NATIONAL TAIWAN UNIVERSITY

Deep Learning for Computer Vision - Tiger Re-Identification

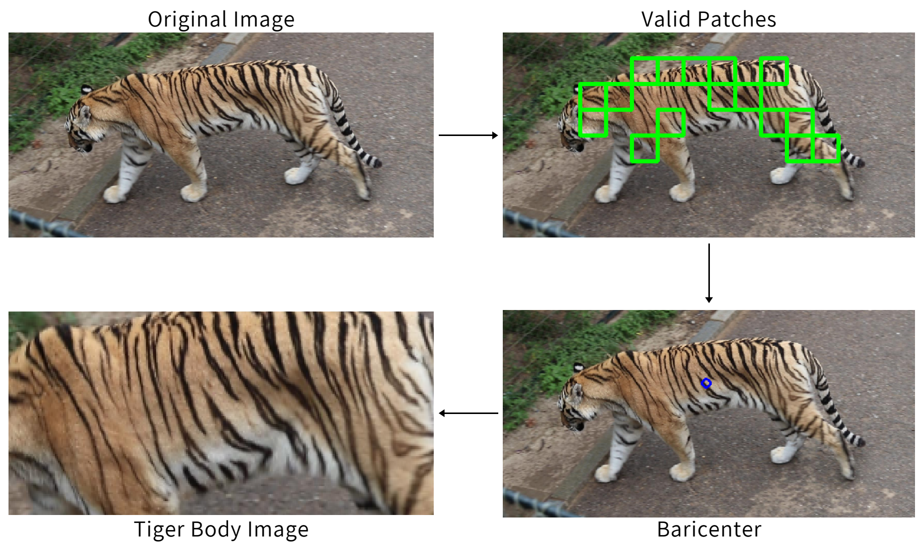
Ricardo Manzanedo Andrea Agiollo Alberto Rech

INTRODUCTION

Re-identification is a very challenging task in the computer vision field due to the changes in camera view and background. Moreover tiger re-id is even more challenging due to the similarity between different tigers of the same specie.
For this task we decided to implement a model based on the global and local appearence of tiger images. This idea was inspired by the fact that most state-of-the-art models [1], [2] implement a similar structure to achieve best results.
Moreover since this task presents lot of similarities with person re-identification we implemented also some tricks used in [3] to obtain better performances.

BODY EXTRACTION

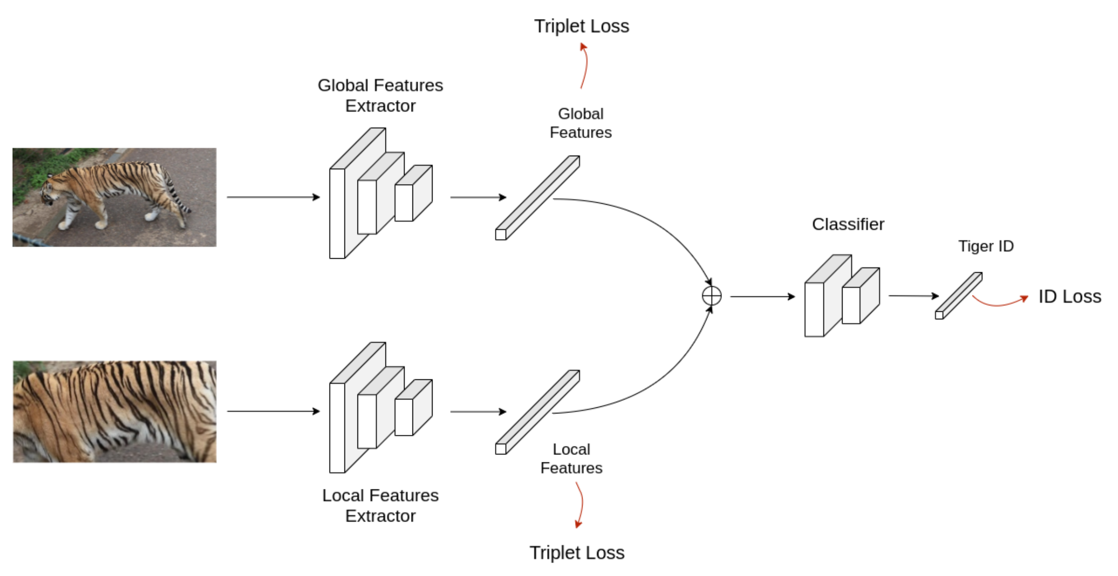
In order to achieve great results our model bases its decision on the combination of global information, i.e. the whole image, and local information. To extract this local information we developed a reliable preprocessing mechanism to extract the body of the tiger from the original image. The proposed solution to this problem is a reliable algorithm that works with color and ORB features [6] of the original image.



To start up the algorithm will divide the image in 32x32 patches, on each of these patches it will then compute the percentage of orange shaded pixels and the amount of ORB features that we can find in the patch (ORB features should correspond to corners in the image and therefore to beginning or ending of tiger stripes). Each patch will then be considered valid if the amount of orange pixels surpass a certain threshold and the number of ORB features is more than a certain percentage of the number of ORB features of the whole image.
These two conditions should be satisfied if the considered patch belongs to the body of the tiger.
From the valid patches we can then compute the baricenter of their position which will be then considered as the center of the tiger body. From the baricenter we can then crop the region (with a fixed shape) that will result being the body of the tiger that will be used for the local appereances of the tiger.

MODEL STRUCTURE

The model we implemented is structured as shown in the following figure:



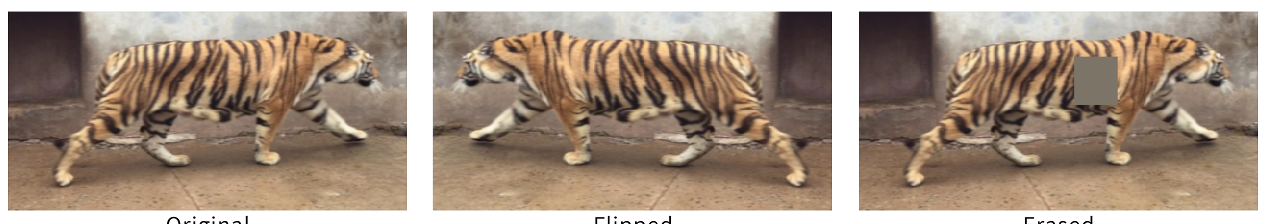
There are two different backbones, the first is used to extract the global features from the whole image, the second is used to extract the local features from the body of the tiger. Once the features are extracted the two triplet losses are computed while the two features vectors are summed together to obtain the overall features vector. Then a simple classifier is used in order to derive the tiger IDs and compute the cross-entropy loss. The sum of triplet loss and cross entropy loss is used for the backward propagation step.
Since local features are used as regularizer, they will be used only in the training process as done in [1], [2].

The triplet loss function [7] is defined as follows:

$$T(X) = \sum_{i=1}^{P}\sum_{a=1}^{N}[m + \max_{p=1\ldots N} D(f(x_a^i), f(x_p^i))) - \min_{p=1\ldots N, n=1\ldots N, i\neq j} D(f(x_a^i), f(x_n^i))]$$

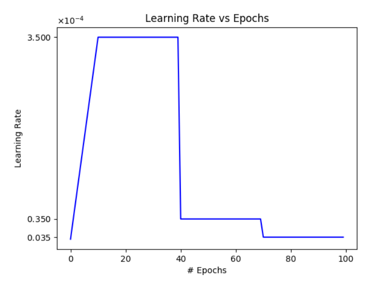
where X is a training batch with P tigers and N images per tiger, $f(.)$ is the output feature map of the network, x_j^i is the j-th image of the i-th tiger, $D(.,.)$ is the Euclidian distance function and m is the margin parameter. This loss finds the pair of images of the same tiger with maximum distance and the pair of images of different tigers with minimum distance and guides the model to make the difference between these two at least equal to the margin m.

BAG OF TRICKS



As done in [3] we implemented some tricks for the training of our model to obtain better performance. The bag of tricks that we implemented contains some techniques regarding the preprocessing part, some regarding the training and some regarding loss computation.

- Preprocessing:
  - **random flip**: the image will be horizontally flipped with a certain probability (e.g. 0.5)
  - **random erasing**: the pixels corresponding to a certain portion of the image will be set to a fixed value (e.g. the average of the dataset)
  - **square images**: images will be resized to a size of 256x256
- Training:
  - **learning rate with warm up**: the learning rate will follow the behaviour of the plot on the right
  - **batch normalization before classification**: a batch normalization layer will be added to help the classifier
- Loss Computation:
  - **label smoothing cross entropy loss:** label smoothing is performed to change the ground-truth labels distribution. In this way we can make the model more adaptable by adding prior distribution over the labels. This kind of loss helps to prevent overfitting issues, the loss function is defined as follows:



$$H(q', p) = -\sum_{k=1}^{K} \log p(k) q'(k) = (1-\epsilon)H(q,p) + \epsilon H(u,p)$$

where ϵ is a regularizer value, p(k) is the output of the model , q and u are respectively the groundtruth and the uniform distribution and q' is defined as: $q' = (1-\epsilon)q(k) + \frac{\epsilon}{K}$

RESULTS

DenseNet121 + Bag of tricks

Batch Normalization	59.64%
Random flipping	61.75%
Random erasing	63.25%
Warm up learning rate	65.21%
256x256	65.21%
512x256	62.35%

DenseNet121 + Loss

ID Loss only	59.49%
Triplet Loss only	60.39%
Triplet Loss + ID Loss	65.21%

To find out which combination of model and tricks gives the highest rank-1 accuracy we performed the training combining the different possibilities. The upper tables show the influence of different tricks and losses. From this it is possible to see that the best combination involves using all the tricks and the summation of triplet and cross-entropy loss. In the first, the margin parameter has been set to m=1, in the second the regularizer parameter has been set to 0.1 for each experiment.

Different Networks

ResNet50	60.24%
VGG16	61.14%
DenseNet121	65.21%

Different Paradigms

Single backward	64.16%
Multiple split backward	65.21%

Moreover we tested the influence of different backbones on the proposed model and the impact of computing the backward step on the total loss or having a single backward step for each backbone, results are shown in the above tables.
For all the experiments Adam optimizer was used while all the backbones were pre-trained on the "ImageNet" dataset.
Analysing all the results we can see that the best model for this specific task is DenseNet121 that allows, using different tricks, to achieve rank-1 accuracy 65.21%

CONCLUSIONS

In this work we designed and presented a strong model to solve the tiger re-identification task. Having conducted different experiments with different parameters, tricks and models, we found out that the best backbone for solving this task is given by DenseNet121. We were also able to show the importance and the effectiveness of the tricks cited in this poster.
Moreover we showed the significance of the local features and their combination with global features, their use in fact played an important role in the training process letting us achieve acceptable rank-1 accuracy.
Finally, it must be highlighted also the difference in the achieved accuracy between the different paradigms that we showed.

REFERENCES

[1] J. Yu, H. Su, J. Liu, Z. Yang, Z. Zhang, Y. Zhu, L. Yang, B. Jiao - A Strong Baseline for Tiger Re-ID and its Bag of Tricks
[2] C. Liu, R. Zhang, L. Guo - Part-Pose Guided Amur Tiger Re-Identification
[3] H. Luo, Y. Gu, X. Liao, S. Lai, W. Jiang - Bag of Tricks and A Strong Baseline for Deep Person Re-identification
[4] R. Quispe, H. Pedrini - Improved person re-identification based on saliency and semantic parsing with deep neural network models
[5] G. Wang, Y. Yuan, X. Chen, J. Li, X. Zhou - Learning Discriminative Features with Multiple Granularities for Person Re-Identification
[6] E. Rublee, V. Rabaud, K. Konolige, G.R. Bradski - ORB: An efficient alternative to SIFT or SURF
[7] A. Hermans, L. Beyer, B. Leibe - In Defense of the Triplet Loss for Person Re-Identification