Triplet Descriptor

Yao Lei Xu
YLX15
01062231

Abstract

This report explores the design, training and evaluation of deep learning architectures for the generation of image representations using the noisy HPatches dataset [1]. Specifically, the report explores a 2-step system where the first model is a encoder-decoder denoising network, while the second model generates an image descriptor and is trained as a triplet network. The performance is measured against verification, matching and retrieval tasks using mean average precision as metric.

1. Dataset Description

The dataset used for this experiment is a noisy version of the HPatches [1]. The dataset contains 116 sequences, where each sequence represents the same scene and contains 1 reference image and 5 target images with some photo-metric or geometric variation. For each of these sequences, 32x32 square region patches are sampled from the reference image and projected on the target images using ground-truth homographies. Finally, these patches are perturbed with geometric transformations (rotation, anisotropic scaling and translation) of various degree to simulate the noise that occurs when a detector extracts these patches [1].

2. Problem Formulation

The goal of this machine learning problem is to provide an end-to-end design of a deep learning system that can perform well in terms of patch verification, image matching and patch retrieval. Verification measures how well a descriptor can classify two patches belonging to the same scene, matching measures how well it matches two images, and retrieval measures how well it retrieves similar patches from a large sample.

The mean average precision (mAP) will be used as metric to measure the performance of above tasks, since the unbalanced nature of the data (there are more negative matches than positive ones) makes it more suitable than classical metrics such as ROC and AUC.

mAP is the mean of the average precision score AP of all Q queries, which is defined mathematically as [5]:

$$mAP = \frac{1}{Q}\sum_{q=1}^{Q} AP(q) \tag{1}$$

$$AP = \sum_{k=1}^{n} P(k)\Delta r(k) \tag{2}$$

where k is the rank of the elements ordered in term of confidence, $P(k)$ is the precision (percentage of correct positive predictions) up until k, and $\Delta r(k)$ is the change in recall (percentage of positive cases found) of consecutive items.

3. Baseline Model

3.1. Baseline Denosing Model

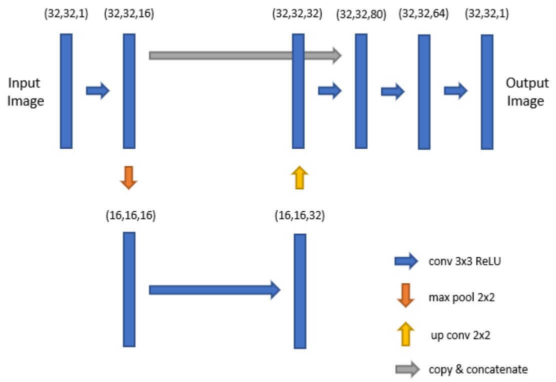

Figure 1. Baseline U-Net Architecture

The baseline model for the first part of the project consists of a shallow U-Net [3] that de-noises the input patches. This model first encodes the input images through a contracting path formed by a series of convolutional layers that captures the essential information, followed by a symmetric expanding path (hence forming a 'U' shape) with up-sampling layers that decode and reproduce the cleaned image [3] (figure 1 shows the architecture of this model).

Specifically, it is formed by a series of 3x3 kernel convolutional layers with 2x2 max-pooling layers and 2x2 upsampling layers. The activation functions are ReLU to avoid possible gradient problems in training and the weights are initialized using He's normal initialization.

Finally, the training is done using stochastic gradient descent (lr=0.00001) with Nesterov momentum (momentum=0.9), with the loss being the mean absolute error (which is more robust to outliers than the squared error) between the denoised image and the clean image (equation 3).

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i| \tag{3}$$

Figure 2 shows the training details of the network for all the data available and figure 3 shows an example of a denoised image. The training showed satisfactory results, settling with a validation MAE of 5.42.

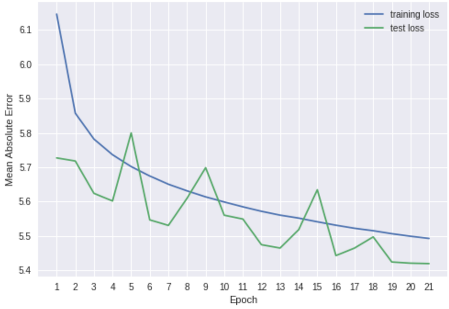

Figure 2. Baseline U-Net Training

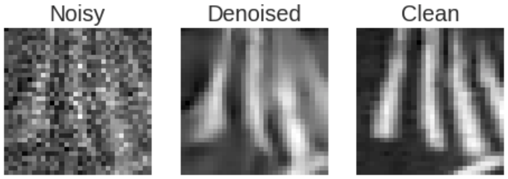

Figure 3. Baseline Denoised Image

3.2. Baseline Descriptor Model

The second part of the baseline model is a L2-Net [4] based on a series of convolutional layers that computes a descriptor vector of size 128. The architecture of this network is shown in figure 4. The network is trained with triplet loss (equation 4), which consists of generating descriptors for an anchor patch, a positive patch and a negative patch such that the squared distance between the descriptors is small for anchor-positive patches ($||\mathbf{D}_p - \mathbf{D}_a||^2$), but big for anchor-negative patches ($||\mathbf{D}_n - \mathbf{D}_a||^2$). This is therefore a metric learning method that is able to group similar images closer together, with a tolerance level set by the variable α.

$$loss = max(0, ||\mathbf{D}_p - \mathbf{D}_a||^2 - ||\mathbf{D}_n - \mathbf{D}_a||^2 + \alpha) \tag{4}$$

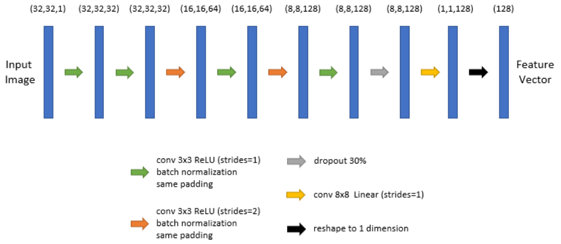

Figure 4. L2-Net Architecture

Specifically, the descriptor model uses 6 3x3 kernel convolutional layers with batch normalization, 1 dropout layer for regularization and a final convolutional layer with 8x8 kernel with output reshaped as a descriptor vector. As with the U-Net, ReLU is used for activation and the weights are initialized using He's initialization. For the training, stochastic gradient descent (lr=0.1) is used to optimize the triplet loss function. Figure 5 shows the training result of this model using clean patches and denoised patches. As expected from the training, the performance of clean patches (test loss=0.07) is much better than the denoised one (test loss=0.14), which acts as an upperbound for the performance.

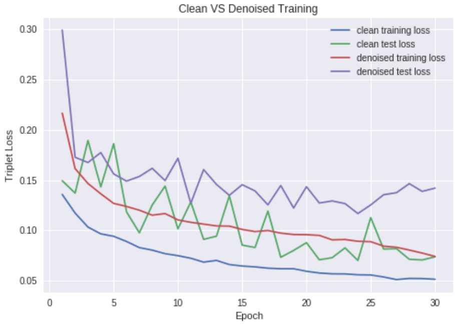

Figure 5. Baseline L2Net Training

3.3. Baseline Model Results

Patches	Verification	Matching	Retrieval
Denoised	0.8045	0.2137	0.5078
Clean	0.8904	0.3934	0.6981

Table 1. Baseline Benchmarks

By combining the above networks, the descriptor generated by the model are used to evaluate the 3 benchmarks proposed earlier, with the mAP results listed in the table above. The performance using clean patches is also provided for comparison purposes.

4. Improved Model

4.1. Improved Denoising Model

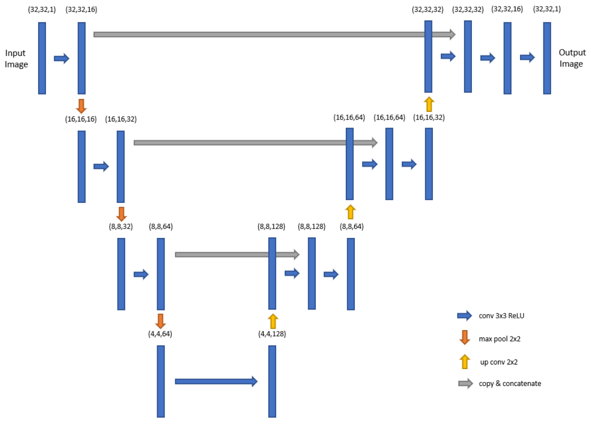

Figure 6. Improved U-Net Architecture

The baseline U-Net is a shallow model which can be further improved by simply adding additional contracting and expanding path. This should improve model's performance since more contracting layers means decreased spatial information with increased feature information, which should increase the quality of images generated after up-sampling and concatenation in the expanding layers. The deep U-Net architecture is shown in figure 6.

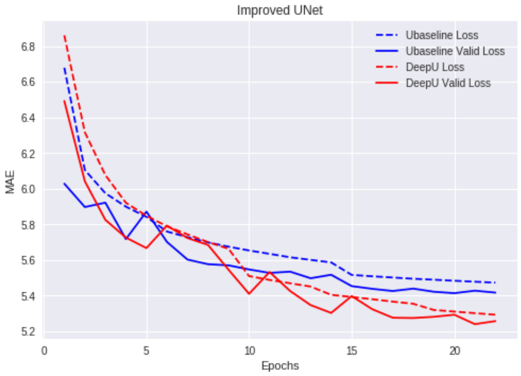

Figure 7. Deep U-Net Training

Figure 7 shows the training characteristics of this architecture compared to the baseline model. The performance improves more slowly due to higher number of model parameters, but it eventually results in better training and generalization results. Note that all other hyperparameters are kept the same to make the comparison sensible.

4.2. Improved Descriptor Model

To improve the descriptor model, 2 modifications to the triplet loss function are proposed. Firstly, the distance between descriptors are measured via the squared distance, which is less robust to outliers than the absolute distance [2]. Since this is likely to have a significant impact in a noisy environment, the first proposed improvement is to change the triplet loss function to equation 5. This is the MAD improvement.

$$loss = max(0, ||\mathbf{D}_p - \mathbf{D}_a|| - ||\mathbf{D}_n - \mathbf{D}_a|| + \alpha) \quad (5)$$

Secondly, since the baseline triplet loss primarily penalizes relative distances, it loses all the information as long as the anchor-positive distance is less than anchor-negative distance. Consider the example in figure 8 for instance, the penalty for case 2 should be greater than case 1 since the negative patch is significantly more distant, but the baseline loss function is zero for both cases.

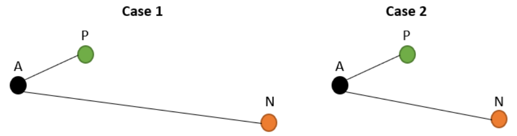

Figure 8. Triplet Loss Case Study

Therefore, the second proposed improvement is to change the triplet loss function to capture this difference. The exponential function e^x can potentially satisfy this requirement for $x = ||\mathbf{D}_p - \mathbf{D}_a||^2 - ||\mathbf{D}_n - \mathbf{D}_a||^2$, since the loss for case 2 is greater than case 1. In addition, the penalty for $||\mathbf{D}_p - \mathbf{D}_a||^2 > ||\mathbf{D}_n - \mathbf{D}_a||^2$ is much greater than the penalty for $x < 0$ so that the relative distance penalty is prioritized. However, since an exponential loss function is very unstable due to exploding gradients, the exponential-linear function with offset of 1 is used as triplet loss instead (equation 6). This loss has value $1 + x$ for $x \geq 0$ and e^x for $x < 0$. This is the ELU improvement.

$$loss = ELU(||\mathbf{D}_p - \mathbf{D}_a||^2 - ||\mathbf{D}_n - \mathbf{D}_a||^2) + 1 \quad (6)$$

Lastly, since the convolution kernels scan the pixels of given patches sequentially, they can capture locally correlated patterns that are shift invariant, but they perform poorly with to other variations such as rotation. Therefore, the last proposed improvement is to train the descriptor model with augmented images by applying random transformations (rotation, scaling, etc.) to the training sample as shown in figure 9.

However, due to memory restrictions, it isn't possible to create a massive augmented training set. For this reason, the

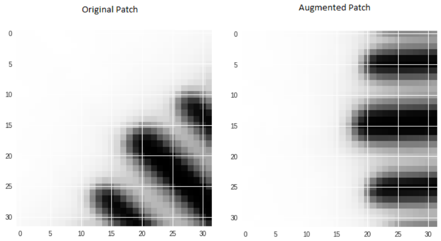

Figure 9. Augmented Patches

model weights are loaded from the baseline model , and it is further trained on a set of augmented images for 10 epochs, and it is repeated for 5 augmented sets in total. This is the augmentation improvement.

4.3. Individual Improvements Comparison

Improvement	Verification	Matching	Retrieval
None	0.8904	0.3934	0.6981
MAD	0.8846	0.3699	0.6759
ELU	0.8889	0.3951	0.7050
Augmentation	0.9043	0.4306	0.7395

Table 2. Improved Descriptors Benchmarks (Clean)

The table above compares the performance of the 3 benchmark tasks for each of the above descriptors using clean patches. Overall, most of the proposed improvements showed satisfactory results except for the MAD modification. Note that for each of the changes, all other hyperparameters are kept the same as the original baseline. In addition, the descriptor training is performed on clean patches on the assumption that an improvement on the clean patches implies an improvement on the denoised patches. Figure below shows the training curves of the above mentioned changes.

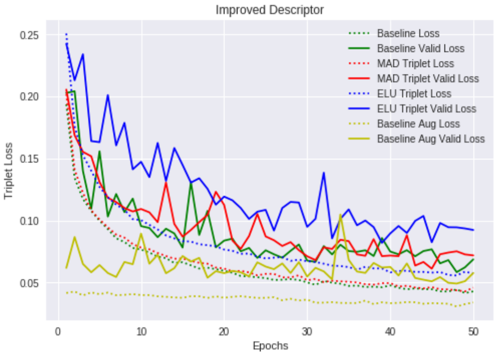

Figure 10. Descriptor Training

4.4. Final Results

The final descriptor model incorporates the ELU improvement and is trained on an augmented dataset. The performance is shown in the table below using both the patches denoised by the proposed Deep U-Net and the clean patches. Overall, the model showed significant improvement from the baseline in all three tasks.

Patches	Verification	Matching	Retrieval
Denoised	0.8521	0.2863	0.5899
Clean	0.9062	0.4365	0.7438

Table 3. Final Model Benchmarks

5. Conclusion and Future Work

This report explored several methods of generating image descriptors from a noisy dataset by modifying both the denoising network and the descriptor network, focusing mostly on deep architectures, loss function formulation and data variation. Overall, the final joint system demonstrated satisfactory results, reporting higher performance in all three benchmark tasks. However, there are many more modification that can be explored that can potentially improve the performance, such as the joint training of the two models or the training of the descriptor model using noisy images directly.

References

[1] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5173–5182, 2017.

[2] Jon F Claerbout and Francis Muir. Robust modeling with erratic data. *Geophysics*, 38(5):826–844, 1973.

[3] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[4] Yurun Tian, Bin Fan, and Fuchao Wu. L2-net: Deep learning of discriminative patch descriptor in euclidean space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 661–669, 2017.

[5] Mu Zhu. Recall, precision and average precision. *Department of Statistics and Actuarial Science, University of Waterloo, Waterloo*, 2:30, 2004.

Appendix

1. User Instruction for the Source Code

The notebook with the source code can be found on https://github.com/gylx/Deep-Learning-HPatches-Triplet-Descriptor. The notebook needs to be run on Google Colaboratory. Run the code sequentially to reproduce the results reported in this report.