Loop-closure detection by LiDAR scan re-identification

Jukka Peltomäki*, Xingyang Ni*, Jussi Puura†, Joni-Kristian Kämäräinen*, and Heikki Huttunen*
*Tampere University, Finland, †Sandvik Mining and Construction Ltd

***Abstract*—In this work, loop-closure detection from LiDAR scans is defined as an image re-identification problem. Re-identification is performed by computing Euclidean distances of a query scan to a gallery set of previous scans. The distances are computed in a feature embedding space where the scans are mapped by a convolutional neural network (CNN). The network is trained using the triplet loss training strategy. In our experiments we compare different backbone networks, variants of the triplet loss and generic and LiDAR specific data augmentation techniques. With a realistic indoor dataset the best architecture obtains the mean average precision (mAP) above 0.94.**

I. Introduction

Loop-closure is an important sub-problem in robot navigation and mapping since visiting the same location again allows to reduce the map and location uncertainties. This is particularly important for the task of Simultaneous Localization and Mapping (SLAM) where the robot simultaneously builds a map and estimates its location on the map [1]. SLAM is based on *incremental localization* where current location is estimated from the last known global pose [2] and uncertainties accumulate during the process. When the loop-closure is detected the uncertainties can be assigned small values in these locations and propagated to the nearby locations. Visual loop-closure is particularly important indoors where global positioning system (GPS) is not available. Vision-based loop-closure detection is performed by comparing the image of the current location to the images captured from previous locations. In this sense, vision-based loop-closure detection comes down to an online image retrieval task.

A popular method for loop-closure detection is the visual Bag-of-Words (BoW) [3], [4], [5]. The BoW loop-closure detection has been shown to work for large scale data [6] and it has been combined with depth data [7]. However, following the recent trend in computer vision the more recent methods are based on convolutional neural network (CNN) embedding where the feature layers provide a feature vector for image matching. The re-identification architectures adopt the image recognition [8], [9] or the autoencoder structure [10], [11]. Xia et al. provide a comparison of various approaches in [12].

In this work, the type of input data differs substantially from the previous works where conventional RGB images are used. Our input is panoramic intensity images obtained from the intensity channel of a high quality Ouster OS1 LiDAR sensor (Figure 1). The intensity images are of particularly low resolution (64 by 2048 pixels), but are invariant to many imaging distortions such as lighting and shadows since the intensity correlates with material properties. LiDAR scan results to a $360^\circ \times 33^\circ$ view angle image which is suitable for navigation purposes of heavy machinery in indoor work sites and mines. The image matching for loop-closure detection is cast as a re-identification problem. Image re-identification has been successfully used in vehicle recognition [13], [14] and face recognition [15], [16], [17].

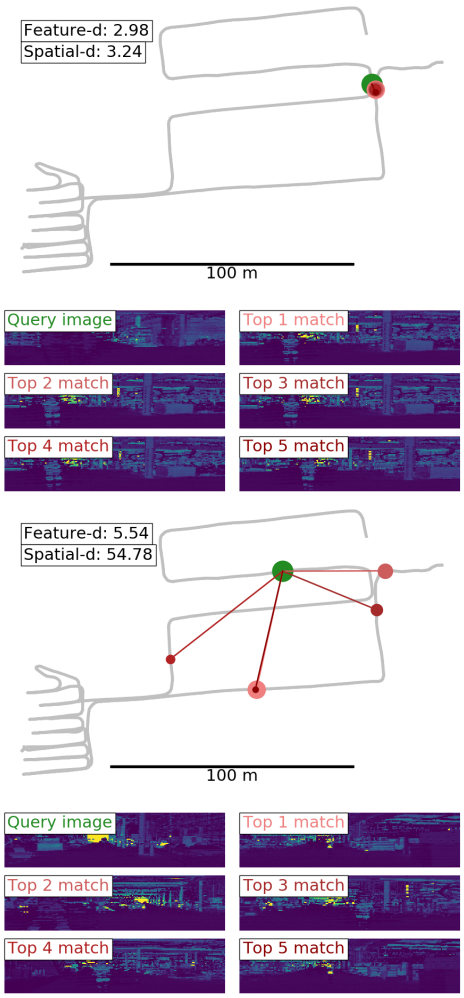

Fig. 1. Examples of loop closure (top) and non-loop closure locations (bottom) where our method effectively detects the loop closure. In the loop-closure location the best five matches (red points) are all near the query spatial location (green point) and within a small feature distance while in the non-loop-closure location the feature distances are large and the matches are found in random spatial locations.

Contributions – The main contributions of this work are:

- We propose feature space embedding for distance based matching of panoramic LiDAR intensity scans. Embedding is computed using a convolutional neural network (CNN) architecture trained using the triplet loss training strategy.
- We provide an experimental comparison of backbone networks, variants of the triplet loss and generic and LiDAR scan specific data augmentation techniques with a realistic data collected from the path of 1,026 meters in industrial environment.

II. Related work

Ability to navigate in known or unknown environments is an essential skills and research challenge in mobile robotics. There are various sensor modalities available, such as sonars, but substantial efforts have been dedicated to vision based navigation since 1970's [2]. Navigation can be sub-divided to map-based navigation and map building, but with the help of spatial uncertainty modeling [18], [19] the simultaneous localization and mapping (SLAM) that combines the two sub-tasks has become an important technique [1]. A critical task in SLAM is to correctly associate observations of landmarks (locations) with landmarks held in the map. Incorrect association can lead to catastrophic failure of the SLAM algorithm, but successful association helps to reduce uncertainties. Data association is particularly important when a vehicle returns to a previously mapped region after a long excursion, the so-called *loop-closure problem* [20].

Vision-based loop-closure detection – The loop-closure problem can be divided to *loop-closure detection* and *loop closing* where the first refers to detection whether the current observation is from a previously visited location or not and the second to data association where the map and location uncertainties are updated. In our work we focus on the loop-closure detection only. One of the first vision-based SLAM method was introduced by Cummins and Newman [4]. They introduced the Fast Appearance-based Mapping (FAB-MAP) algorithm that was inspired by the Bag of Visual Words [21] image classification approach. First a vocabulary of visual words is established from data and then every scene is represented as a histogram of the found words. Histogram features are matched by a distance function and loop-closure is detected by setting a match threshold. A similar approach during the same time was proposed by Angeli et al. [3] and these both works provide methods for the both loop-closure detection and data association. FAB-MAP 2.0 [6] added an inverted index for sparse approximation which boosted the matching speed and scalability so that they did not anymore restrict the map size. A 3D FAB-MAP 3D was introduced in [7]. In 3D FAB-MAP the camera image is augmented with depth information that gives the visual words a relative 3D position. With the help of depth information they were able to improve the loop closure recall from 0.42 to 0.71 with the same dataset. A number of different approaches were compared for monocular visual SLAM in [22].

In recent works, Convolutional Neural Network (CNN) based feature embedding [23] has replaced the BoW features [24] in image retrieval and in loop-closure detection. Hou et al. [8] compared hand-crafted features to CNN-learned features for loop-closure detection. They found that fully connected layers are not useful for the task by systematically testing features from different layers. In their experiments, the features from the last pooling layer were the best for image matching. Their network was trained for the scene classification task. Unsupervised visual loop closure method was introduced by Merrill and Huang in 2018 [11]. Their method is based on an auto-encoder CNN, where multiple image transformations are created and used for training. One of the image transforms is the training input, while another transform of the same image is used to calculate the histogram of oriented gradients (HOG), which is then used as the ground truth to be learned.

LiDAR intensity images have also been studied for localization by Bârsan et al. in 2018 [25], where they introduced a real-time localization method for self-driving cars, combining LiDAR intensity images with LiDAR scans to a combined embedding.

Image re-identification – Image re-identification is closely related to image retrieval [24], [23] where a query image is matched against a gallery set to find whether the same object or place appears in the gallery. Suitable datasets for robotics are those containing real places and scenes. Gomez et al. [9] explored the CaffeNet CNN with a triplet loss function to train the network for appearance-invariant place recognition. Noh et al. [26] introduced attentive deep local features for learning local feature descriptors. They also released a big Google-Landmarks dataset with over a million images and almost 13,000 different landmarks.

Various other applications and datasets not related to places and scenes also exist. For example, Lou et al. [13] studied vehicle re-identification and proposed a hard negative mining scheme for visually similar images. They released the VERI-Wild dataset of over 400,000 images of over 40,000 vehicles. Kuma et al. [14] provide a solid benchmarking of vehicle re-identification and experiment with different loss functions. They acknowledge that vehicle re-identification is different from, for example, face re-identification, as vehicles are coarser in details and two cars with the same model and color are very hard to distinguish without additional data (such as a visible licence plate). Face re-identification is another popular application. Schroff et al. [15] introduced FaceNet which inspired the network architecture and training setup used in our work. For face re-identification, Ustinova et al. [16] propose a hybrid architecture of a CNN and a bilinear CNN.

III. Method

For loop-closure detection problem, we train a deep neural network for the task of image re-identification. Our images are panoramic (360-degree) images generated from the intensity channel of LiDAR scans. Re-identification is effectively and

efficiently solved by learning a mapping from images to a compact Euclidean space where distances directly correspond to semantic similarity, i.e. whether this location is already visited or not.

Training of an effective embedding network requires a suitable architecture, loss function, and hard negative/positive mining. The two popular approaches are the Siamese structure trained with the contrastive loss [27] and a single CNN pipeline trained with the triplet loss [15]. In this work, we adopt the triplet loss approach which is more difficult to implement but is shown to perform well in face and vehicle re-identification [15], [14] and place recognition [28]. The triplet loss has also been shown to outperform many recent loss functions for person re-identification by large margins [29]. Moreover, we experiment variants of the triplet loss: lifted structured loss [30] and Hermans triplet loss [29].

Positive and negative samples - As the main difference to the above works on face, vehicle, and places recognition we need to define the meaning of positive match between two LiDAR scans. In our dataset, this is achieved by setting a spatial distance threshold $\tau_{pos} = 4.0m$ (four meters) which means that each sample s_i within four meters $dist(s_i, s_j) < \tau_{pos}$ is defined as a positive sample for the query scan s_j and all other as negative samples. During the data capture the spatial distances were obtained through an indoor positioning system available in the industrial work site (Section IV).

A. Backbone network

The backbone network used with the triplet loss has a strong impact on the training speed and accuracy and the final test performance. Various backbone networks have been used in literature and they mainly differ in the number and size of layers and the types of pooling layers. We adopt the procedure from other similar works and use only the features from the convolutional part of the network and ignore the final fully-connected layers. We add a global average pooling layer the top of backbones [36] . This gives us the output flattened as a vector, the size of which is dependent on the backbone architecture and input size. We can conveniently control the size of the output vector by having a stack of convolution, batch normalization [37], and ReLU [38] layers between the backbone and the global average pooling layer. For the lifted structured loss and the Hermans triplet loss we also added a batch normalization layer after the average pooling to make the networks perform properly.

After training the backbone network for image re-identification the network is used to extract a global feature vector from a query image (current location) which is matched against the gallery vectors (previous locations). For matching, the closest matches are found using the Euclidean distance which is fast to compute from the query to all gallery vectors. The backbone networks used in our experiments are listed in Table I.

TABLE I
THE BACKBONE NETWORKS USED IN OUR EXPERIMENTS FOR LIDAR SCAN RE-IDENTIFICATION. THE FEATURE VECTOR DIMENSION IS MANUALLY RESTRICTED FOR OTHER EFFICIENTNET VARIANTS EXCEPT B3. THE TOTAL NUMBER OF PARAMETERS IS COUNTED BY SUMMING THE BACKBONE PARAMETERS AND THE PARAMETERS IN EXTRA LAYERS TO ACCOMMODATE THE FEATURE DIMENSIONS.

Backbone	$\dim(f)$	*# of params*
MobileNet V2 [31]	1280	2.3 M
ResNet 50 [32]	2048	23.6 M
SEResNet 50 [33]	2048	26.1 M
DenseNet 121 [34]	1024	7.0 M
EfficientNet B0 [35]	1024	5.4 M
EfficientNet B1 [35]	1024	7.9 M
EfficientNet B2 [35]	1024	9.2 M
EfficientNet B3 [35]	1024	12.4 M
EfficientNet B3 [35]	1536	10.8 M

B. Loss functions

In image re-identification the network topology is coupled with a metric learning loss function that minimizes Euclidean distances in the feature vector space for images that are close in the spatial space, and maximizes for ones further away. Since we selected the triplet loss function and a single backbone architecture we experimented with the three loss functions that can be considered as variants of the triplet loss: *triplet loss* [15], *lifted structured loss* [30] and *Hermans triplet loss* [29].

Triplet loss – Before FaceNet [15] the triplet loss was already used by Weinberger and Saul [39] for clustering and later in a number of other works [40], [41]. Our work is similar to FaceNet in the sense that we apply the triplet loss directly on top of the convolutional part of the backbone network without classification layers.

The triplet loss is based on three data samples selected so that the first is an anchor (a), the second is a positive match (p) (same location as the anchor), and the third is a negative match (n) (different location). The loss function minimizes the squared Euclidean distance from the anchor to the positive sample and maximizes the distance from the anchor to the negative sample,

$$L = \sum_{i}^{N} \left[||f(x_i^a) - f(x_i^p)||_2^2 - ||f(x_i^a) - f(x_i^n)||_2^2 + \alpha \right]_+ \quad , \tag{1}$$

where α sets the distance margin that the loss function tries to maintain between the locations. The subscript '+' denotes the positive numbers only and therefore the minimum value for the loss is zero when ideally the first term is zero and the second term is $\geq \alpha$.

Hermans triplet loss – Hermans triplet loss is an improvement proposed by Hermans et al. [29] for the face re-identification task. Hermans triplet loss does not require offline

hard negative mining which makes the loss function itself more complicated:

$$L_{LG}(\theta; X) = \sum_{i=1}^{P}\sum_{a=1}^{K}\Bigg[\log\sum_{\substack{p=1\\p\neq a}}^{K} e^{D\left(f_\theta(x_a^i), f_\theta(x_p^i)\right)} + \log\sum_{\substack{j=1\\j\neq i}}^{P}\sum_{n=1}^{K} e^{\alpha - D\left(f_\theta(x_a^i), f_\theta(x_n^j)\right)}\Bigg]_+ . \quad (2)$$

In (2) D is a distance function that does not necessarily need to be the squared Euclidean distance. The function f_θ maps the semantically close points in the data manifold to a metrically close points in the feature space, with the parameters θ. In our case, the f_θ is the neural network. The X denotes the batch of data we are learning from.

Lifted structured loss – Lifted structured embedding loss by Oh et al. [30] implements a similar concept to the triplet loss, but utilizes a combination of all images in each batch. Instead of comparing only three images (A+P+N) with each other, in the lifted structured loss all the images in the same batch are compared to each other. As shown in the following equation all pairings of the negative and positive samples are compared:

$$\tilde{J}_{i,j} = \log\left[\sum_{(i,k)\in N} e^{(\alpha - D_{i,k})} + \sum_{(j,l)\in N} e^{(\alpha - D_{j,l})}\right] + D_{i,j}$$
$$\tilde{J} = \frac{1}{2|P|}\sum_{(i,j)\in P}\max(0, \tilde{J}_{i,j})^2 . \quad (3)$$

C. Data augmentation

Our data augmentation schemes include popular 2D image augmentation techniques and special techniques available for panoramic images. If not otherwise indicated, the experimental results are for all below augmentation techniques enabled.

2D image augmentation – For improving generalization, we employ the standard 2D image augmentation methods during training. Specifically, we use random erasing [42], horizontal flipping as proposed by Simonyan and Zisserman [43], and random cropping proposed by Krizhevsky et al. [44].

Random panoramic rotation – Images constructed from the LiDAR scans span the full 360 degree circle around the sensor (panoramic view). Panoramic view enables a simple augmentation scheme through random rotations. In specific, the Yaw angle of the camera was randomly rotated by $[0°, 360°[$. Rotation was implemented as a simple pixel shift of the image.

Direction flipping – Direction flipping is a variant of the random panoramic rotation. In direction flipping the yaw angle is changed to the opposite direction ($180°$) to simulate vehicle running to the opposite direction. The probability of direction flipping was set to 50% and it was implemented similar to the rotation augmentation using pixel shifts.

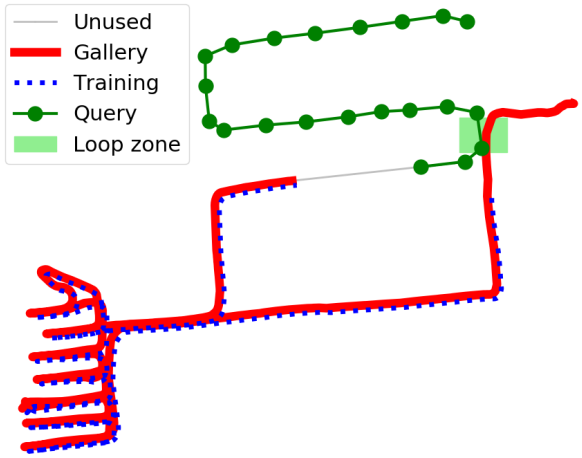

Fig. 2. Dataset split visualized along the spatial path of image capture. Training and gallery sets share significant portion with each other, but the query images as well as substantial zones before and after the gallery images (including the loop-closure zone) are not used in the training set.

D. Training details

Network training and testing were implemented in Python using Keras and TensorFlow. All networks were trained for 200 epochs which was clearly sufficient for convergence for all the networks. The Adam optimizer was used.

The batch size was set according to the available GPU memory (11 GB). The original 2048×64 LiDAR intensity images were scaled down to 512×64 to allow faster and more stable training via bigger batches. For most networks we used 12 anchor images per batch, each with 8 positive examples (within 4.0 m from the anchor) per anchor. For Densenet-121 the anchors per batch was dropped to 10 and for EfficientNets to 8, in order to fit into the memory.

The training samples were batched by taking the specified amount of random anchor points along the training set, and randomly taking corresponding positive match points from within τ_{pos} as defined in the beginning of this section.

IV. Data

LiDAR capturing – For gathering the data, we used a first generation Ouster OS1 LiDAR sensor by Ouster Inc. The Ouster OS1 (1st gen) is a mid-range high resolution imaging LiDAR. The minimum range is 0.8 meters and the maximum range is 120 meters. With an 80% scene reflectivity the sensor can detect to 105 meters with a detection probability of over 90 %. With 10% reflectivity and 90% probability, the range is 40 meters. The sensor captures a 360 by 33.2 degree field of view, and can output 2D images at a maximum of horizontal resolution of 2048 pixels at 10 Hz, and 1024 pixels at 20 Hz. The amount of channels used corresponds to the vertical pixel resolution, and can be chosen to be 16, 32, or 64. The Ouster LiDAR uses intrinsic calibration, has fixed resolution per frame, and boasts a camera-grade ambient and intensity data.

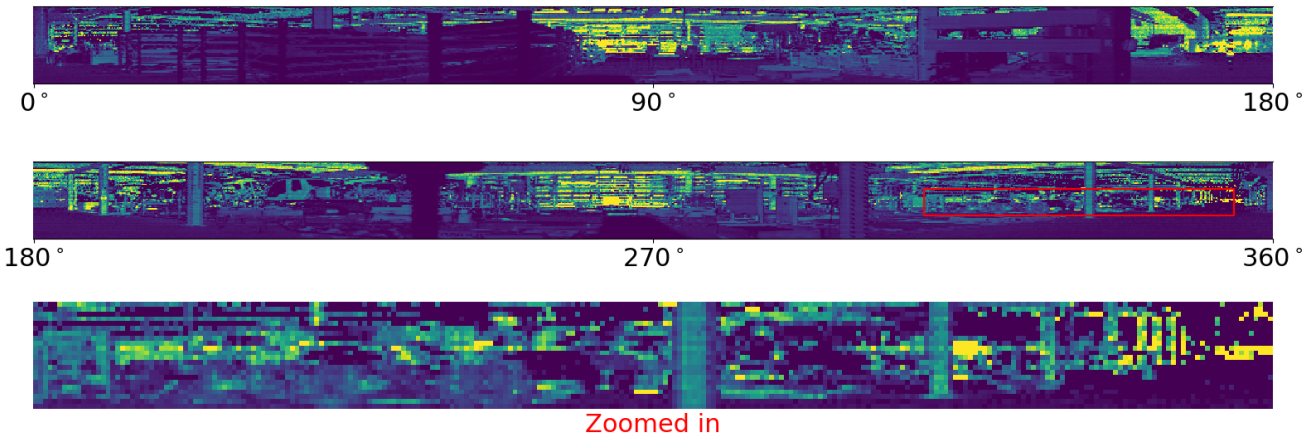

Fig. 3. Example 2D intensity image output from the Ouster OS1 (first gen) LiDAR. It is 2048 x 64 pixels big and greyscale. The image represents the LiDAR measurements from 360 by 33.2 degree angles. It is hard to see details in the unedited format, as in the test scene most details show up quite similar and very dark. The image shown here is colored, brightened up, and split into two 1024 x 64 slices. There is also a zoomed portion (source highlighted in red) to better show off the sensor detail.

We generated our own dataset consisting of a single continuous path in a decently lit indoors industrial environment (a factory). The sensor was mounted to a wheeled stable platform for capture. The height and orientation of the sensor was fixed within the platform. The scale of the environment was well matched to the mid-range specification of the OS1 sensor. We used the highest possible resolution. The path fits into a 173 by 135 meter area, and has less than one meter of vertical range. The total length of the route is 1,026 meters.

The data from the sensor was exported as intensity images. The set was culled to 9,707 images. The corresponding position data are x,y,z coordinates in meters, which is used to generate the positive and negative ground truth labels. The position data was collected via a proprietary high quality SLAM system. The images and the position data was fused together to finalize the dataset. There are multiple loops in the path. The data points are nearly uniformly distributed along the collection path, as there is only slight variance in the distances between data points.

The Ouster LiDAR conveniently outputs LiDAR attribute channels as well as 3D point clouds. In this work we utilized the panoramic intensity images. The sensor is also capable of outputting depth and ambient 2D images, which were not used. The images were directly used as the network inputs. The intensity images are 8-bit greyscale and the resolution of 2048×64.

Evaluation protocol – The dataset was divided into training (6,600 images), gallery (7,200 images), and query (2,100 images) sets visualized in Figure 2. The training and gallery sets are overlapping. There is also an unused section (300 images) between query and gallery set paths to prevent trivially easy matching in the beginning of the query set. Our gallery set includes a prominent loop area that has a corresponding half in the query dataset, but is excluded from the training set. This loop point is manually tagged as being a loop point, while the non-loop points are tagged as being non-loop points. This was thought to be the hardest realistic way to test for loop detection with our dataset.

The mean average precision (mAP) calculation is based on the fact that we manually determined which samples in the query and gallery sets formed a loop and which did not. This gave us a way to determine if the network feature matching gallery images for a given query were either correct (query and gallery points both in the loop zone or both off it) or incorrect (query and gallery points in different zones). This allowed us to use the standard way to calculate mAP.

All the mAP figures presented in Section V are based on a full query set and on the top-1 match (closest in the network computed feature space) from the gallery set. Matches within $\tau_p < 4.0\ m$ are counted as correct recall.

V. Experiments

A. Backbone network

In the first experiment, a number of backbone networks and different loss functions were tested using our data (Section IV). All backbone networks had pre-trained weights using the ImageNet data. The networks were fine-tuned using our training data (Section IV). In these experiments the random panoramic rotation augmentation was applied (Section III-C) with data augmentation procedures of random erasing, horizontal flip, and random cropping. The images were downscaled to 512x64 resolution to allow for bigger batches within available GPU memory. The batch size was 96, but for DenseNet-121 the size was reduced to 80, and for EfficientNet to 64.

The backbone networks from Section III-A were tested: MobileNet V2 [31], Resnet 50 [32], DenseNet 121 [34], EfficientNet [35], and SEResNet 50 [33]. The networks were

tested with the different triplet loss variants from Section III-B: lifted structured loss [30], Hermans triplet loss [29] and the triplet loss [15]. The results for all combinations are in Table II.

TABLE II
TOP-1 RESULTS (MAP) FOR DIFFERENT BACKBONE NETWORKS AND DIFFERENT LOSS FUNCTIONS. DIM(f) IS THE FEATURE VECTOR DIMENSION. EFFICIENTNET HAS MULTIPLE IMPLEMENTATIONS DENOTED BY B0-B3 [35]. IN THIS EXPERIMENT ONLY THE BEST MATCHES ARE USED (TOP-1).

Backbone	dim(f)	*Loss function*		
		Lifted Structured	Hermans triplet	Triplet
MobileNet V2	1280	0.711	0.239	0.460
ResNet 50	2048	0.589	0.754	0.640
SEResNet 50	2048	0.686	0.636	0.685
DenseNet 121	1024	0.607	0.604	0.695
EfficientNet B0	1024	0.583	0.756	0.758
EfficientNet B1	1024	0.389	**0.946**	0.732
EfficientNet B2	1024	0.847	0.908	0.765
EfficientNet B3	1024	0.918	0.759	0.734
EfficientNet B3	1536	0.846	0.511	0.822

The EfficientNet variants achieved the best mAP across all three loss functions: 0.918 for lifted structured loss with B3 (1024), 0.946 for Hermans triplet loss with B1 (1024), and 0.822 for conventional triplet loss with B3 (1536). SEResNet 50 and DenseNet 121 performed consistently mAPs ranging from 60% to 70% across all three loss functions. MobileNet V2 performed best with the lifted structured loss, and ResNet 50 with the Hermans triplet. The overall best performance was achieved with EfficientNet B1 (1024) and Hermans triplet loss (mAP 0.946).

To visualize the image re-identification based loop closure the top-1 Euclidean feature space distances are plotted to Figure 4. It is clear the the Euclidean distances are substantially lower in the loop closure region of the test (query) data, which indicates that the learned embedding represents discriminatively the images of different locations in our LiDAR intensity dataset.

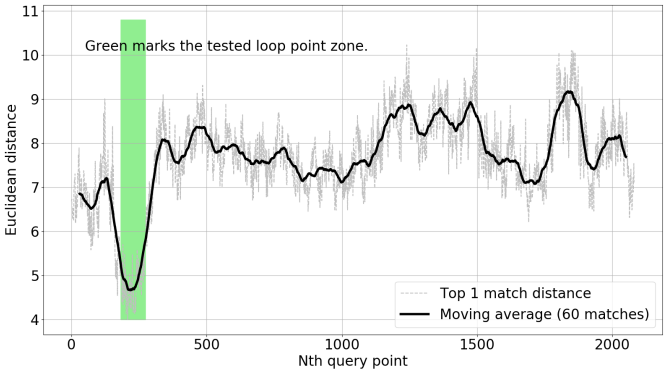

Fig. 4. Query-gallery feature space distances (Euclidean) for the dataset. The green background highlights the loop zone (see Figure 2). The loop-closure is easily detectable from the feature space distances in the network embedding space. The distances in the graph are computed in the space obtained using EfficientNet B1 with Hermans triplet loss and provides mAP of 0.946. (mAP calculation is described in Section IV - "Evaluation protocol")

B. Data augmentation

TABLE III
PERFORMANCE OF THE STRONGEST BASELINE (EFFICIENTNET B1 DIM(f) = 1024 W/ HERMANS TRIPLET LOSS) WITH VARIOUS COMBINATIONS OF DATA AUGMENTATION (SECTION III-C).

Augmentation	*mAP*
Baseline - EfficientNet B1 (1024)	0.884
+ random erasing	0.471
+ random crop	0.728
+ horizontal flip	0.402
+ random panoramic rotation	0.825
+ random erasing	0.205
+ all above	**0.946**

We experimented with the image augmentation techniques in Section III-C: random erasing, horizontal flip (50% chance), and random crop. The results are shown in Table III. The results clearly indicate the importance of data augmentation. It is clear that the most effective data augmentation is achieved by enabling all augmentation techniques which yielded to the highest performance (mAP 0.946). Interestingly, the other combinations of the augmentation techniques degraded the performance as compared to the baseline without data augmentation (mAP 0.884).

To study further the complementary nature of the 2D and panoramic augmentation techniques we conducted another set of experiments with various combinations. The results are shown in Table IV. Clear, the only combination that is clearly superior to the baseline using no augmentation is the one that combines all 2D augmentation techniques and the the random panoramic rotation.

VI. CONCLUSION

The main goal of this work was to find the best network architecture, loss function, and data augmentation for CNN-based metric feature embedding so that the embedding can be used in LiDAR image loop-closure detection. Potential applications are heavy machinery localization and SLAM in industrial indoor work sites. For experiments we collected a realistic dataset with an industry quality LiDAR.

We formulated embedding network optimization as an image re-identification problem and adopted the triplet loss as the objective function. The best performance, mAP 0.946, was obtained using EfficientNet B1 as the backbone network, using the Hermans triplet loss function and the following data augmentation techniques: random panoramic rotation, random erasing, random cropping, and random flipping.

Our future work will include collection of large scale public datasets, long-term localization, and integration of the proposed vision-based LiDAR loop-closure detection to a real robot navigation and SLAM.

TABLE IV

THE PERFORMANCE OF DIFFERENT AUGMENTATION SCHEMES AS TESTED ON EFFICIENTNET B1 (1024) BACKBONE AND HERMANS TRIPLET LOSS. THE COLUMNS REPRESENT THE COMMON 2D IMAGE AUGMENTATION TECHNIQUES AND THE ROWS ARE THE SPECIFIC AUGMENTATIONS TECHNIQUES FOR PANORAMIC LIDAR DATA. THE BEST COMBINATION IS RANDOM ERASING, RANDOM CROPPING, AND HORIZONTAL FLIPPING COMBINED WITH THE RANDOM PANORAMIC ROTATION (0.946 MAP).

	None	**Erasing**	**Cropping**	**Erasing & Cropping**	**Erasing & Cropping & Horizontal Flipping**
None	0.884	0.471	0.312	0.728	0.402
Panoramic direction flipping	0.669	0.526	0.847	0.539	0.420
Random panoramic rotation	0.824	0.205	0.493	0.173	**0.946**

REFERENCES

[1] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part i," *IEEE Robotics & Automation Magazine*, vol. 13, no. 2, 2006.

[2] G. DeSouza and A. Kak, "Vision for mobile robot navigation: A survey," *PAMI*, vol. 24, no. 2, 2002.

[3] A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer, "A fast and incremental method for loop-closure detection using bags of visual words," *IEEE Transactions on Robotics*, pp. 1027–1037, 2008.

[4] M. Cummins and P. Newman, "Fab-map: Probabilistic localization and mapping in the space of appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[5] A. Glover, W. Maddern, M. Warren, S. Reid, M. Milford, and G. Wyeth, "Openfabmap: An open source toolbox for appearance-based loop closure detection," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 4730–4735.

[6] M. Cummins and P. Newman, "Appearance-only slam at large scale with fab-map 2.0," *The International Journal of Robotics Research*, vol. 30, no. 9, pp. 1100–1123, 2011.

[7] R. Paul and P. Newman, "Fab-map 3d: Topological mapping with spatial and visual appearance," in *2010 IEEE international conference on robotics and automation*. IEEE, 2010, pp. 2649–2656.

[8] Y. Hou, H. Zhang, and S. Zhou, "Convolutional neural network-based image representation for visual loop closure detection," in *2015 IEEE international conference on information and automation*. IEEE, 2015, pp. 2238–2245.

[9] R. Gomez-Ojeda, M. Lopez-Antequera, N. Petkov, and J. Gonzalez-Jimenez, "Training a convolutional neural network for appearance-invariant place recognition," *arXiv preprint arXiv:1505.07428*, 2015.

[10] X. Gao and T. Zhang, "Loop closure detection for visual slam systems using deep neural networks," in *2015 34th Chinese Control Conference (CCC)*. IEEE, 2015, pp. 5851–5856.

[11] N. Merrill and G. Huang, "Lightweight unsupervised deep loop closure," *arXiv preprint arXiv:1805.07703*, 2018.

[12] Y. Xia, J. Li, L. Qi, H. Yu, and J. Dong, "An evaluation of deep learning in loop closure detection for visual slam," in *2017 IEEE international conference on internet of things (iThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData)*. IEEE, 2017, pp. 85–91.

[13] Y. Lou, Y. Bai, J. Liu, S. Wang, and L. Duan, "Veri-wild: A large dataset and a new method for vehicle re-identification in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3235–3243.

[14] R. Kuma, E. Weill, F. Aghdasi, and P. Sriram, "Vehicle re-identification: an efficient baseline using triplet embedding," in *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2019, pp. 1–9.

[15] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[16] E. Ustinova, Y. Ganin, and V. Lempitsky, "Multi-region bilinear convolutional neural networks for person re-identification," in *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, 2017, pp. 1–6.

[17] Y. Wang, J. Shen, S. Petridis, and M. Pantic, "A real-time and unsupervised face re-identification system for human-robot interaction," *Pattern Recognition Letters*, vol. 128, pp. 559–568, 2019.

[18] R. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *Int. J. of Robotics Research*, vol. 5, no. 4, 1986.

[19] H. Durrant-Whyte, "Uncertain geometry in robotics," *IEEE J. of Robotics and Automation*, vol. 4, no. 1, 1988.

[20] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part ii," *IEEE Robotics & Automation Magazine*, vol. 13, no. 3, 2006.

[21] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Automatic differentiation in pytorch," in *ECCV Workshops*, 2004.

[22] B. Williams, M. Cummins, J. Neira, P. Newman, I. Reid, and J. Tardos, "A comparison of loop closing techniques in monocular SLAM," *Robotics and Autonomous Systems*, vol. 57, 2009.

[23] F. Radenović, G. Tolias, and O. Chum, "CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples," in *ECCV*, 2016.

[24] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Obj. retrieval with large voc. and fast spatial matching," in *CVPR.*, 2007.

[25] I. A. Barsan, S. Wang, A. Pokrovsky, and R. Urtasun, "Learning to localize using a lidar intensity map," in *Proceedings of The 2nd Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, A. Billard, A. Dragan, J. Peters, and J. Morimoto, Eds., vol. 87. PMLR, 29–31 Oct 2018, pp. 605–616. [Online]. Available: http://proceedings.mlr.press/v87/barsan18a.html

[26] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han, "Large-scale image retrieval with attentive deep local features," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 3456–3465.

[27] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 539–546.

[28] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[29] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017.

[30] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4004–4012.

[31] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[33] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[34] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[35] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *arXiv preprint arXiv:1905.11946*, 2019.
[36] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.
[37] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.
[38] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010.
[39] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification." *Journal of Machine Learning Research*, vol. 10, no. 2, 2009.
[40] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1386–1393.
[41] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *International Workshop on Similarity-Based Pattern Recognition*. Springer, 2015, pp. 84–92.
[42] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation." in *AAAI*, 2020, pp. 13 001–13 008.
[43] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[44] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.