EE3-25 Deep Learning - Final Report

Konstantinos Barmpas
MEng 3rd Year - Department of Electrical and Electronic Engineering
Imperial College London
konstantinos.barmpas16@imperial.ac.uk

Abstract

This report outlines a proposed neural network model for generation of image descriptors for image representation in the Euclidean Space. With these descriptors, matching, verification and retrieval tasks should be performed successfully. The baseline code consists of a shallow UNet for the de-noising of the images followed by a triple L2-Net responsible for the generation of the image descriptors using triplet loss. In the proposed improved method, the image de-noising model is an TernausNet while the descriptor generation model is the baseline L2-Net fine-tuned using Tree of Parzen Estimators(TPE) method and hard-mining for the triplet loss function. For the evaluation and comparison of the two approaches, the HPatches benchmark was used.

1. Introduction

The target of the model is the generation of learned image descriptors in Euclidean Space from noisy images. The input of the model is a 32x32 monochrome noised image ($\mathbf{x} \in R^{32x32x1}$) and the output is the descriptor, a one-dimensional 128-element vector ($\mathbf{y\_descriptor} \in R^{128x1}$).

1.1. Dataset

The HPatches dataset is based on 116 sequences. Each sequence includes a reference image and 5 target images presenting photometric changes and geometric deformations. Patches are sampled in the reference image. For our model's training and evaluation, the N-HPatches dataset is used which is a noisy version of HPatches dataset. The extracted patches were downsampled from the original size 65x65 to 32x32. 76 sequences are used for training and 40 sequences for validation. The N-HPatches dataset allows us to train and evaluate the models with noisy, denoised and clean images.

2. Baseline Approach

The baseline approach consists of two neural networks connected in series (Figure 1). To diminish the effect of the images' noise, a shallow UNet[3] is used, followed by an L2-Net[6] responsible for the descriptors' generation.

Figure 1: Baseline Approach Pipeline

2.1. Baseline Model Analysis

The shallow UNet receives a 32x32 monochrome noised image as an input ($\mathbf{x} \in R^{32x32x1}$), uses the mean absolute error ($MAE = \frac{1}{N} \times \sum_1^N |y_i - x_i|$) as the loss function and outputs the denoised image ($\mathbf{y\_clean} \in R^{32x32x1}$).

In general, a U-Net architecture consists of a contracting path to capture context and of a symmetrically expanding path that enables precise localization. The contracting path follows the typical architecture of a convolutional network with alternating convolution and pooling operations and progressively downsamples feature maps, increasing the number of feature maps per layer at the same time. Every step in the expansive path consists of an upsampling of the feature map followed by a convolution.

The shallow UNet consists of a one-stage encoding 16-channel convolutional layer, one 32-channel convolutional layer bottleneck and 64-channel convolutional layer connected with the encoding layer as the decoder. To produce the output a final one-channel convolution is used. All convolutional layers have 3x3 kernels, same padding and ReLU activations. Optimization is done by stochastic gradient descent with learning rate of 0.00001 and momentum of 0.9.

The L2-Net uses the denoised image as input and generates the image descriptor, a one-dimensional 128-element array. This array ($\mathbf{y\_descriptor} \in R^{128x1}$) is the output of the overall model.

The L2Net architecture is presented in Figure 2. Padding with zeros is applied to all convolutional layers, to preserve the spatial size, except to the final one. Batch normalization layer followed by ReLU non-linearity activation is added

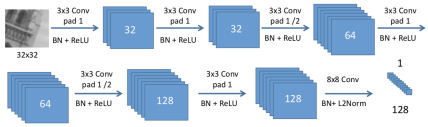


Figure 2: L2-Net architecture

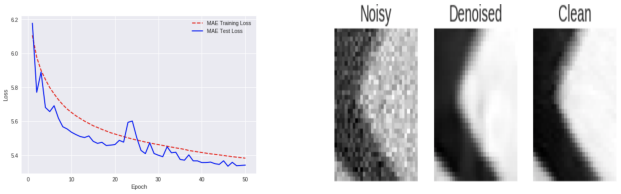
after each layer, except the last one. Dropout regularization with 0.1 dropout rate is applied before the last convolution layer. The output of the network is L2 normalized to produce a one-dimensional 128-element descriptor.

The L2-Net uses the triplet loss function which takes an anchor patch, a negative patch and a positive patch $TripletLoss = \frac{1}{N} \times \sum_1^N max(0, 1 + d(a,p) - d(a,n))$.

The network is trained by three parallel L2-Net instances such that the anchor and positive patch descriptors have the possible minimum Euclidean distance between them, and the negative and anchor patch descriptors have the possible maximum Euclidean distance. Optimization is done by stochastic gradient descent with learning rate of 0.1.

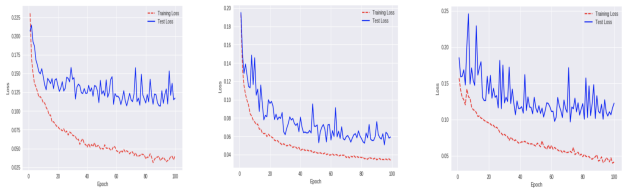
2.2. Baseline Training

Using the baseline code, we trained the overall model aiming to reach the maximum of its potential. For that purpose the whole dataset was used during training. The shallow UNet was trained for 50 epochs and the L2-Net for 100 epochs.



(a) UNet Learning Curve (b) Denoise Images Result

Figure 3: Figures (left to right): (a) Shallow UNet learning Curve for training and (b) Denoise images results



(a) Noisy Patches (b) Denoised Patches (c) Clean Patches

Figure 4: L2-Net learning Curves for training with (left to right) noisy, denoised and clean patches and evaluate with denoised. For denoiser we used the final shallow UNet model.

2.3. Baseline Evaluation

The performance of the network is determined by the mean average precision (mAP) on three different tasks: Patch Verification, Image Matching and Patch Retrieval using the HPatches Benchmark [4]. By training with the noisy and clean data, we could determine the lower and upper bounds.

Metric	Clean	Denoised	Noised
Patch Verification	0.839	0.827	0.632
Image Matching	0.271	0.254	0.202
Patch Retrieval	0.583	0.552	0.485

From the learning curves, it can be seen that for both networks the validation and training curves converge but do not overfit. This suggests that the networks can still be improved. A deeper architecture for the shallow UNet as well as a better hyperparameter selection for L2-Net can definitely improve their performance.

3. Proposed Model

The proposed model consists of a modified TernausNet for the de-noising of the images and a fine-tuned L2-Net for the descriptor generation. Both models were improved individually and tested against the corresponding baseline models before connecting in series to test the final model.

3.1. Image denoising model

The main drawback of the shallow UNet is that it lacks in depth, consisting of only one encoding level and one decoding level. The shallower the network, the more it fails to extract important features of the image and to filter-out the added noise. Through a series of experiments, we focused on improving the network's architecture, making it deeper, to extract more information and perform better de-noising.

3.1.1 Model's architecture

The proposed model is a modified TernausNet [5]. It follows the same structure as a typical UNet as described in section 2.1. Instead of the original TernausNet which uses a pre-trained VGG11 network as the encoder, the proposed modified TernausNet uses a pre-trained VGG16 network for the encoding path.

VGG16 is chosen over VGG11 since as a deeper network and it can extract more features hence perform better de-noising. Its architecture is described in Figure 8.

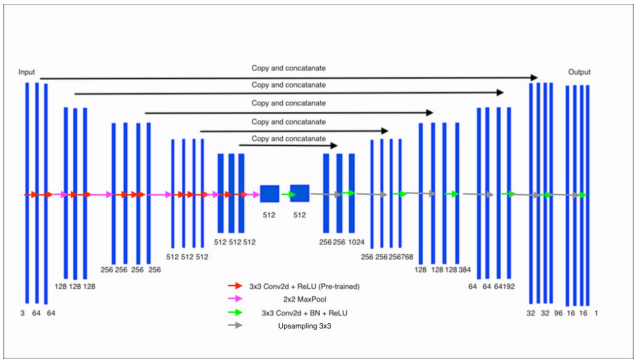
All convolutional layers of the modified TernausNet have 3x3 kernels and the number of channels is given in Figure 3.1.1. The first convolutional layer produces 64 channels and as the network deepens the number of channels doubles after each max pooling operation until it reaches 512. On the next layers, the number of channels remains the same.

The fully connected layers of VGG16 were removed and were replaced with a single convolutional layer of 512 channels that serves as a bottleneck. To construct the decoder

and avoid artifacts creating by transposed convolution, upsampling convolutions layers were used that double the size of a feature map while halving the number of channels. The output of a upsampling convolution is then concatenated with an output of the corresponding part of the decoder. The resultant feature map is treated by convolution operation to keep the number of channels the same as in a symmetric encoder term. This upsampling procedure is repeated 5 times to pair up with 5 max poolings.

Using transfer learning, the VGG16 encoding part of the network was initialized with the weights of VGG16 trained on ImageNet dataset. The encoding part was frozen during training. This allowed the model to increase its initial accuracy and reduce the training time since now only half of the network (the bottleneck and decoder) are trainable.

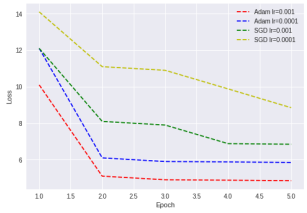
Since our input is a 32x32 monochrome noised image, we tripled and concatenated them to generate a 32x32x3 input. This is a necessary step since the ImageNet dataset is 3 channel dataset.



(a) Improved TernausNet

3.1.2 Model's optimiser and loss function

The original TernausNet uses Adam optimizer while the baseline shallow UNet SGD optimizer. Both optimizer with different learning rates were tested with the proposed TernausNet.



(a) Learning Validation Curves for learning rates and optimisers

Figure 5: Learning Validation Curves for learning rates and optimisers Adam (with lr=0.001 and lr=0.0001) and SGD (with lr=0.001 and lr=0.0001)

From the graph we can see that the best combination is Adam optimizer with learning rate of 0.001.The loss function for the model was not changed and is the mean absolute error $MAE = \frac{1}{N} \times \sum_1^N |y_i - x_i|$.

3.1.3 Model's evaluation against other denoisers

The proposed TernausNet outperformed not only the baseline shallow UNet but also a lot of other state of the art proposed denoisers when trained for same dataset. This is because it is a deeper network and its feature extracting path is trained in a powerful large dataset (ImageNet).

Metric	Patch Verification	Image Matching	Patch Retrieval
Shallow Net	0.523	0.114	0.230
DnCNN	0.582	0.143	0.280
UNet	0.593	0.154	0.316
VGG11-TNet	0.601	0.167	0.316
VGG16-TNet	**0.612**	**0.180**	**0.324**

Table 1: Denoising models were trained with 1500 patches for 30 epcohs. In the HPatch Benchmark, a baseline L2-Net descriptor, trained with 10000 triplets of clean images for 50 epochs, was used to generate the above results.

3.2. Image descriptor generation model

Although the baseline L2-Net has a state-of-the-art architecture for the generation of image descriptors, we proved that it can still be improved by modifying the training dataset, tune the hyper-parameters and hard-mining the generation of the triplets for the triplet loss function.

3.2.1 Training Dataset

For training the baseline L2-Net, denoised patches were used. By keeping the baseline L2-Net the same and train for the same number of epochs with clean patches, the L2-Net produces higher results when tested with noisy patches. This is justified since the higher the resolution of the image, the better feature extraction the L2-Net can perform. The final image descriptor generation model was trained using clean patches.

Trained with	Patch Verification	Image Matching	Patch Retrieval
Denoised	0.753	0.159	0.442
Clean	**0.799**	**0.180**	**0.472**

Table 2: Baseline L2-Net trained for 50 epochs with clean and denoised patches. As the denoiser the baseline shallow UNet was used.

3.2.2 Model's architecture and hyper-parameters

To tune the parameters, instead of using a Grid Search [2] where hyperparameters are pre-established with fixed steps increase, Tree-structured Parzen Estimator (TPE) algorithm [1] was used. TPE intelligently explores the search space

while narrowing down to the estimated best parameters. TPE is used to perform 10 tests each of duration of 20 epochs over the whole dataset, the algorithm provided significant improvements. More specifically, in the proposed model, the dropout value changed to 0.5, the 3rd and 4rd layer activation functions changed to sigmoid and the optimiser to adam with learning rate of 0.001.

The above values were not uncritically used. A series of measurements and tests took place to prove the results that the TPE outputted.

Metric	Relu	Sigmoid	L2-Net TPE params
Patch Verification	0.701	**0.703**	**0.740**
Image Matching	0.117	**0.117**	**0.115**
Patch Retrieval	0.387	**0.389**	**0.390**

Table 3: Baseline L2-Net and L2-Net with sigmoid activations in the intermediate layers trained for 50 epochs with clean patches and 10000 triplets. Tested using clean images in the HPatches Benchmark

The results are justified not only from the experimental results: Adam optimizer converges faster than the baseline sgd due to nesterov momentum and since the algorithm performs verification tasks, the choice of sigmoid fucntion for some intermediate layers is also reasonable.

3.2.3 Triplet loss improvement

While experimenting with different loss functions, it turned out that the triplet loss outperformed all of them and it remained unchanged in the proposed model.

Loss with	PV	IM	PR
Magnet Loss	0.210	0.020	0.003
TL-Manhattan Distance	0.520	0.057	0.145
Lossless Triplet Loss	0.601	0.096	0.250
Triplet Loss	**0.701**	**0.117**	**0.387**

Table 4: L2-Net trained with different loss functions for 50 epochs with clean patches and 10000 triplets. Tested using clean images in the HPatches Benchmark

To boost its performance, hard-mining of tough positive patches was performed. By using only tough positive patches for each anchor, the triplet loss tries always to minimise the distance with a very distant positive patch. Hard-mining methods can improve the performance of classification and verification tasks after initial training. Although this technique creates a bias, it resulted in a increase in all of the measurements. (Table 5)

3.2.4 Evaluation against baseline L2-Net

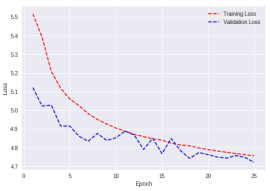
Combining all the above mentioned improvements, the proposed descriptor model was tested against the baseline L2-Net. (See Table 5) It is clear that from the above results that the change in the training dataset, the method the data for the loss function are extracted as well as fine-tuning with reasonable changes in the hyper-parameters, activations layers and optimisers resulted in an improved descriptor model.

Metric with	Verification	Matching	Retrieval
Baseline L2-Net	0.701	0.117	0.387
L2-Net Hard Mining	**0.823**	**0.359**	**0.392**
Proposed L2-Net	**0.832**	**0.365**	**0.412**

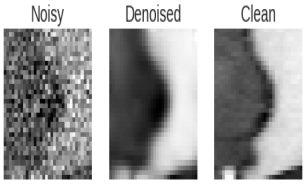
Table 5: Baseline L2-Net and PTE proposed L2-Net trained for 50 epochs with clean patches and 10000 triplets. Tested using clean images in the HPatches Benchmark

4. Final Training

The final complete model was trained using the whole dataset. More specifically, the improved TernausNet was trained for 25 epochs and the fine-tuned L2-Net for 80 epochs.

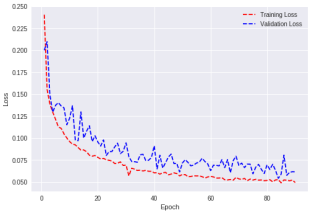


(a) Improved TernausNet



(b) Denoise Images Result

Figure 6: Figures (left to right): (a) Improved TernausNet learning Curve for training and (b) Denoised images results



(a) Proposed L2-Net Learning Curve

Figure 7: Proposed L2-Net learning Curve

Models	Validation	Matching	Retrieval
Baseline	0.827	0.254	0.552
Proposed Model	0.801	0.181	0.474

Although the final model performed worse than the baseline model, there are indications that the proposed model will eventually outperform it. The stage by stage analysis in Section 3, the fact that final model was trained in smaller number of the epochs of the baseline and the learning plots do not overfit suggest that with further training the proposed model will eventualluy outperform the baseline approach. Unfortunately, the large training time especially of the improved TernausNet did not allow us to see a clear improvement on time.

References

[1] J. Bergstra. Algorithms for hyper-parameter optimization https://papers.nips.cc/paper/4443.pdf.

[2] R. Joseph. Grid search https://towardsdatascience.com/grid-search.

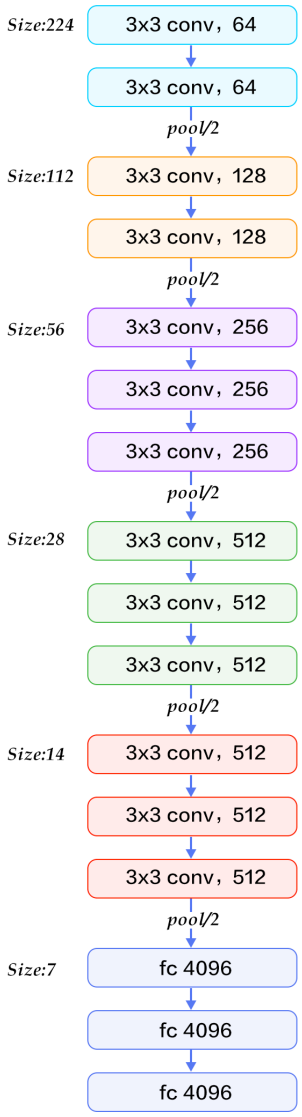
[3] A. V. K. M. Vassileios Balntas, Karel Lenc. Hpatches a benchmark and evaluation of handcrafted and learned local descriptors https://arxiv.org/pdf/1704.05939.pdf.

[4] A. V. K. M. Vassileios Balntas, Karel Lenc. Hpatches a benchmark and evaluation of handcrafted and learned local descriptors https://arxiv.org/pdf/1704.05939.pdf.

[5] A. S. Vladimir Iglovikov. Net with vgg11 encoder pre-trained on imagenet for image segmentation https://arxiv.org/pdf/1801.05746.pdf.

[6] . B. F. F. W. Yurun Tian1. Deep learning of discriminative patch descriptor in euclidean space. 2017.

5. Appendix



(a) VGG 16 Network Diagram

Figure 8: VGG16 Architecture: VGG16 contains twelve convolutional layers, each followed by a ReLU activation function and 5 max polling operations, each one reducing the feature map by 2.

Instructions for running the code

The notebook contains all the code and is self-contained. Although to train with the proposed hard-mining possible triplet loss, you need to use the $read\_data.py$ that is in the .zip file instead of the one that is downloaded from GitHub. You need to delete the old and place the new file in the same directory.