wei et al., 2016), who uses deep learning to estimate the location of body parts in the image and connects best fitting parts together to produce a poselet: a simplified human skeleton that approximates a pose. This technique was further improved by the work of Cao *et al.* (Cao et al., 2017), who introduced affinity fields for better handling of multi-person scenes.

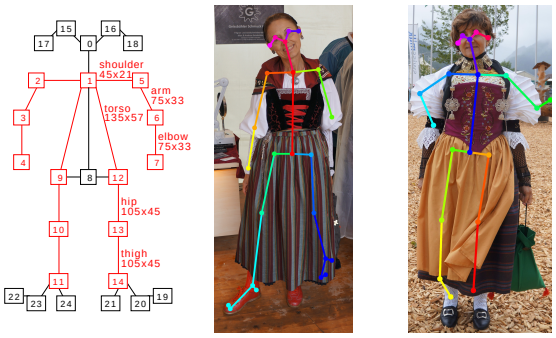
We use OpenPose with the BODY25 model for poselet detection, which produces a poselet with 25 keypoints, numbered as shown in Figure 2a. In most cases the approximations are correct, however, OpenPose sometimes fails to properly locate legs concealed by a skirt or a dress as seen in Figure 2b. It can also be misled by unusual shapes of some clothing parts, as shown in Figure 2c. In this example, the elbow keypoints are placed onto balloon-sleeves. As a consequence, the attached hand keypoints are misplaced as well. Poselet parts from incorrectly recognized keypoints must be filtered from the training set manually.

We identify twelve poselet connections, marked red in Figure 2a, where we expect to find clothing parts. From each connection we cut out a rectangular image patch, rotated in alignment with this connection. The use of patches between two keypoints as features removes the variance in rotation and scale. We proceed by mirroring the opposing patches. This is possible due to the symmetry of costumes. Mirroring reduces the number of features for each class from twelve to six. At the same time, it doubles the amount of patches for each feature and, thus, mitigates the scarcity of samples in the dataset.

4 DESCRIPTOR CONSTRUCTION

Our goal is to produce a general description for each feature, meaning that it must resemble all possible patches equally well. Ideally, such a description should be computed by acquiring mean pixel intensities of all patches. In our case this approach is not sufficient, since there is always a certain shift among the patch contents caused by variation in camera angle and fit of clothes. Our strategy is to allow for some transformation of patches and combine them at a location where they are the most similar.

We denote a patch collection $C = \{c_1, c_2, ..., c_n\}$,



(a) (b) (c)

Figure 2: Pose estimation with BODY25 model: a) positions of poselet joints, indexes and patch sizes (in pixels) used in our work; demonstration of misplaced positions of b) a leg and c) an elbow in the poselet.

which contains n patches of the same feature, originating from the same subtype of costume. The patches in the collection are compared in pairs and the pair which appears to be most similar is merged producing a new patch. In the next iteration, this patch is compared to the remaining patches in the collection, and, again, the most similar pair is combined. Note, that there is no need to recalculate the pairs that have not been merged in the previous iteration. This process is repeated until all patches have been merged into one. This last patch is the feature descriptor f.

For each pair of patches we denote a target patch t and a query patch q, where $t, q \in C$. The query patch is displaced relative to the target patch by $\bar{x}$, $\bar{y}$ and rotation α. The best position to merge them is where they are most similar. For similarity $\rho(t,q)$ between the two patches we evaluate two different metrics: mean squared error (*mse*) and correlation coefficient (r). We will cover both metrics in detail in Section 4.5.

4.1 Displacement

The similartiy of a patch pair is calculated by iterating through every pixel x and y in t, and their counterparts x' and y' in q, for every displacement $\bar{x}$, $\bar{y}$ and α. For convenience, we denote the displacement parameters ξ, thus:

$$\xi = (\bar{x}, \bar{y}, \alpha). \tag{1}$$

Our goal is to determine the displacement position where *mse* is minimal or r is maximal, depending on the metric used:

$$\rho(t,q) = \begin{cases} \arg\min\limits_{\xi} & mse_{\xi}(t,q) \\ \arg\max\limits_{\xi} & r_{\xi}(t,q) \end{cases} \tag{2}$$

We limit the displacement to $1/4$ of the width and height of a target patch, and rotation of -10° to 10°

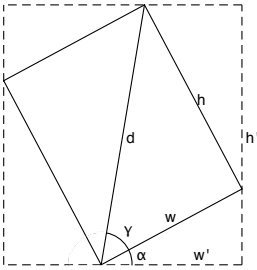


Figure 3: Rotating the patch of size $h \times w$ by α extends its bounding box (dashed lines) to new dimensions $h' \times w'$. Note that the length of the hypotenuse stays unchanged, while its angle to w' varies.

with a step of 2°. Our experiments have shown that allowing for larger displacement enables the algorithm to achieve maximal similarity by reducing the overlap area of the two patches to a minimum instead of matching the patterns.

4.2 Displacement Calculation

Mapping of every pixel (x,y) in q to its transformed position (x',y') in t is accomplished by rotating of q by α, translating it by $\bar{x}$ and $\bar{y}$ and compensating the size differences of the two patches, which gives us the following equation:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = M_\xi \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}. \tag{3}$$

The affine transformation matrix M is computed from five separate matrices as follows:

$$M_\xi = T_D T_\Delta T_{o^+} R T_{o^-}, \tag{4}$$

where T_{o^+} translates the center $(\hat{x}_q, \hat{y}_q)$ of q to the origin (coordinates $(0,0)$), R rotates the image around the origin by α, and T_{o^-} returns the center of q to the original position:

$$T_{o^{+/-}} = \begin{pmatrix} 1 & 0 & {}^{+/-}\hat{x}_q \\ 0 & 1 & {}^{+/-}\hat{y}_q \\ 0 & 0 & 1 \end{pmatrix}, R = \begin{pmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{5}$$

After rotation the patch is translated by T_Δ in order to align the centers of t and q, and translated by T_D containing the displacement parameters:

$$T_\Delta = \begin{pmatrix} 1 & 0 & \hat{x}_q - \hat{x}_t \\ 0 & 0 & \hat{y}_q - \hat{y}_t \\ 0 & 0 & 1 \end{pmatrix}, \quad T_D = \begin{pmatrix} 1 & 0 & \bar{x} \\ 0 & 0 & \bar{y} \\ 0 & 0 & 1 \end{pmatrix}. \tag{6}$$

Thus, from equation 3 we derive the following formulae for calculation of x' and y':

$$x' = \quad x\cos\alpha + y\sin\alpha + x_{const}, \tag{7}$$

$$y' = -x\sin\alpha + y\cos\alpha + y_{const}, \tag{8}$$

where x_{const} and y_{const} are parts of the equation, that do not depend on x and y. These static members can be precomputed separately for each $\bar{x}$, $\bar{y}$ and α:

$$x_{const} = -\hat{x}_q\cos\alpha - \hat{y}_q\sin\alpha + \bar{x} + 2\hat{x}_q - \hat{x}_t, \tag{9}$$

$$y_{const} = \quad \hat{x}_q\sin\alpha - \hat{y}_q\cos\alpha + \bar{y} + 2\hat{y}_q - \hat{y}_t. \tag{10}$$

4.3 Bounding Box Extension

A bounding box resulting from a combination of a patch pair is at least as small as the largest input patch (when displacement is $(0,0)$). It will grow with wider translation and rotation. In the former case, height and width of the patch will be extended by $\bar{x}$ and $\bar{y}$ respectively. For in-place rotation, the height h and width w of the query patch must be extended as follows:

$$h' = \sin(|\alpha| + \gamma)\, d, \qquad w' = \cos(|\alpha| - \gamma)\, d, \tag{11}$$

where γ is the angle between the hypotenuse d and the opposite h, as shown in Figure 3. These equations are based on the fact that γ and the d do not change with rotation of the rectangular patch by α.

4.4 Overlap Calculation

Note, that we can take into account only the overlapping regions of the two patches that change with every displacement. We denote the coordinates of an overlapping region I for displacement of ξ:

$$I_\xi = T \cap Q_\xi, \tag{12}$$

where T and Q_ξ are the sets of coordinates in t and $M_\xi q$ respectively. The number of overlapping pixels K_ξ is the cardinality of this set: $K_\xi = |I_\xi|$.

4.5 Similarity Calculation

We propose two methods for pixel-wise comparison of the patches: mean squared error (*mse*) and the correlation coefficient (r). The first method introduces solid similarity results and is easy to compute. The second one incorporates the ability to cancel out constant brightness intensity in the patches that results from different lighting conditions in the scene, thus, minimizing discrepancies between the patches. This method, however, also removes color information from patches with no color alterations.

Note, that the mean squared error must be minimized to maximize similarity. The correlation coefficient produces values between -1 and 1: higher value means better similarity. To avoid confusion, we will aim to maximize similarity, regardless of the metric method used. The equations in this section are to be applied to each channel separately and their mean is the similarity between t and q at a given displacement.

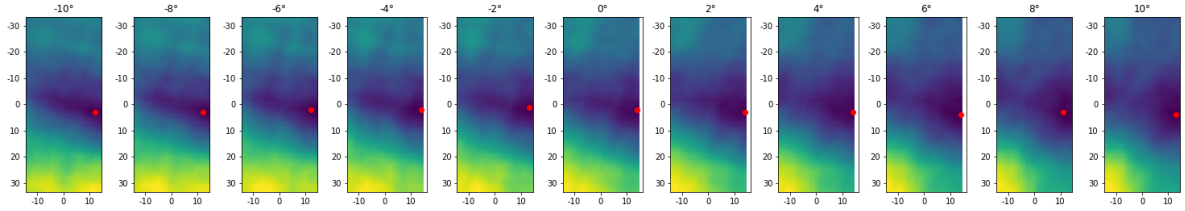


Figure 4: Slices of similarity space produced by the mean squared error between two patches in RGB color space. These patches are later shown in Figures 5a and 5b, and the merged result is shown in Figure 5d. Red dots point to the optimal displacement at each rotation. The red dot at rotation of -4° marks the best overall merging location.

4.5.1 Mean Squared Error

For the calculation of the *mse* we apply the common equation to every position in the overlap region:

$$mse_\xi = \frac{1}{K_\xi} \sum_{(x,y)\in I_\xi} [(t(x,y) - (M_\xi q)(x,y))^2]. \quad (13)$$

4.5.2 Correlation Coefficient

The correlation coefficient r of a given displacement is computed in two steps. First, the mean intensities $\bar{t}$, $\bar{q}$ of each input patch must be computed for every displacement combination:

$$\bar{t}_\xi = \frac{\sum_{(x,y)\in I_\xi} t(x,y)}{K_\xi}, \quad \bar{q}_\xi = \frac{\sum_{(x,y)\in I_\xi} (M_\xi q)(x,y)}{K_\xi}. \quad (14)$$

In the second step, these values are used to compute the correlation coefficient r_ξ:

$$r_\xi = \frac{\sum_{(x,y)\in I_\xi} [(t(x,y) - \bar{t}_\xi)((M_\xi q)(x,y) - \bar{q}_\xi)]}{\sqrt{\sum_{(x,y)\in I_\xi} [t(x,y) - \bar{t}_\xi]^2} \sqrt{\sum_{(x,y)\in I_\xi} [(M_\xi q)(x,y) - \bar{q}_\xi]^2}}. \quad (15)$$

4.6 Merging

The pixel-wise displacement and rotation produce a 3-dimensional similarity space, as presented in Figure 4. Its size is determined by the maximum allowed displacement. In our work we produce eleven planes with size of a maximal displacement in both directions, including displacement of $(0,0)$ and rotation of 0°. The best displacement to merge a pair of patches t and q is considered to be at global similarity *maxima* $\rho(t,q)$. To compensate for outliers that do not fit to the larger part of the patches in the collection, we combine pixel values weighted by the number of patches already merged to it in previous iterations. By iteratively combining the most similar patches, we produce a general description f for this feature. The six descriptors that describe all six features of a costume subtype v is denoted a feature set $\{f_1^v, f_2^v, ..., f_6^v\}$. The subtype v is one of m learned subtypes $V = \{v_1, v_2, ..., v_m\}$.

Using exhaustive search in order to find global extremes is a bold and time-consuming approach, and we are certain that faster heuristics can be employed for time-critical applications. However, in our case the similarity space is not uniformly distributed and, thus, contains many local *extremas*, which leads to increased complexity of optimizations required to yield comparable results.

4.7 Prediction

At prediction phase, we create a sample from an input image (as described previously in Section 3.2) containing twelve patches $S = \{s_i, s'_i : i \in 1..6\}$. The patches denoted as s' are the counterparts of s from the opposite side of the poselet. All patches in the sample are fitted as query patches to the descriptors of the corresponding features of each costume subtype as discussed before in this section.

We denote the similarity of a descriptor f and a patch s $\delta(f,s)$. In the essence, function δ consists of same equations as ρ but it yields the actual similarity score instead of the optimal displacement position:

$$\delta(f,s) = \begin{cases} \min\limits_{\xi} & mse_\xi(f,s) \\ \max\limits_{\xi} & r_\xi(f,s) \end{cases} \quad (16)$$

The cumulative similarity of each descriptor of a subtype to the corresponding patch in a sample denotes the overall subtype similarity. The costume subtype with the highest overall similarity is considered to be the best match:

$$\hat{v} = \arg\max_{v\in V} \frac{\sum_{i=1}^6 \delta(f_i^v, s_i)\beta_i}{\sum_{i=1}^6 \beta_i} + \frac{\sum_{i=1}^6 \delta(f_i^v, s'_i)\beta'_i}{\sum_{i=1}^6 \beta'_i}, \quad (17)$$

where m is the number of costume subtypes. The presence indicators $\beta, \beta' \in \{0,1\}$ make sure that only those patches which are present in the sample are taken into account, in case some features were not recognized by pose estimation at the previous stage.

(a) Normal (b) Warm (c) Dark (d) Merged

Figure 5: Comparison of patches from images shot in different lighting conditions (a – c). The merged patch (d) was produced by combining (a) and (b).

Note, that the goal of classification is to predict the supertype of the costume, not its subtype (where applicable). Mapping the prediction from a subtype to a supertype cancels out the confusion between similar subtypes and produces a higher probability for choosing the correct costume.

4.8 Color Spaces

Even though the costumes of the same subtype are meant to be of the same color, the pixel values in the images vary intensely. This color variance results from two factors. First, a tailor might interpret the guidelines provided by the traditional costume committee differently, which leads to slight variation of the color. Second, the lighting of the scene where the picture was taken adds further alteration to color appearance. An example of the latter is presented in Figure 5 (a – c). The lighting difference is very profound in the RGB color space since every color channel is affected. This is not the case with the YIQ and L*a*b* color spaces, where brightness is isolated in a separate channel (Y and L* respectively). In this work we compare these three color spaces.

4.8.1 YIQ

The YIQ color space is a television broadcast standard. It is linear to the RGB color space and it approximates the luminance in a separate channel. This allows us to leave out the Y-channel entirely and use only the chrominance channels for similarity computation. Therefore, we introduce the color model where all values in the Y-channel are set to 0 and call the resulting color space 0IQ.

4.8.2 L*a*b*

The CIELAB standard was selected, since one of its main properties is perceptual uniformity. It is one of the standards recommended for measuring color differences. In fact, the International Commission of Illumination (CIE) defines a color difference metric for L*a*b*, called ΔE. There is a number of standardized formulae for computing ΔE. One of them is CIE76 which corresponds to the Euclidean distance over all three channels (i.e. mean squared error). The chrominance channels a* and b* store the color information, and the channel L* holds the luminosity information. Same as for YIQ, we create the color model 0a*b* from L*a*b* with no brightness information.

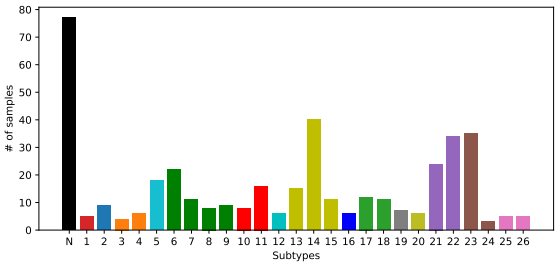


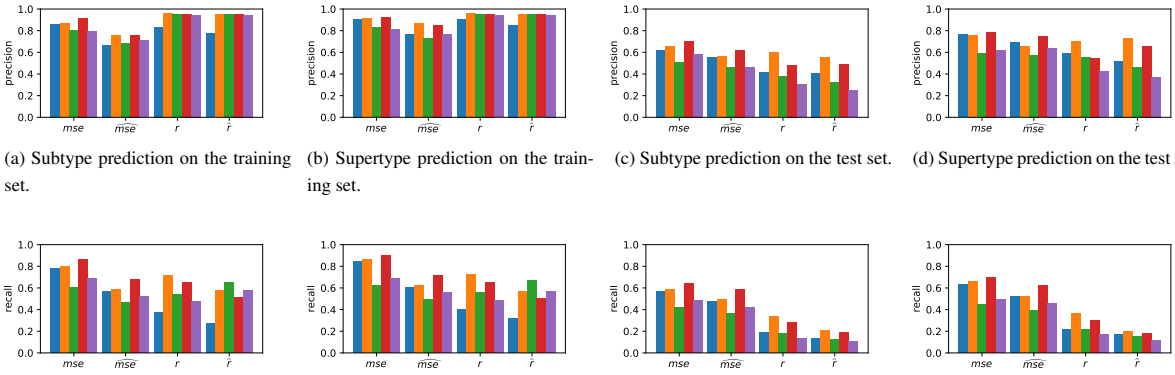
Figure 6: Number of samples for each costume subtype. Each bar represents a subtype; same supertypes are presented in the same color. Subtype N is constituted of negative samples.

5 EXPERIMENTS

5.1 Conditions

5.1.1 Dataset

From our dataset we take a subset of 26 largest costume subtypes that have six or more samples. This subset constitutes positive samples for costume subtypes to be learned. Sample collections in this subset are split into a training and a test sets with a ratio of 2:1. From the training set, descriptors are computed to be matched against the samples from the test set. The remaining classes in the dataset with fewer samples will never be learned by the descriptors, and, thus, must be rejected from classification as unknown costumes. We use them as negative samples for classification, under the condition that these samples must not belong to the same costume supertype as the positive samples, in order to avoid conflicts during supertype classification. These negative samples are appended to both, the training and test sets. The distribution of sets per costume subtype is presented in Figure 6.



(a) Subtype prediction on the training set.
(b) Supertype prediction on the training set.
(c) Subtype prediction on the test set.
(d) Supertype prediction on the test set.
(e) Subtype recall on the training set.
(f) Supertype recall on the training set.
(g) Subtype recall on the test set.
(h) Supertype recall on the test set.

Figure 7: Comparison of average precision and recall for each similarity metric using color spaces RGB (blue), YIQ (orange), 0IQ (green), L*a*b* (red), 0a*b* (purple).

5.1.2 Rejection Threshold

A classifier rejects a sample from classification when its similarity has fallen under its rejection threshold. This threshold is placed to balance out the precision and recall of this classifier, usually maximizing the F-score. In our work, we aim to reduce the Type-II error, and, thus, maximize the $F_{0.5}$-score. This threshold is computed for each descriptor separately. The cumulative threshold of every descriptor in a costume subtype v, we denote a classifier threshold θ^v.

5.1.3 Parameters

Originally, all images in the dataset are stored as RGB values. For our experiments, we convert these images into four additional color spaces mentioned in Section 4.8: YIQ, 0IQ, L*a*b* and 0a*b*.

We compute subtype descriptors using the aforementioned similarity metrics *mse* and *r*. Additionally, we prepare versions of these methods with a disabled displacement, $\widehat{mse}$ and $\hat{r}$, in order to evaluate the advantage of allowing for transformation during similarity calculation.

With each combination of color space and similarity metric, we produce a classifier model. This model includes the trained descriptors and their rejection thresholds.

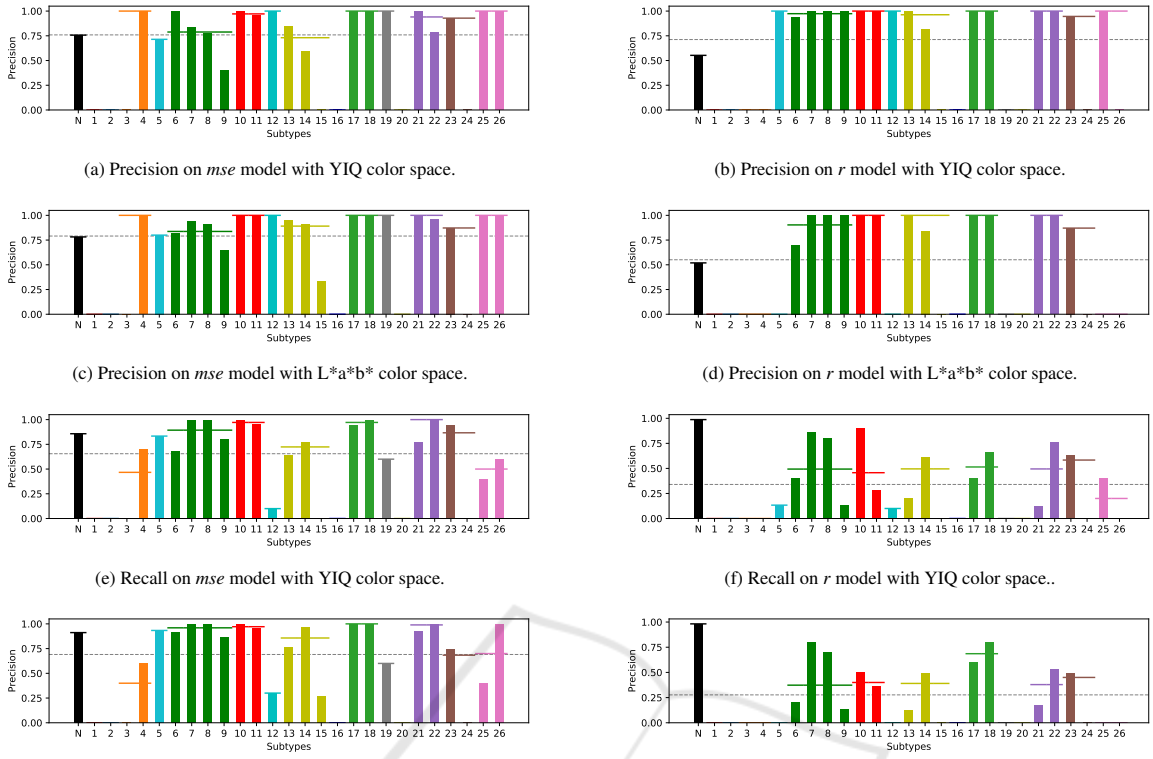
5.2 Experiment Pipeline

First, we use the patches from the positive samples to produce the descriptors, as previously described in Section 4.6. These descriptors are then used to determine the similarity of the corresponding patches in the training set. Based on the predictions, the rejection threshold for each descriptor is determined by maximizing the $F_{0.5}$-score. Having the six descriptors and their rejection thresholds for each costume subtype, we combine them into a costume classifier, which we use to predict the similarity of a sample to a particular costume subtype.

Afterwards, we predict the samples in the test sets. A sample is classified by determining its similarity to every costume subtype, as described in Section 4.7. If the similarity is lower than the threshold of a subtype classifier – the sample is rejected by it. When a sample has been rejected by every classifier, it is labeled as rejection class. If the classifier has not been rejected by at least one classifier, the subtype $\hat{v}$ with the highest similarity determines the subtype of the sample. Since our goal is to determine the costume supertype, we map the subtype prediction to its supertype.

5.3 Evaluation

In Figure 7 we present average prediction results of the training and test sets. Obviously, supertype classification results are always better, and classification precision dominates over recall, as expected from the maximized $F_{0.5}$-score. The models trained with similarity metrics r and $\hat{r}$ show better precision on the training set than the ones trained with *mse* and $\widehat{mse}$, albeit presenting lower recall. Both, precision and recall, drop on the test set, indicating a high bias of the r and $\hat{r}$ models. However, in the YIQ and L*a*b* color spaces the drop in precision is a little less prominent. Surprisingly, models trained with similarity metric $\hat{r}$ with color spaces without the brightness data, 0IQ and 0a*b*, show same level of training set prediction precision as other models with the same similarity metric, even a higher recall.



(a) Precision on *mse* model with YIQ color space.

(b) Precision on r model with YIQ color space.

(c) Precision on *mse* model with L*a*b* color space.

(d) Precision on r model with L*a*b* color space.

(e) Recall on *mse* model with YIQ color space.

(f) Recall on r model with YIQ color space..

(g) Recall on *mse* model with L*a*b* color space.

(h) Recall on r model with L*a*b* color space.

Figure 8: Evaluation of per-subtype test set classification. Lines spanning multiple bars of the same color mark supertype classification performance for each class. Dashed line indicates an average score. The color code corresponds to that in Figure 6.

5.3.1 Average $F_{0.5}$-scores

In Tables 1 and 2 we compare the $F_{0.5}$-scores achieved with our models. The similarity metric *mse* presents the best average performance in both supertype and subtype classification. The highest $F_{0.5}$-score is achieved with the color space L*a*b*. YIQ and RGB perform just a little poorer due to a lower recall. Models trained with r perform better only in training set classification because of the descriptor bias.

Our method of descriptor construction with displacement provides an improved subtype prediction $F_{0.5}$-score by 0.15 on the training set and 0.08 on the test set on *mse* models. In r-models this improvement is lower: at around 0.06 and 0.05. For supertype prediction these scores are only a little reduced with 0.09 and 0.12 in YIQ, and 0.1 and 0.05 in L*a*b* color spaces for the *mse*-models. In r-models this rise is comparable to subtype classification with 0.07 and 0.04 for both, YIQ and L*a*b* color models. It is interesting to note that in the r-models, the $F_{0.5}$-score deteriorates slightly on the training set in color spaces without luminosity data, presumably taking away some of classifier bias.

5.3.2 Performance per Class

We now focus on the two color spaces with the most promising classification performance – YIQ and L*a*b*. Classification results of *mse* and r models

Table 1: $F_{0.5}$-scores achieved during costume subtype classification.

color	*mse*		$\widehat{mse}$		r		$\hat{r}$	
space	train	test	train	test	train	test	train	test
RGB	0.85	0.61	0.65	0.54	0.67	0.34	0.57	0.29
YIQ	0.85	0.64	0.72	0.55	0.90	0.52	0.84	0.41
0IQ	0.76	0.49	0.63	0.44	0.83	0.31	0.87	0.25
L*a*b*	0.91	**0.69**	0.74	0.62	0.87	0.42	0.81	0.37
0a*b*	0.77	0.56	0.66	0.45	0.79	0.24	0.84	0.19

Table 2: $F_{0.5}$-scores achieved during costume supertype classification.

color	*mse*		$\widehat{mse}$		r		$\hat{r}$	
space	train	test	train	test	train	test	train	test
RGB	0.89	0.74	0.73	0.65	0.72	0.44	0.64	0.37
YIQ	0.90	0.74	0.81	0.62	0.91	0.59	0.84	0.48
0IQ	0.78	0.56	0.67	0.53	0.83	0.42	0.88	0.33
L*a*b*	0.92	**0.77**	0.82	0.72	0.88	0.47	0.81	0.43
0a*b*	0.78	0.59	0.72	0.59	0.79	0.33	0.83	0.26

are shown in Figure 8 for each subtype. It is apparent that both, *mse* and *r* models have failed to learn some of the subtypes. Nevertheless, the *mse* model was able to distinguish more classes in both color spaces compared to the *r* model. The *mse* model also presents a more balanced performance in terms of precision and recall.

Low classification precision of negative samples in *r* models indicates that many positive samples have been rejected during classification and assigned to the rejection class. This is, again, explained by the classifier bias. Furthermore, this assumption is confirmed by the high recall score of the rejection class, meaning that most of true negative samples have been classified correctly and low precision is a result of erroneously rejected positive samples.

Note that supertype classification still holds a high overall performance, even when some subtypes have not been recognized. This means that samples in those subtypes have been assigned to a sibling subtype, contributing to a better supertype classification overall. It is worth noting, that only supertypes with just one subtype have not been recognized. Poor classification performance also correlates with low amount of samples available for each class.

6 CONCLUSION

It is evident that the Swiss traditional costume dataset is desperately small, but this is also the rationale for this work. We use poselets, similar to (Chen et al., 2012), to define reproducible features that cannot be located visually. We also propose to compute descriptors for features by iteratively merging sample images of these features, while allowing for displacement during pair-wise comparisons. We demonstrate that the $F_{0.5}$-score of *mse*-models computed with displacement increases by 0.07-0.12 on the test set, compared to $F_{0.5}$-score without displacement. This model performs best in L*a*b* color space on both, subtype and supertype costume classification.

REFERENCES

Cao, Z., Hidalgo, G., Simon, T., Wei, S.-E., and Sheikh, Y. (2018). OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. In *arXiv preprint arXiv:1812.08008*.

Cao, Z., Simon, T., Wei, S.-E., and Sheikh, Y. (2017). Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*.

Chen, H., Gallagher, A., and Girod, B. (2012). Describing clothing by semantic attributes. In Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., and Schmid, C., editors, *Computer Vision – ECCV 2012*, pages 609–623, Berlin, Heidelberg. Springer Berlin Heidelberg.

Eichner, M., Marin-Jimenez, M., Zisserman, A., and Ferrari, V. (2012). 2d articulated human pose estimation and retrieval in (almost) unconstrained still images. *International journal of computer vision*, 99(2):190–214.

Kalantidis, Y., Kennedy, L., and Li, L.-J. (2013). Getting the look: clothing recognition and segmentation for automatic product suggestions in everyday photos. In *Proceedings of the 3rd ACM conference on International conference on multimedia retrieval*, pages 105–112. ACM.

Liu, Z., Luo, P., Qiu, S., Wang, X., and Tang, X. (2016). Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1096–1104.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee.

Varma, M. and Zisserman, A. (2005). A statistical approach to texture classification from single images. *International journal of computer vision*, 62(1-2):61–81.

Wei, S.-E., Ramakrishna, V., Kanade, T., and Sheikh, Y. (2016). Convolutional pose machines. In *CVPR*.

Yamaguchi, K., Kiapour, M. H., Ortiz, L. E., and Berg, T. L. (2012). Parsing clothing in fashion photographs. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3570–3577. IEEE.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS