EE3-25 Deep Learning Coursework: Final Report

Miroslav Gasparek
mg6015, 01068411
miroslav.gasparek15@imperial.ac.uk

Abstract

This report describes denoising and description of Noisy HPatches, a form of Homography Patches dataset with added noise [3], using deep neural networks. The baseline framework employs shallow U-Net [15] to obtain the denoised images and the Hard-Net [14] architecture based on the L2-Net [18] to obtain the image descriptors. The improved design consists of deeper U-Net for the denoising, and deep network with introductory branching module and residual blocks to obtain the descriptors. The best performing architecture leads to average improvements of approx. 12%, 35% and 20% in the verification, matching, and retrieval tasks relative to the baseline.

1. Introduction

1.1. Problem Formulation

The objective of this machine learning problem is to develop the learning algorithm $\mathcal{A}$ that performs matching, verification, and retrieval tasks on the noisy version of HPatches dataset [3], using the combination of the denoising and descriptor network. In the *verification* task, the objective is to find the matching approach that describes the correspondence of any two patches[3]. In *matching*, descriptors match the patches from a reference to the target images. The goal of the algorithm is to determine the patch that best matches the reference[3]. In *patch retrieval* task, descriptors are used to find the patch correspondences in a large collection of patches with multiple distractors[3]. The detailed specification of the tasks can be found in the original paper by Balnats et. al. [3] The performance metrics of the algorithm is the *mean average precision* (mAP), the average of the maximum precisions at the different recall values [10]. Formally, the aim is to find the function composition $g = g_1 \circ g_2 : \mathcal{X} \rightarrow \mathcal{Y}$, where $g_1 : \mathbb{R}^{1\times32\times32} \rightarrow \mathbb{R}^{1\times32\times32}, g_2 : \mathbb{R}^{1\times32\times32} \rightarrow \mathbb{R}^{128\times1}$ are the respective functions approximations for the first and second neural net, $\mathcal{X}, \mathcal{Y}$ are input and output domains respectively [1].

1.2. Dataset Description

The dataset $\mathcal{D}_j = \{(\mathbf{x}_1, \mathbf{x}_1^{ref}), ..., (\mathbf{x}_N, \mathbf{x}_N^{ref})\}$ contains $N = 116$ sequences of images, where each sequence contains a reference image $\mathbf{x}_i^{ref} \in \mathbb{R}^{1\times32\times32}$ and 5 target images $\mathbf{x}_i \in \mathbb{R}^{5\times32\times32}$ with the noise caused by the geometric transformation (affine jitter) [3]. The dataset contains subsets denoted as *clean* (no noise added), *easy*, *hard*, and *tough*, depending on the amount of noise and distortion added [2]. The purpose of the dataset is to serve as the benchmarks for evaluation of the local descriptors, which, unlike the other datasets, has not been almost saturated [3]. In the baseline setting, the input to the (overall) network are the images of size $32 \times 32 \times 1$, and the output is a vector of size 128×1. The number of samples used for the baseline denoise training and testing is $45,952$ and $13,648$, while the number of training and testing samples for the baseline descriptor network is $100\,000$ and $10\,000$ images respectively, using the default split `'a'` as in the original HPatches dataset [4].

2. Baseline Approach Analysis

As mentioned above, the baseline network consists of two parts: a denoising part based on the shallow U-Net, which combines encoder, bottleneck, and decoder parts and has been successfully for denoising of the biomedical images [11] [15]. The network contains 59 777 trainable and no non-trainable parameters and aims to minimize the mean absolute error (MAE) [6]. The descriptor part of the network copies the HardNet [14], which is based on the L2-Net that learns the discriminative patch descriptors [18]. This network contains $1\,336\,032$ trainable and 896 non-trainable parameters. This part of the network aims to minimize the triplet loss [9] in Euclidean space, which has the form of

$$\mathcal{L} = max\left(\|\mathbf{a} - \mathbf{p}\|_2^2 - \|\mathbf{a} - \mathbf{n}\|_2^2 + b,\ 0\right) \qquad (1)$$

where $\mathbf{a}$ is an anchor patch, $\mathbf{p}$ is a positive patch, and $\mathbf{n}$ is a negative patch, while $b = 1.0$ is a margin. The aim is to minimize the anchor-positive Euclidean distance, and to maximize the anchor-negative Euclidean distance.

2.1. Training

Both denoising and descriptor parts of the network are trained for one epoch, which takes 28 and 172 seconds respectively, using the Google Colab engine. Both parts of the network use the Stochstic Gradient Descent (SGD) as an optimizer. The learning rate of the denoising network is set to 0.00001 and this network also uses the Nesterov momentum, while the size of a batch is 50 samples. On the other hand, the descriptor network's learning rate is 0.1 without Nesterov momentum. Initially, both networks attempt to minimize the MAE. The training loss of both denoising and descriptor networks over a single epoch are shown in the Figure 2 and Figure 3 in the Appendix.

2.2. Evaluation

Denoising network. The performance of the denoising part of the network can be qualitatively seen in the Figure 1. While some of the noise might has been removed, the true shape of the image is not well identifiable when compared to the clean patch on the right. The area of the image with lower intensity variation (low frequency) at the bottom right resembles the original image more than the area around the edges, with the high intensity variation (high frequency). Based on this observation, high frequency noise seems to play important role.

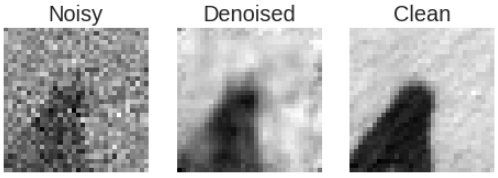

Figure 1. The denoising of the input image with the baseline shallow U-Net. The noisy input image (left), denoised image (center) and the clean reference image (right).

Descriptor network. The mAP scores (which we aim to improve) on the verification, matching, and retrieval tasks are approx. **0.71**, **0.12**, and **0.37** respectively. This shows that there is a significant room for improvement, especially in the matching tasks. More detailed breakdown of the scores can be found in the attached `.ipynb` files.

2.3. Analysis

The baseline model offers a significant room for improvement. The architecture of the baseline denoising network is shallow, with a single encoder convolutional layer and three decoder convolutional layers, which might not allow for the extraction of the low-level features [6]. Furthermore, denoising network does not use any regularization (such as Batch Regularization, L1 or L2 loss regularization). The MAE loss used by the denoising network is more robust with respect to the outliers, but it can be replaced by the mean squared error (MSE), which can offer the better convergence properties [5].
The descriptor network does contain batch normalization, which can help avoid the vanishing gradient problem [13]. However, its performance can be improved by the inclusion of the residual blocks and the deeper architecture inspired by the ResNet [7], effectively increasing the carrying capacity of the network [6]. Furthermore, both networks use SGD optimizer, which requires a well-tuned learning rate for its optimal performance. On the other hand, the more sophisticated optimizers such as Adam [12] can be more robust with respect to the learning rate changes in the case of hyperparameter variation. The improved approach should also involve the training for increased number of epochs, so that the algorithm can avoid underfitting. On the other hand, the stopping conditions for the training, based on the validation loss, should be employed to prevent overfitting.

3. Improved Neural Network

The main requirements for the improved network design include the removal of the high-frequency noise through the construction of the deeper denoising network, building the more complex architecture for the descriptor network to provide the more optimal set of the descriptors, and the improvement of the training process through use of the more appropriate set of the hyperparameters and extension of the training length.
The final architecture consists of the **deeper version of the U-Net** [15] for the denoising and the **combination of the *Inception*-inspired branched introductory stage** [16] **and the residual blocks described in** [7] for the descriptor network.

3.1. Denoising Network Architecture

Similar to the baseline approach, the denoising architecture is based on the U-Net architecture [15]. It contains the contracting path, the bottleneck, and the expanding path. The contracting path contains the convolutional layers with the increasing sizes of the filters (64, 128, 256, and 512) followed by the nonlinear ReLU activations and MaxPooling layers. The pooling makes the representation approximately invariant to the small changes translations in the input [6]. This allows the representation to learn whether the feature is present rather than where exactly it is and also improves the computational efficiency of the network [6]. Subsequently, the bottleneck convolutional layer with the largest size of the filter followed by the dropout layer is included. Finally, the convolutional layers in the expanding path are followed by the upsampling operators to increase the resolution of the output. The layers of the expanding path with the upsampling layers are then combined with the high resolution features of the contracting path to produce

the more precise input [15]. In the end, we obtain the characteristic symmetrical U-shaped architecture, as the filter numbers in the contracting and expanding paths are equal. This approach combines the invariance to the small translation, which might arise due to the added geometric noise as described in [3], and the local context learning to remove the high frequency noise, hence meeting the requirements for this part of the network.
While the original UNet architecture [15] employs a pair of the convolutional layers at each stage of the contraction and expansion paths, during the training, it was found that such architecture exhibits high computational demands and its performance is suboptimal compared to the actual architecture used and displayed in the Figure 10 in the Appendix. Overall, the denoising network contains 15 324 097 trainable parameters, which is an considerable increase in complexity compared to the baseline approach with its $59,777$ parameters.

3.2. Descriptor Network Architecture

The descriptor network was inspired by the Inception network described by Szegedy et al. in [17], which has introduced the Inception module, and by the residual representation as described in [7], where He et al. have shown that the blocks that learn the residual functions are easier to optimize and can enjoy the benefits of the deeper architectures [7]. The use of the identity shortcuts in [7], allows for the more straightforward learning of the possible identity representations, effectively by-passing the need to approximate the linear functions by the strong non-linearities. The overall idea of the network is to expose the training samples to the wide range of the features and let the network to select the useful ones in the residual stage.

The overall architecture of the descriptor network is displayed in the Figure 11. The descriptor network starts with the *introductory stage* containing two convolutional layers with 16 filters each, both followed by the nonlinear ReLU activation and the Batch Normalization (BN) units. Subsequently, the output of these layers is fed into a non-symmetrical branched network. The first branch includes two identical blocks of convolutional layer with 32 filters, ReLU activation, and batch normalization. The other branch contains three convolutional layers with 8, 8, and 32 filters respectively. Subsequently, the outputs of these branches are added together. The purpose of this stage is to enable the network to learn the large number of the features.
The output of the introductory stage is fed into the *residual stage*, which consists of the convolutional residual block and two identity residual blocks, schematics of which is displayed in the Figure 2. Finally, the network is terminated with the Dropout layer, two Convolution+ReLU+BN layers with larger kernel sizes, followed by the final reshape to provide the output of the desired size 128×1. Overall, the descriptor part of the network contains $1,324,656$ trainable and $1,088$ non-trainable parameters, compared to the $1,336,032$ trainable and 896 non-trainable parameters of the original network. During the experiments, the architecture of the residual stage was kept intact, but the initial stage was amended both in terms of the number of the introductory convolutional layers and the number of the filters in these layers was altered, and it was found to have the significant impact on the performance.

3.3. Training

The training of the networks was done separately, i. e. first, the denoising part of the network was trained, the image samples were denoised using the trained model, and subsequently were fed into the descriptor model for its training and the evaluation in verification, matching, and retreival tasks.
The most influential hyperparameters were the number of the training samples for the descriptor network and the depths of the denoising and descriptor stages. The number of training samples for descriptor network varied from $5,000$ and 500 training and testing samples respectively, to $50,000$ and $5,000$ training/testing split. The descriptor model trained at the lower number of samples performed significantly worse than the model trained at the higher number of samples.
The second very influential hyperparameter was the depth of the denoising network and the depth of the introductory stage. If only 8 filters were used in the introductory stage of the descriptor network for both convolutional layers, the network underperformed, as well as with 32 filters used. Similarly, increasing number of the layers in contracting and expanding paths of the denoising network did not lead to the improved performance when compared to the baseline over a single epoch.
The number of the training and testing samples for the denoising network was equal to the number of the samples for the baseline approach (three and one sequences for training and validation respectively). The best performing descriptor network was trained on the largest number of samples as mentioned above. The reducing of the number of training/testing samples compared to the baseline was done to reduce the computational burden, as in this setting, the single epoch of descriptor network training takes approximately 120 seconds, which is less than the training time across the single epoch of the baseline model, which is approximately 170 seconds. The overall training process of denoising and descriptor networks in the Google Colaboratory environment with the GPU took approximately 150 minutes, including the extraction and preparation of the samples.
Both denoising and descriptor networks were trained using

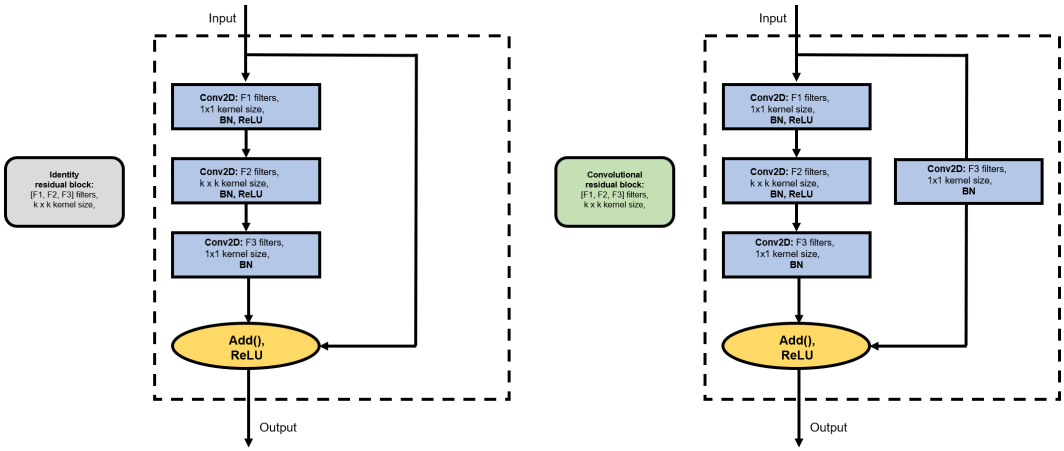

Figure 2. The schematics of the structure of the residual blocks. The full architectures of the networks are in the Appendix.

the Adam optimizer [12] with the initial learning rate 0.001 for 20 epochs. If the validation loss reached the plateau, in the subsequent epoch it was multiplied by the factor of 0.2. The minimum learning rate was set to 0.0001. The layers in both networks were initialized with He initialization [8]. Also, denoising network was trained with MSE loss, instead of MAE loss, while the descriptor network employed the triplet loss and MAE loss, as in the baseline.

3.4. Performance Evaluation

The best performing architecture achieved the improvements of approx. **12%**, **35%** and **20%** in the verification, matching, and retrieval tasks relative to the baseline, with the specific mAP of **0.80**, **0.16**, and **0.45** in the respective tasks. More detailed breakdown of the scores can be found in the attached `.ipynb` files. This can be considered to be a significant improvement over the baseline. The overall training time across 20 epochs for the denoising network was approximately 80 min (approx. $20\times$ increase in computation time), while the training of the descriptor network took approximately 40 min (approx. $14\times$ increase). Overall, the computational costs increased, but were still within a reasonable time intervals for a one-off task, when the online learning is not required. The performance of the improved denoising approach is shown in the Figure 3, showing the qualitative improvement in the performance of the denoising network.

3.5. Discussion

Overall, this work designed a pair of the neural networks that demonstrated a significant improvement above the baseline performance in the verification, matching, and retrieval tasks using the HPatches dataset.
Further improvements are indeed possible - in terms of architecture, the denoising network could include the Batch

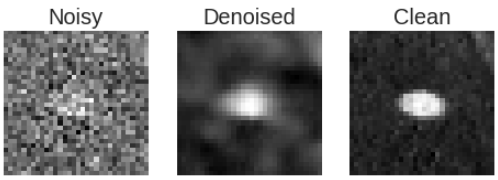

Figure 3. The denoising of the input image with the improved deeper U-Net. The noisy input image (left), denoised image (center) and the clean reference image (right).

Normalization, while the descriptor network could have more residual blocks included, although this would probably significantly increase the computational burden. Also, longer training time could be use to see if the validation loss would decrease further and to avoid the convergence in the local minima due to the non-convexity of the problem [5]. The number of the samples for both denoising and descriptor network could be increased, as it has been observed that such increase in the number of samples improved the performance of the model. Furthermore, the fine-tuning of the optimizer could be employed, so that the appropriate convergence is reached, perhaps over a greater number of the epochs. Generally, a more rigorous hyperparameter grid search would be beneficial, using the specialized hyperparameter optimization modules, such as `Talos`.
Furthermore, it will be interesting to use the model on a different dataset, for instance on a medical imaging dataset, where the number of the potential training samples is modest [11]. It would be interesting how the different nature of the noise present would influence the performance of the combined model, and how this model can be used in conjunction with the data augmentation to deliver the overall practical result.

4. Appendix

4.1. Pledge

I, Miroslav Gasparek, pledge that this assignment is completely my own work, and that I did not take, borrow or steal work from any other person without referencing, and that I did not allow any other person to use, have, borrow or steal portions of my work. I understand that if I violate this honesty pledge, I am subject to disciplinary action pursuant to the appropriate sections of Imperial College London.

I have discussed my work with Martin Ferianc.

4.2. Code

The final code that generated the discussed results is provided in the attached `.zip` package in the form of two `.ipynb` files, one for the Baseline approach, and one for the Improved approach.

Figure 4. The loss for the baseline approach denoising network over a single epoch.

4.3. Baseline Network Architectures

The baseline network architecture in a schematic format is on the next page.

4.4. Improved Network Architectures

The improved network architecture in a schematic format is on the following page.

Figure 5. The loss for the baseline approach descriptor network over a single epoch.

Figure 6. Training and validation loss for the improved denoising network over the course of 20 epochs.

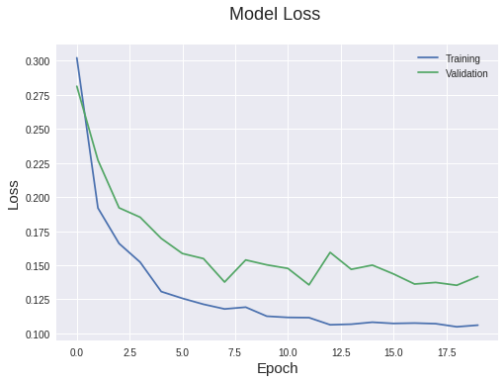

Figure 7. Training and validation loss for the improved descriptor network over the course of 20 epochs.

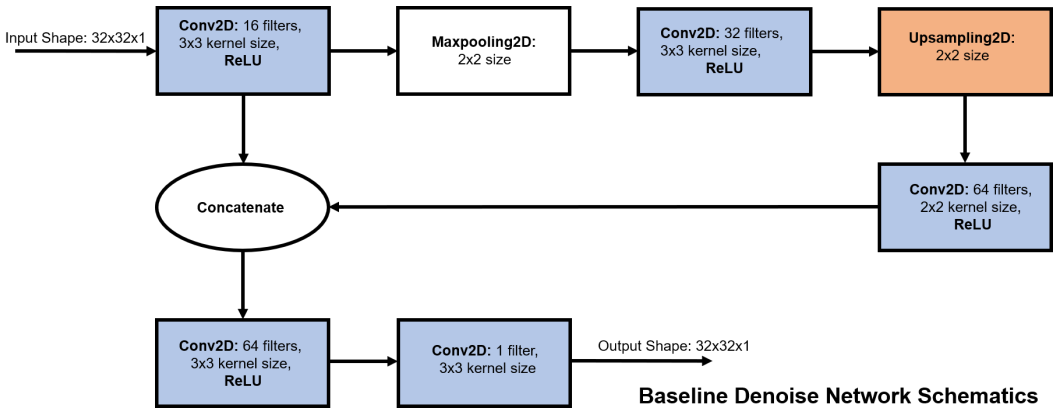

Figure 8. The schematics of the structure of the Baseline denoising network. The block containing *Conv2D*, *ReLU* and *BN* in given order means the convolutional layer followed by the Rectified Linear Unit Activation, followed by the Batch Normalization.

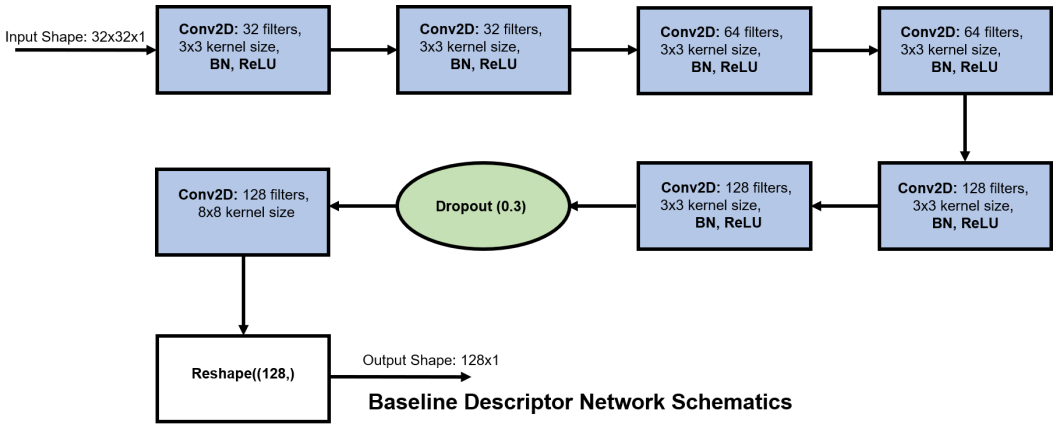

Figure 9. The schematics of the structure of the Baseline descriptor network. The block containing *Conv2D*, *ReLU* and *BN* in given order means the convolutional layer followed by the Rectified Linear Unit Activation, followed by the Batch Normalization.

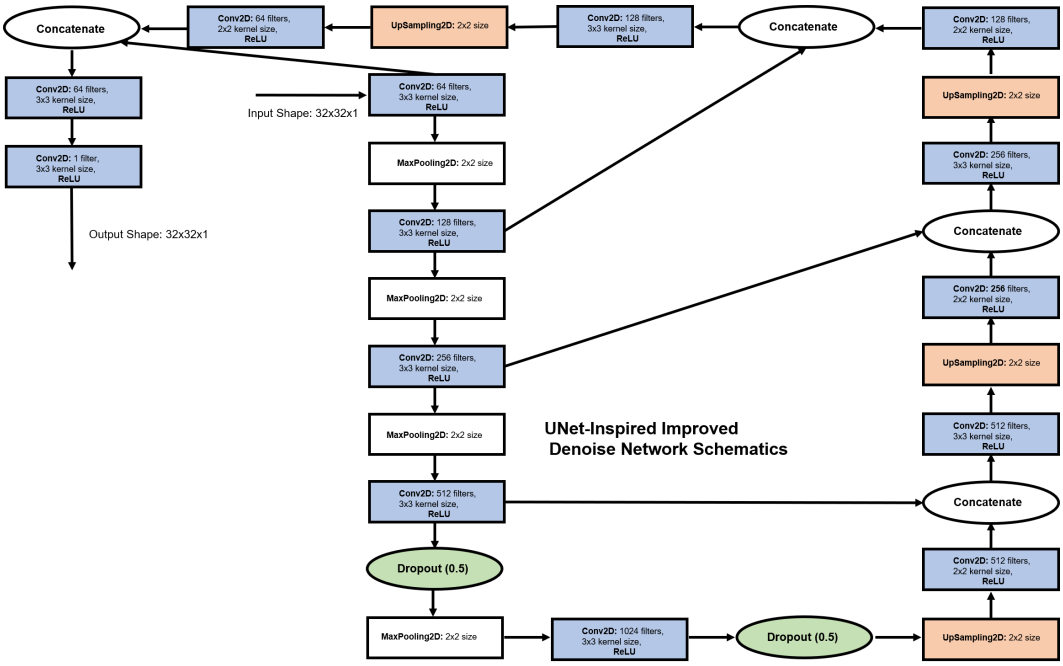

Figure 10. The schematics of the structure of the Improved denoising network.The block containing *Conv2D*, *ReLU* and *BN* in given order means the convolutional layer followed by the Rectified Linear Unit Activation, followed by the Batch Normalization.

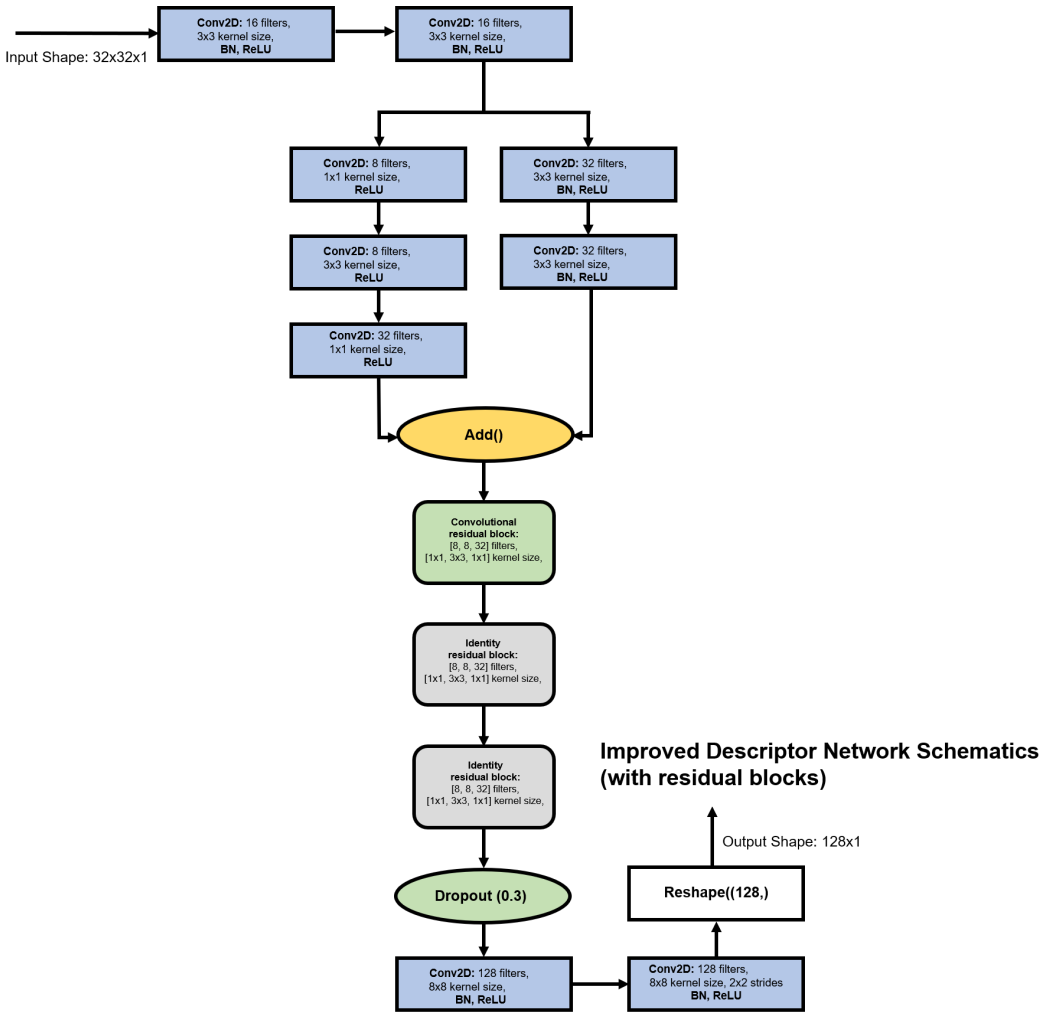

Figure 11. The schematics of the structure of the Improved descriptor network. For the structure of the residual blocks, please refer to the Figure 2.The block containing *Conv2D*, *ReLU* and *BN* in given order means the convolutional layer followed by the Rectified Linear Unit Activation, followed by the Batch Normalization.

References

[1] Y. S. Abu-Mostafa, M. Magdon-Ismail, and H. T. Lin. *Learning from Data: A Short Course*. 2012.

[2] V. Balnats. HPatches: Homography-patches dataset, 2017.

[3] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk. HPatches: A benchmark and evaluation of handcrafted and learned local descriptors. Technical report.

[4] A. Barroso and A. Lopez. keras_triplet_descriptor, 2018.

[5] S. Boyd and L. Vandenberghe. *Convex Optimization*. 2004.

[6] I. Goodfellow, Y. Bengio, and A. Courville. Deep Learning Book. *Nature*, 2016.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. Technical report.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. Technical report.

[9] A. Hermans, L. Beyer, and B. Leibe. In Defense of the Triplet Loss for Person Re-Identification. Technical report.

[10] J. Hui. mAP (mean Average Precision) for Object Detection Jonathan Hui Medium, 2018.

[11] P. F. Jaeger, S. A. A. Kohl, S. Bickelhaupt, F. Isensee, T. A. Kuder, H.-P. Schlemmer, and K. H. Maier-Hein. Retina U-Net: Embarrassingly Simple Exploitation of Segmentation Supervision for Medical Object Detection. Technical report.

[12] D. P. Kingma and J. Lei Ba. ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION. Technical report.

[13] J. F. Kolen and S. C. Kremer. Gradient Flow in Recurrent Nets: The Difficulty of Learning LongTerm Dependencies. Technical report, 2010.

[14] A. Mishchuk, D. Mishkin, F. R. Radenovic, and J. Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. Technical report.

[15] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. Technical report.

[16] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. Technical report.

[17] C. Szegedy, W. Liu, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. Technical report.

[18] Y. Tian, B. Fan, and F. Wu. L2-Net: Deep Learning of Discriminative Patch Descriptor in Euclidean Space. Technical report.