IEIE Transactions on Smart Processing and Computing, vol. 6, no. 3, June 2017
https://doi.org/10.5573/IEIESPC.2017.6.3.193

Evaluation of Recurrent Neural Network Variants for Person Re-identification

Cuong Vo Le[1], Nghia Nguyen Tuan[1], Quan Nguyen Hong[1], and Hyuk-Jae Lee[2]

[1] School of Electronics and Telecommunications, Hanoi University of Science and Technology / Hanoi, Vietnam
[2] Inter-university Semiconductor Research Center, Department of Electrical and Computer Engineering, Seoul National University / Seoul, South Korea hyuk_jae_lee@capp.snu.ac.kr

* Corresponding Author: Cuong Vo Le, cuong.vole@hust.edu.vn

Received May 3, 2017; Accepted May 26, 2017; Published June 30, 2017

* *Regular Paper*

***Abstract*:** Instead of using only spatial features from a single frame for person re-identification, a combination of spatial and temporal factors boosts the performance of the system. A recurrent neural network (RNN) shows its effectiveness in generating highly discriminative sequence-level human representations. In this work, we implement RNN, three Long Short Term Memory (LSTM) network variants, and Gated Recurrent Unit (GRU) on Caffe deep learning framework, and we then conduct experiments to compare performance in terms of size and accuracy for person re-identification. We propose using GRU for the optimized choice as the experimental results show that the GRU achieves the highest accuracy despite having fewer parameters than the others.

***Keywords*:** Person re-identification, Multi-shot, Recurrent neural network, Optimization

1. Introduction

Person re-identification (the process of recognizing an individual in images from a camera network) is a fundamental task in automated surveillance. It has been receiving attentions for years [1]. This task is challenging due to problems such as appearance variations of the individual across different cameras, and low quality in video and image resolution. There are a number of proposals addressing the re-identification problem. The core idea behind them is to build a discriminative set of features for each person. These works can be divided into two methods: single-shot and multi-shot.

The single-shot methods aim to learn the features of each person from a single image. Many types of feature have been explored, from basic global ones like color and texture [2, 3] to other more descriptive ones, like local binary patterns (LBP) [4]. One branch of single-shot methods investigates salient-characteristics learning [5-7] to distinguish one person from the other in the dataset. Recently, deep learning features have been trending due to their effectiveness [8].

However, in practical surveillance, persons usually appear in a video, rather than a still image. In addition, heavy occlusion and viewpoint variations cause appearances of the person in multiple cameras to be either unclear or inconsistent, thus, making single-shot methods ineffective. As a result, recent approaches focus on video-based person re-identification, in which features are learned from an image sequence. Temporal information combined with spatial information makes the feature set more discriminative. A few methods have been proposed to solve this task, most of which use recurrent neural networks to extract temporal information [9-11].

There are a lot of different RNN architectures, and choosing the most optimized one is essential for such a computationally expensive task. Our work aims to solve this optimization problem for multi-shot deep learning-based person re-identification, and can be considered an extension of the work by Greff et al. [12] in which the performance comparison is based on various problems other than person re-identification. We use recurrent feature aggregation network (RFA-Net) [9] in combination with simple input and a basic metric learning method to highlight the performance of a core recurrent layer. We conduct experiments with five popular RNN variants to find out which one is the most suitable for a re-identification task that balances network size and matching accuracy.

2. Related Work

Video-based person re-identification has been a research field of interest recently. Since the task is similar to classification, the related work mainly focuses on building a feature set to represent a person, and on distance metric learning for accurate matching. Some work [9, 13] inherits single-shot methods that re-uses a few popular feature types, such as LBP, colors and histogram of oriented gradients (HOG) [14]. However, these features are not descriptive enough for sequence matching using either cosine distance or rank support vector machine (RankSVM) [15]. Some others proposed new ways to describe a person by combining spatial and temporal information. Most of those proposals used a recurrent neural network, a powerful tool to learn sequential data. Yan et al. [9] constructed a model with a long short term memory network to build a sequence-level representation from simple features. McLaughlin et al. [10] successfully used a convolutional neural network in combination with a recurrent one to boost the accuracy of video matching. Although there is no restriction on choosing a recurrent architecture, it is reasonable to employ the most optimized one for the highest accuracy. In addition, since video classification is computationally more expensive than an image-based one, it is important to consider the trade-off between size and accuracy.

3. Network Implementation

In this section, we review five popular recurrent architectures that will be implemented and used in our experiments.

3.1 Architecture Overview

3.1.1 Recurrent Neural Networks

Although traditional neural networks work well on a single-image description, they do not make use of temporal information. When it comes to describing a sequence, they process each image disjointedly. Therefore, it is clearly incompatible to use traditional neural networks alone for video-based classification tasks in which the order of images in the sequence matters. Recurrent neural networks were proposed to solve that problem.

RNNs are networks with loops in them. It turns out that an unrolled RNN is similar to a network containing a series of identical layers, in which one layer connects and transfers its output to another layer sequentially. The role of the loops is to connect the information of previous nodes with the current one and to make use of parameter sharing. In other words, a single output of the network concurrently depends on the current and a number of previous inputs. The feed-forward function of RNN at a time step t can be described as:

$$h_t = \tanh\left(W_{hx}x_t + W_{hh}h_{t-1} + b_h\right) \quad (1)$$

$$o_t = \tanh\left(W_{oh}h_t + b_o\right) \quad (2)$$

where x_t and o_t are input and output of the network at current time step t while h_{t-1} and h_t are hidden states at the previous and current time step respectively. As shown in Eqs. (1) and (2), the RNN makes a decision based on current input and previous hidden states. At first, it looks at the input and previous hidden state to calculate a new state. Then, the output is generated based on that new state. Logically, changes in the input order lead to different behaviors in the output. During training, three weights (W_{hx}, W_{hh}, and W_{oh}) and two biases (b_h and b_o) are tuned. Since parameter sharing is applied, these weights and biases will not alter as time step t varies.

3.1.2 Long Short-term Memory

Although an RNN is capable of remembering information over time, its memory is limited. As the input sequence becomes longer, information from previous nodes starts to vanish, more or less depending on how far they are from the current node. In practical terms, a short sequence is unlikely discriminative enough for video-based tasks, thus making the long-term dependency problem a challenging one. In order to solve this problem, Hochreiter and Schmidhuber introduced the long short-term memory (LSTM) network [16], a special kind of RNN.

An LSTM includes a more complicated gate structure inside one chunk of the network, and a new output cell state, which works like memory. Those gates filter sequential input flows into the network, granting LSTM the ability to store information in, and remove information from, the cell state. There are three gates inside one repetitive chunk: forget gate f, input gate i and output gate o. Their feed-forward functions are the following:

$$f_t = \sigma\left(W_{fx}x_t + W_{fh}h_{t-1} + b_f\right) \quad (3)$$

$$i_t = \sigma\left(W_{ix}x_t + W_{ih}h_{t-1} + b_i\right) \quad (4)$$

$$o_t = \sigma\left(W_{ox}x_t + W_{oh}h_{t-1} + b_o\right) \quad (5)$$

$$g_t = \tanh\left(W_{gx}x_t + W_{gh}h_{t-1} + b_g\right) \quad (6)$$

where σ is the sigmoid function. The forget gate allows relevant parts of information stored in the previous cell state to pass through while forgetting the rest. After flushing some information, the cell state looks at candidate vector g and uses input gate i to decide which will be updated. The updating function of cell state is:

$$c_t = f_t \cdot c_{t-1} + i_t \cdot g_t \quad (7)$$

in which "·" denotes an element-wise multiplication operator. Finally, hidden output h is decided based on current cell state with the help of filter output gate o:

$$h_t = o_t \cdot \tanh\left(c_t\right) \quad (8)$$

In comparison with an RNN, an LSTM network learns the temporal features more selectively. The complicated structure of an LSTM usually results in better performance while the trade-off is a higher number of parameters.

3.1.3 Long Short-term Memory with Coupled Gate

A long short-term memory network with a coupled gate (LSTMC) is a slight variant of the original version. Instead of having gates f and i to forget and update information separately, this version combines these two into one, reducing the number of parameters. The idea is that we immediately replace the flushed part in the cell state with a new one. The architecture can be described by the following equations:

$$f_t = \sigma\left(W_{fx}x_t + W_{fh}h_{t-1} + b_f\right) \quad (9)$$

$$g_t = \tanh\left(W_{gx}x_t + W_{gh}h_{t-1} + b_g\right) \quad (10)$$

$$c_t = f_t \cdot c_{t-1} + \left(1 - f_t\right) \cdot g_t \quad (11)$$

$$o_t = \sigma\left(W_{ox}x_t + W_{oh}h_{t-1} + b_o\right) \quad (12)$$

$$h_t = o_t \cdot \tanh\left(c_t\right) \quad (13)$$

3.1.4 Long Short-term Memory with Peephole Connection

A long short-term memory network with peephole connection (LSTMP), which was introduced by Gers et al. [17], is another slight variant of the original version. However, while one with a coupled gate tends to simplify the original architecture, this version does the opposite by adding more connections between gates and inputs, flexibly allowing gates to have more information before making a decision. The following equations show how things flow inside one chunk:

$$f_t = \sigma\left(W_{fx}x_t + W_{fh}h_{t-1} + W_{fcp}c_{t-1} + b_f\right) \quad (14)$$

$$i_t = \sigma\left(W_{ix}x_t + W_{ih}h_{t-1} + W_{icp}c_{t-1} + b_i\right) \quad (15)$$

$$g_t = \tanh\left(W_{gx}x_t + W_{gh}h_{t-1} + b_g\right) \quad (16)$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot g_t \quad (17)$$

$$o_t = \sigma\left(W_{ox}x_t + W_{oh}h_{t-1} + W_{oc}c_t + b_o\right) \quad (18)$$

$$h_t = o_t \cdot \tanh\left(c_t\right) \quad (19)$$

3.1.5 Gated Recurrent Unit

The GRU architecture [18] changes significantly in comparison with the original LSTM. Not only forget and input gates, but also cell state and hidden output, are combined into one. That results in a simple model, yet still maintains its effectiveness in capturing temporal information. The four equations below describe the GRU architecture:

$$z_t = \sigma\left(W_{zx}x_t + W_{zh}h_{t-1} + b_z\right) \quad (20)$$

$$r_t = \sigma\left(W_{rx}x_t + W_{rh}h_{t-1} + b_r\right) \quad (21)$$

$$g_t = \tanh\left[W_{gx}x_t + W_{ghr}\left(h_{t-1} \cdot r_t\right) + b_g\right] \quad (22)$$

$$h_t = \left(1 - z_t\right) \cdot h_{t-1} + z_t \cdot g_t \quad (23)$$

Table 1. Number of parameters in the RNN variants in general.

Variant	Number of parameters
RNN	$1 \times D_h \times D_x + 2 \times D_h \times D_h + 2 \times D_h \times 1$
GRU	$3 \times D_h \times D_x + 2 \times D_h \times D_h + 3 \times D_h \times 1$
LSTMC	$3 \times D_h \times D_x + 3 \times D_h \times D_h + 3 \times D_h \times 1$
LSTM	$4 \times D_h \times D_x + 4 \times D_h \times D_h + 4 \times D_h \times 1$
LSTMP	$4 \times D_h \times D_x + 7 \times D_h \times D_h + 4 \times D_h \times 1$

3.2 Complexity Comparison

In this section, five RNN variants are compared with each other to determine their complexity. We calculate the number of parameters that each layer has to learn during the training phase. Commonly, for each time step t, all five variants take data x_t and the hidden output of previous step h_{t-1} as inputs. The original LSTM, LSTMC and LSTMP have additional input c_{t-1}. In our implementation, x_t has a dimension of $1 \times N \times D_x$, and h_{t-1} has a dimension of $1 \times N \times D_h$ where N is batch size, D_x is the length of the data input vector, and D_h is the length of the hidden output; c_{t-1} has the same dimensions as h_{t-1}. Since batch size N does not affect the number of parameters, we assume $N = 1$. From Eqs. (1)-(23), we list in Table 1 the number of parameters for each architecture. As can be seen from the table, all variants have their sizes scaled as the length of inputs D_x and D_h become larger. However, the order of sizes is fixed such that the RNN will always have the smallest size and the LSTMP is the largest for any input dimensions. When D_x is more than ten times larger than D_h (to ensure that the network learns the essential features and filters out the others), the size of the recurrent layer is more dependent on D_x. In these cases, a GRU and an LSTMC will have almost the same size, and this applies to an LSTM and an LSTMP as well. They are three and four times larger than the smallest one (an RNN).

Size and processing speed are important to the applications of neural networks. However, another factor that must not be ignored is performance. In the next section, we conduct some experiments to quantitatively measure how well they work on person re-identification tasks.

4. Experiments

4.1 Datasets

Two datasets (iLIDS-VID [19] and PRID 2011 [20]), which are common for video-based person re-identification with public benchmarks available, are used to evaluate the performance of different architectures.

iLIDS-VID dataset. This dataset consists of 600 image sequences of 300 persons. Each person has one pair of image sequences, and each image sequence has variable lengths of 23 to 192 frames, with an average of 73 frames. This dataset was created based on two non-overlapping cameras from a CCTV network in an airport arrival hall. The variations of lighting and viewpoint, cluttered

background, and occlusions make this dataset a challenging one.

PRID 2011 dataset. This dataset includes 400 image sequences of 200 persons. Each person has one pair of image sequences, each of which consists of 5 to 675 frames, with an average of 100 frames. The images were captured from two cameras in a public outdoor environment with a clean background and rare occlusions, which make it simpler than the iLIDS-VID dataset.

In both datasets, each person has two image sequences. Like Yang et al. [19], we filter out from each dataset the persons whose sequences have fewer than 21 frames. As a result, 600 image sequences of 300 persons in iLIDS-VID and 356 image sequences of 178 persons in PRID 2011 were used for evaluation. Each dataset was randomly split into two subsets, one for training the RFA-Net, and other for testing. During the testing phase, image sequences from the first and second camera were regarded as probe and gallery, respectively.

4.2 Reuse of Recurrent Feature Aggregation Network

The recurrent feature aggregation network [9] was proposed by Yan et al. in 2016. Its architecture is simple, with an LSTM as the core layer to learn sequential information. Yan et al. [9] proposed using a simple LBP and color (LBP&Color) features as input and a basic metric learning method with cosine distance to evaluate the performance. Inspired by their idea, we conducted extensive experiments to further compare the performance of different RNN variations. To highlight the difference of using various architectures, simple features and classifier were also used in our experiments.

LBP&Color. Like Hirzer et al. [21], we created LBP&Color features for the two datasets. Each image was resized to 128 × 64, then divided into multiple patches of 16 × 8, each one overlapping its neighbor with an 8 × 4 region. Histograms of LBP codes from gray-scale representation and mean values of hue-saturation-value (HSV) and Lab color channels were computed for each patch, then concatenated into a single frame-level feature vector, for which the dimension is 58950.

Cosine Distance Metric Learning. A cosine distance metric learning method is used to find a probe-gallery pair. Let d be the distance; then, it is computed as follows:

$$d_{ij} = \frac{s_i^a \cdot s_j^b}{s_i^a s_j^b} \tag{24}$$

where s_i^a and s_j^b are final sequence level representation of person i from camera a and person j from camera b respectively, and v denotes the L_2 norm of vector v. For one probe, we compute the distances to all instances in the gallery and list them in ascending order. If the true match s_i^a and s_i^b is expected in the top k, distance d_{ii} has to appear within the ascending list of k distances.

Other Settings. An RFA-Net includes a core recurrent layer: the original LSTM. The output of each time step is a 512-element feature vector. During training, the output of the LSTM is fed through a fully connected layer and then a Softmax layer to calculate loss. Networks are trained as a classification task with the number of classes, N, equal to the number of persons in the training set, i.e. N = 89 and N = 150 for the PRID 2011 and iLIDS-VID datasets, respectively. As suggested by Yan et al. [9], we chose to fuse feature vectors of 10 consecutive time steps into a 5120-element sequence-level representation. To avoid sequence-length variation of different persons, we also choose 10 random subsequences to create an average final sequence-level representation.

4.3 Evaluation Protocols

To obtain stable statistical results, we conducted 10 trials for each architecture with different training/testing splits of each dataset and report the averaged results. A cumulative match characteristic (CMC) curve was adopted to show the accuracy of each model with ranking.

To fairly compare the performance of different architectures, we tried different sets of hyper-parameters to find suitable ones for training. In detail, we trained models for at least 30,000 iterations, with a batch size of 8 on the PRID 2011 dataset. On the iLIDS-VID dataset, we trained for 20,000 iterations with a batch size of 20. The learning rate was initially set to 0.001 and dropped to 0.0001 after 20,000 and 10,000 iterations using the PRID 2011 and iLIDS-VID dataset respectively. Early-stopping strategy was applied to avoid over-fitting. The final models were carefully selected from a few snapshots to ensure that they were the most optimized ones. All implementation and training was based on the Caffe deep learning framework [22]. It took an average of 50 minutes to train a model on a machine with an Intel® Xeon CPU E3-1245 v5 and an NVIDIA Titan X GPU card. The same machine was used to measure the processing time of each network variant.

4.4 Results

As can be seen from Table 2, the RNN has the least number of parameters to learn. However, its performance is approximately 10% lower compared to one of its nearest neighbors for rank-1 accuracy on either dataset. The difference is noticeably large, such that it makes the RNN a poor choice for accurate matching.

Three LSTM variants, including vanilla LSTM, LSTMC, and LSTMP, showed similar performance. For the iLIDS-VID dataset, LSTMC achieved the highest rank-1 accuracy, while LSTM had the lowest of three. The order is reversed for other dataset. While the performance of the three varies by only 1% or 2%, the model size is notably different. Compared to the original LSTM, LSTMP adds three more weight vectors that depend on the output size of the recurrent network - 512 in our experiments. LSTMC drops one weight vector where the dimension depends on the size of the input, which is a 58950-element vector. Since an input dimension of 58950 is significantly higher than 512-dimension output, the resulting LSTMC model is 25% smaller than LSTM and LSTMP (which are almost the same), as shown in Table 3.

Table 2. Performance comparison of different recurrent architectures.

Dataset	iLIDS-VID				PRID 2011			
CMC Rank	1	5	10	20	1	5	10	20
RNN	34.4	64.8	76.8	87.5	44.0	76.1	88.7	96.2
GRU	**48.4**	**74.3**	**83.0**	**91.3**	**59.2**	**87.2**	**95.3**	**98.8**
LSTMC	46.2	72.2	81.4	90.3	53.8	81.5	92.6	97.8
LSTM [9]	44.5	71.9	82.0	90.1	54.9	84.2	93.7	98.4
LSTMP	45.7	71.8	81.9	90.2	54.1	81.8	91.5	97.8

Table 3. Size and processing time comparison.

Variant	Number of parameters	Training phase time (ms/iteration)			Testing phase time (ms/sequence)
		Batch size $N = 1$	Batch size $N = 8$	Batch size $N = 16$	
RNN	30,707,712	12.784	54.029	103.963	6.643
GRU	91,073,024	27.778	65.187	121.961	7.265
LSTMC	91,335,168	28.001	66.245	124.772	7.354
LSTM	121,780,224	36.694	69.664	132.197	8.462
LSTMP	122,566,656	37.977	71.861	134.724	9.629

Having the second smallest size, the GRU gives the best results in all ranks on both datasets. This architecture shows a distinctive difference to the four others. It achieved 48.4% and 59.2% rank-1 accuracy on iLIDS-VID and PRID 2011 datasets respectively. For iLIDS-VID, the gap between GRU and the second one (LSTMC) is 2.2%, which is not remarkable. However, with the less difficult dataset, the gap is more noticeable. Also, with such a compact size, the performance of GRU is the most impressive out of the five.

From Table 3, we can see that the processing time scales with size. For the testing phase, we recorded the time it takes to forward a sequence of 10 frames through the network. For the training phase, which includes forward and backward calculations, we recorded the time it takes to do both things with different batch sizes, N. However, when looking at the results, we can say that all variants are in order. If the size is big, it will take more time to train and test the model. In our experiment, the RNN is the best in terms of size and speed. But due to its poor performance, it is not a good choice. Thus, the most optimized choice is the GRU, which has the best performance and the second lowest processing time, as well as the second smallest size.

5. Conclusion

In this work, we implement five recurrent network architectures, including RNN, LSTM, LSTM with a coupled gate, LSTM with peephole connection, and GRU. Extensive experiments were conducted to compare performance in terms of size and accuracy for the video-based person re-identification task. RNN has the smallest size but performs more poorly than the others. LSTM and its slight variants have almost the same performance. Finally, GRU is the most optimized. It not only has a small number of parameters; it also gives the highest accuracy for all ranks for both datasets. The small number of parameters reasonably leads to a faster feed-forward process, which effectively enhances the performance of a practical re-identification system.

Acknowledgement

This research was supported by the Computer Architecture and Parallel Processing Lab, Department of Electrical and Computer Engineering, Seoul National University, Seoul, South Korea.

References

[1] Bedagkar-Gala and S. K. Shah, "A survey of approaches and trends in person re-identification," Image and Vision Computing, vol. 32, no. 4, pp. 270–286, 2014. Article (CrossRef Link)

[2] M. Farenzena, L. Bazzani, A. Perina, V. Murino, and M. Cristani, "Person re-identification by symmetry-driven accumulation of local features," in Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on. IEEE, 2010, pp. 2360–2367. Article (CrossRef Link)

[3] D. Gray and H. Tao, "Viewpoint invariant pedestrian recognition with an ensemble of localized features," Computer Vision–ECCV 2008, pp. 262–275, 2008. Article (CrossRef Link)

[4] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," IEEE Transactions on pattern analysis and machine intelligence, vol. 24, no. 7, pp. 971–987, 2002. Article (CrossRef Link)

[5] R. Zhao, W. Oyang, and X. Wang, "Person re-identification by saliency learning," IEEE transactions on pattern analysis and machine

intelligence, vol. 39, no. 2, pp. 356–370, 2017. Article (CrossRef Link)
[6] V. Le, Q. N. Hong, T. T. Quang, and N. D. Trung, "Superpixel-based background removal for accuracy salience person re-identification," in Consumer Electronics-Asia (ICCE-Asia), IEEE International Conference on. IEEE, 2016, pp. 1–4. Article (CrossRef Link)
[7] T. B. Nguyen, V. P. Pham, T.-L. Le, and C. V. Le, "Background removal for improving saliency-based person re-identification," in Knowledge and Systems Engineering (KSE), 2016 Eighth International Conference on. IEEE, 2016, pp. 339–344. Article (CrossRef Link)
[8] L. Wu, C. Shen, and A. v. d. Hengel, "Personnet: person reidentification with deep convolutional neural networks," arXiv preprint arXiv:1601.07255, 2016. Article (CrossRef Link)
[9] Y. Yan, B. Ni, Z. Song, C. Ma, Y. Yan, and X. Yang, "Person reidentification via recurrent feature aggregation," in European Conference on Computer Vision. Springer, 2016, pp. 701–716. Article (CrossRef Link)
[10] N. McLaughlin, J. Martinez del Rincon, and P. Miller, "Recurrent convolutional network for video-based person re-identification," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1325–1334. Article (CrossRef Link)
[11] H. Liu, Z. Jie, K. Jayashree, M. Qi, J. Jiang, S. Yan, and J. Feng, "Videobased person re-identification with accumulative motion context," arXiv preprint arXiv:1701.00193, 2017. Article (CrossRef Link)
[12] K. Greff, R. K. Srivastava, J. Koutn´ık, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," IEEE transactions on neural networks and learning systems, 2016. Article (CrossRef Link)
[13] T. Wang, S. Gong, X. Zhu, and S. Wang, "Person re-identification by discriminative selection in video ranking," IEEE transactions on pattern analysis and machine intelligence, vol. 38, no. 12, pp. 2501–2514, 2016. Article (CrossRef Link)
[14] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, vol. 1. IEEE, 2005, pp. 886–893. Article (CrossRef Link)
[15] O. Chapelle and S. S. Keerthi, "Efficient algorithms for ranking with svms," Information Retrieval, vol. 13, no. 3, pp. 201–215, 2010. Article (CrossRef Link)
[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997. Article (CrossRef Link)
[17] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with lstm recurrent networks," Journal of machine learning research, vol. 3, no. Aug, pp. 115–143, 2002. Article (CrossRef Link)
[18] K. Cho, B. Van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, ¨ H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," arXiv preprint arXiv:1406.1078, 2014. Article (CrossRef Link)
[19] T. Wang, S. Gong, X. Zhu, and S. Wang, "Person re-identification by video ranking," in European Conference on Computer Vision. Springer, 2014, pp. 688–703. Article (CrossRef Link)
[20] M. Hirzer, C. Beleznai, P. M. Roth, and H. Bischof, "Person reidentification by descriptive and discriminative classification," in Scandinavian conference on Image analysis. Springer, 2011, pp. 91–102. Article (CrossRef Link)
[21] M. Hirzer, P. Roth, M. Kostinger, and H. Bischof, "Relaxed pairwise ¨ learned metric for person re-identification," Computer Vision–ECCV 2012, pp. 780–793, 2012. Article (CrossRef Link)
[22] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in Proceedings of the 22nd ACM international conference on Multimedia. ACM, 2014, pp. 675–678. Article (CrossRef Link)

Nghia Nguyen Tuan is a fifth-year student at the School of Electronics and Telecommunications, Hanoi University of Science and Technology, Hanoi, Vietnam. His research interests are computer vision, deep-learning applications, pedestrian detection, and real-world person re-identification.

Cuong Vo Le received a PhD in ultra-high-speed and ultrahigh-sensitivity camera fields from Kinki University, Osaka, Japan, in 2009. From 2009 to 2011, he worked as a senior research engineer to develop software-defined radio devices at Viettel Group. Since 2011, he has been an Assistant Professor at Hanoi University of Science and Technology. From 2014 to 2015, he worked as a research associate in the Korea Global Frontier Project with the Center for Integrated Smart Sensors at KAIST to develop object detection and structured light deep extraction. His research topics are applications of deep learning, real-world person re-identification and image sensor design.

Quan Nguyen Hong is a first-year PhD candidate in the School of Electronics and Telecommunications, Hanoi University of Science and Technology, Hanoi, Vietnam. He received an MSc in Computer Science from the Military Technical Academy in 2010. Since 2004 he has been a lecturer at Vietnam - Hungary Industrial University. From 2003 to 2011, he worked as a software developer for the Hanoi Informatics and Telecommunication joint stock company. His research interests are computer vision, machine learning and deep learning.

Hyuk-Jae Lee received B.S. and M.S. degrees in Electronics Engineering from Seoul National University, Korea, in 1987 and 1989, respectively, and obtained a PhD in Electrical and Computer Engineering from Purdue University at West Lafayette, Indiana, in 1996. From 1998 to 2001, he worked at the Server and Workstation Chipset Division of Intel Corporation in Hillsboro, Oregon as a senior component design engineer. From 1996 to 1998, he was on the faculty of the Department of Computer Science of Louisiana Tech University at Ruston, Louisiana. In 2001, he joined the School of Electrical Engineering and Computer Science at Seoul National University where he is currently working as a Professor. His research interests are in the areas of computer architecture and SoC design for multimedia applications.