Quadruplet Network for Attribute Recognition and person Re-ID

Dario Fabiani
University of Trento
dario.fabiani@studenti.unitn.it

Giacomo Lazzerini
University of Trento
giacomo.lazzerini@studenti.unitn.it

Abstract

This paper illustrates how the team dealt with the challenges of Attribute Recognition and Person Re-ID. We made a review of the literature regarding these topics, took inspiration from them, and developed our model. We trained end-to-end a Quadruplet Network for both Attribute Recognition and Person Re-ID task.

1. Introduction

In the last few years Attribute Recognition and Person Re-Identification (Person re-ID) have been two widely field of studies. [9] Attribute Recognition regards the identification of a set of attributes of an image, such as accessories, type and colour of clothes, gender, etc. Person re-ID regards the image retrieval of a person queried that was captured by different cameras. The query person we are interested in may be given by an image, frame of a video, or text description[3]. In this project the re-ID is configured as an image retrieval problem. For the attribute recognition part, an attribute is defined as a visual property that appears or disappears in an image, it can be divided in global and local features. For global it is intended a holistic property of an image, e.g. "age", that describes the whole person. Local attributes are used to describe a part of the object, e.g. "bag", describing a small neighbourhood around a point on the object [1]. The architecture we proposed is a network that can perform both tasks. The attribute Recognition is a multi-class classification of the attributes, while the person re-ID one is an image retrieval problem. Our model achieves to consider the pedestrian attributes for the re-ID problem. The paper is structured as follows. Section 2 describes data and major file manipulations. Section 3 analyses in detail the architecture. Last sections will give results and final conclusions.

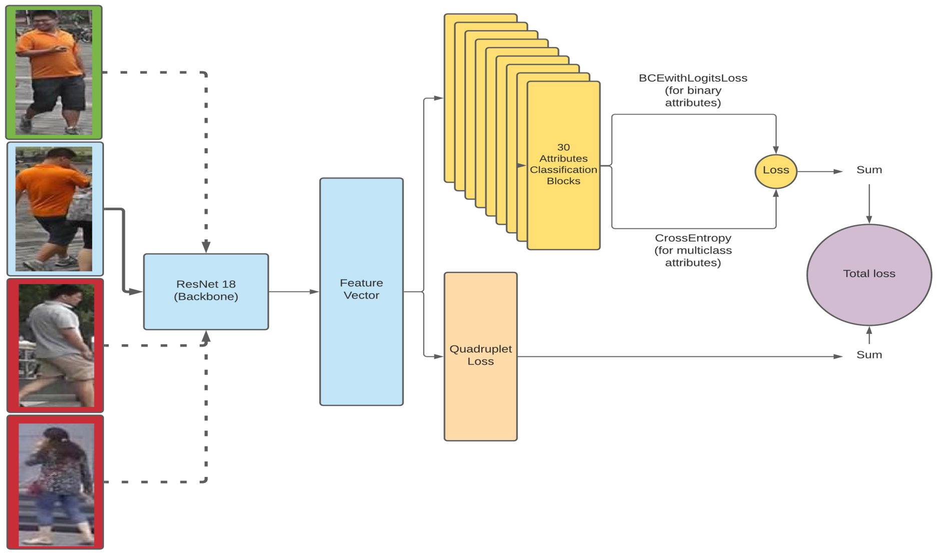
2. Data

The dataset used in this project is the Market-1501, which contains 64x128 px images collected by six cameras placed in front of a campus supermarket. It is divided into Train Test sets. The first one includes 751 distinct person identities, while the second one 750, with in addition a set of distractors and a set of junk images. Moreover, a Queries set is provided to test the person re-ID task, and an annotations train.csv file, for the classification of the attributes, are provided. The latter contains the 27 hand-written attributes for each identity. After uploading the file, we modified it to take into account the case where the colour of the upper-body clothing was not provided, inserting a new column called "upmulti"(8 colors plus the new one), the same goes for the lower-body clothing colour with a new column called "downmulti" (9 colours plus the new one).Then, the final evaluation file, that we called "classification test.csv", has 2 columns more than the original. We split the Train set into Training, with 541 identities, and Validation set, with the remaining 210. To evaluate the Persone re-ID we split the Validation set into two more sets, another validation set, 187 identities, and the query set, 23 identities, one for each identity in the second validation set. Finally, the second evaluation file has been created called "reid test.txt", that has for each query image a set of predictions of the model.

3. Proposed Method: Quadruplet Network

To achieve the two main tasks of Attribute Recognition and Person re-ID we have developed an end-to-end architecture with shared structures. Our network has the purpose to learn the pedestrian ID embedding and at the same time predicts the pedestrian attributes. [4] Exploiting the tasks in the same architecture have led us towards a "local-global feature representation", since after a multi-class classification of the attributes, we used the attributes predictions as further indication for the image retrieval part. According to different studies, the combination of an attribute recognition network and a similarity loss can improve the overall performance [5, 8, 7].

The architecture of the proposed network is shown in Fig. 1. It accepts image quadruplets $\{x_a, x_p, x_{n_1}, x_{n_2}\}$ i.e. an anchor image (x_a), a positive image (x_p) from the same class as the anchor, and two different negative images (x_{n_1}, x_{n_2}) from a different class and applies the same At-



ResNet 18 (Backbone)
Feature Vector
30 Attributes Classification Blocks
BCEwithLogitsLoss (for binary attributes)
CrossEntropy (for multiclass attributes)
Loss
Sum
Total loss
Quadruplet Loss
Sum

Figure 1. Quadruplet Network architecture

tribute Recognition Network for each of them. The network is trained end-to-end using the sum:

Total Loss

$$L_{\text{total}}(\theta) = L_{quad}(\theta) + \lambda L_{attr}(\theta) \tag{1}$$

Where $L_{quad}(\theta)$, $L_{attr}(\theta)$ are respectively the Quadruplet loss and the Attribute loss and λ is a constant that balances the contribution between the two losses. We chose to set this hyparameter to 0.8.

3.1. Attribute Recognition Network

The network uses a pre-trained *ResNet18* as a backbone to which we have attached a classification block for each attribute. Each of the 29 classification blocks consist of a series of concatenated layers:

- Bottleneck layer, introducing a layer of 256 neurons.
- BatchNormalization regularization function.
- LeakyRelu activation function (in this way we avoid the so called "Dead Relu", since it has a negative slope of 0.01).
- Dropout, to help the network not to overfit.
- $(n = class)$-size fully connected layer.

The size of this last layer depends on the size of the class (e.g. *age* = 4, *backpack* = 1). We do not use an activation function to extract probabilities at this stage (Sigmoid for binary or Softmax for multiclass). We implement those activation functions within the loss calculation, using the Pytorch class *torch.nn.BCEwithLogitsLoss()* and *torch.nn.CrossEntropyLoss()*. In this way we seek numerical stability since we take advantage of the log-sum-exp trick for numerical stability [6]

4. Quadruplet Training

During the training, the same Attribute Recognition Network is used to extract the feature vector from each of the four images (*anchor, positive,* $negative_1$, $negative_2$). The four feature vectors are then used to compute the Quadruplet Loss which is a special case of the Triplet Loss. The triplet loss is normally trained on a series of triplets $\{x_a, x_p, x_n\}$, where x_a and x_p are images from the same person, and x_n is from a different person. The triplet loss is designed to keep x_a closer to x_p than x_n. It is formulated as following:

Triplet Loss

$$L_{trp} = \sum_{a,p,n}^{N} \left[\|f(x_a) - f(x_p)\|_2^2 - \|f(x_a) - f(x_n)\|_2^2 + \alpha_{trp}\right]_+ \tag{2}$$

where $[z]_+ = \max(z, 0)$, and $f(x_a), f(x_p), f(x_n)$ mean features of three input images (anchor, positive and negative). A margin α is added to make sure the neural network's gradient disregards abundant far (easy) negatives and leverages scarce nearby (hard) negatives.

Triplet loss trains the model only based on the relative distances between positive and negative pairs with respect to the anchor, the Quadruplet Loss introduces a new constraint which pushes away negative pairs from positive pairs. To do this, anther negative image x_{n_2} is used to compute the loss on the quadruplet: $\{x_a, x_p, x_{n_1}, x_{n_2}\}$.

Quadruplet Loss

$$\begin{aligned} L_{\text{quad}} = & \sum_{a,p,n_1,n_2}^{N} \left[\|f(x_a) - f(x_p)\|_2^2 - \|f(x_a) - f(x_{n_1})\|_2^2 + \alpha_1 \right]_+ \\ & + \sum_{a,p,n_1,n_2}^{N} \left[\|f(x_a) - f(x_p)\|_2^2 - \|f(x_{n_1}) - f(x_{n_2})\|_2^2 + \alpha_2 \right]_+ \\ & s_a = s_p, s_a \neq s_{n_1}, s_a \neq s_{n_2}, s_{n_1} \neq s_{n_2} \end{aligned} \tag{3}$$

where α_1 and α_2 are the values of margins in two terms and s_a refers to the person ID of image x_a. More details regarding the quadruplet loss can be found here [2].

During the optimization process, we used Stocastic Gradient Descend as an optimizer with learning rate = 0.01, a momentum of 0.5 and a weight decay of 10^{-6}.

5. Implemented classes

5.1. `class MarketDataset()`

This class takes an image from a given dataframe, either from the training, validation or queries, and apply to it the transformations we have chosen based on the phase. If we are dealing with the train set it will return three elements. The first element is an image as a tensor, which is called *`anchor_image`* since it is also the anchor for the quadruplet. The second element *`attributes`* is a list of the attributes for that anchor image, and the third element is *`quadruplet`* as a list of tensors of three other images (a positive image, and two negative images). Otherwise, if it is not a train set it will return only the *`anchor_image`*.

5.2. `class ClassificationBlock(nn.Module)`

This class is used to generate a Classification Block for each attribute attached to the end of our Backbone. The classification is composed of an additional fully connected layer with only 256 neurons, where we linearize the input features, then we apply a BatchNormalization, a LeakyRelu with a negative slope of 0.01, and a Dropout with probability 0.5. Finally, we pass it to another fully connected layer with a dimension equal to the number of classes of the attribute. The output of the classification block is this final linearized layer.

5.3. `class Backbone (nn.Module)`

This class takes the pre-trained *Resnet18* model and attach the Classification Blocks for each of the 29 attributes. The class returns three elements. The first one *`raw_pred_label`* is a list of linearized layers, passed through the classification blocks, which is used in the `class OverallWrapper()`. The second element *`prob_pred_label`* is used to compute the accuracy for each attribute, and to create the final *classification_test.csv*, it returns a list of tensors (with probabilities) with the linearized layers passed through a Sigmoid (for binary attributes) or a Softmax. The third element x is the linearized layer of the image returned by the backbone.

5.4. `AttributesLoss (nn.Module)`

This class simply applies to the first element of the output of the Backbone the *`nn.BCEwithLogitsLoss()`* to the binary classes and the *`nn.CrossEntropyLoss()`* to the others. At the end, it returns a sum of the all attribute losses.

5.5. `QuadrupletLoss (nn.Module)`

This class compute the Quadruplet Loss. It takes 4 inputs, the anchor, a positive image, which is an image with the same ID of the anchor, and two other negative images, that do not share the identities with the anchor.

5.6. `QuadrupletLoss (nn.Module)`

It takes the `AttributesLoss` and `QuadrupletLoss` and combine their results in a sum guided by a hyper-parameter λ, set to 0.8.

6. Evaluation and Results

During training and validation, we logged different statistics using TensorBoard. On each epoch the accuracy for each attribute and the total accuracy was computed. The first task is the one that has achieved remarkable results. We have trained our model for 50 epochs, the final total accuracy is 85.13% with a maximum of 99.88 for downpurple and a minimum of 58.31 for gender. We use Mean Average Precision (mAP) as a person Re-ID evaluation metric. This statistic has been computed with the ground truth of the *validation-query set* and the *validation-test set*, and the predicted top k=15 similar images for each query image. For this second task the mAP shows that the network is not able to retrieve an image of the person queried. The average value for the statistic is around 0.001. This may be cause by a wrong implementation of the Quadruplet, that usually takes the images using a Batch Hard techniques, not from the entire dataframe, and do not take the "most" negative ones, but simply an image with a different identity.

7. Conclusions

This project allowed us to deal with some classical problems of computer vision: Attribute Recognition and Person Re-identification. It has given us the chance to work

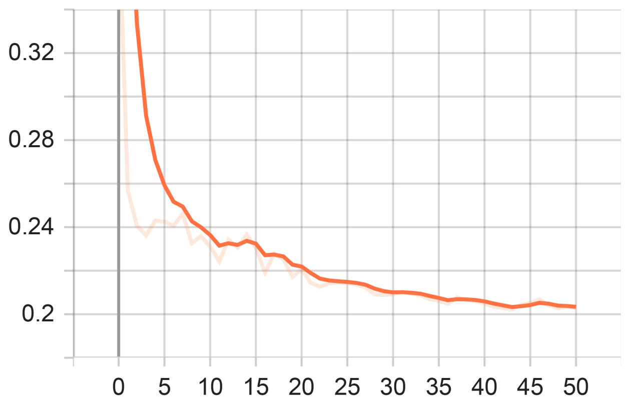


Figure 2. Cumulative Training Loss over 50 epochs

on two tasks that led us to deepen the deep learning landscape, and how different can be according to the approach adopted. The analysis of different papers, codes and networks has shown us a glimpse of how broad the field can be, thanks to the different idea we deployed. In our project we developed a network trying to combine the most appealing ideas. However, it seems that it can properly accomplish only the first task (Classification task) while it needs some adjustment for the second one (Re-identification task).

References

[1] Abrar H. Abdulnabi, Gang Wang, Jiwen Lu, and Kui Jia. Multi-task cnn model for attribute prediction. *IEEE Transactions on Multimedia*, 17(11):1949–1959, 2015. 1

[2] Weihua Chen, Xiaotang Chen, Jianguo Zhang, and Kaiqi Huang. Beyond triplet loss: A deep quadruplet network for person re-identification. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1320–1329, 2017. 3

[3] Theodoros Georgiou, Yu Liu, Wei Chen, and Michael Lew. A survey of traditional and deep learning-based feature descriptors for high dimensional data in computer vision. *International Journal of Multimedia Information Retrieval*, 9(3):135–170, Sep 2020. 1

[4] Yutian Lin, Liang Zheng, Zhedong Zheng, Yu Wu, Zhilan Hu, Chenggang Yan, and Yi Yang. Improving person re-identification by attribute and identity learning. *Pattern Recognition*, 95:151–161, 2019. 1

[5] Dipu Manandhar, Muhammet Bastan, and Kim-Hui Yap. Tiered deep similarity search for fashion. In Laura Leal-Taixé and Stefan Roth, editors, *Computer Vision – ECCV 2018 Workshops*, pages 21–29, Cham, 2019. Springer International Publishing. 1

[6] Pytorch. BCEwithLogitsLoss. `https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html?highlight=bcewit#torch.nn.BCEWithLogitsLoss`, 2021. [Online; accessed 30-August-2021]. 2

[7] Shun Zhang, Yantao He, Jiang Wei, Shaohui Mei, Shuai Wan, and Ke Chen. Person re-identification with joint verification and identification of identity-attribute labels. *IEEE Access*, 7:126116–126126, 2019. 1

[8] Xiaofan Zhang, Feng Zhou, Yuanqing Lin, and Shaoting Zhang. Embedding label structures for fine-grained feature representation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1114–1123, 2016. 1

[9] Liang Zheng, Yi Yang, and Alexander G. Hauptmann. Person re-identification: Past, present and future. *CoRR*, abs/1610.02984, 2016. 1