JIST: Joint Image and Sequence Training for Sequential Visual Place Recognition

Gabriele Berton, Gabriele Trivigno, *Graduate Student Member, IEEE*, Barbara Caputo, and Carlo Masone

***Abstract*—Visual Place Recognition aims at recognizing previously visited places by relying on visual clues, and it is used in robotics applications for SLAM and localization. Since typically a mobile robot has access to a continuous stream of frames, this task is naturally cast as a sequence-to-sequence localization problem. Nevertheless, obtaining sequences of labelled data is much more expensive than collecting isolated images, which can be done in an automated way with little supervision. As a mitigation to this problem, we propose a novel Joint Image and Sequence Training (JIST) protocol that leverages large uncurated sets of images through a multi-task learning framework. With JIST we also introduce SeqGeM, an aggregation layer that revisits the popular GeM pooling to produce a single robust and compact embedding from a sequence of single-frame embeddings. We show that our model is able to outperform previous state of the art while being faster, using eight times smaller descriptors, having a lighter architecture and allowing to process sequences of various lengths.**

***Index Terms*—Representation learning, simultaneous localization and mapping, visual information retrieval.**

I. Introduction

LOCALIZATION is a fundamental functionality for autonomous mobile robots, and one of its key ingredients is Visual Place Recognition (**VPR**) [1], i.e., the task of matching a current visual observation (an image or video stream) to previously visited places. For example, VPR is used for loop closure detection in SLAM [2], for re-localization in the kidnapped robot problem [3] and also for pure localization when a map is already available [4] and when GNSS measurements are precluded [5], [6]. Additionally, VPR is used to select rough candidates for precise 6-DoF pose estimation (i.e., visual localization) [4], [7].

Across these robotics applications, VPR is typically performed using methods that process short sequences of images acquired by cameras onboard the robot - what is called **sequence-to-sequence** or **seq2seq** place recognition [8]. A recent trend in this sense is to frame the seq2seq problem as a retrieval task on learnt embeddings (*sequence descriptors*) that represent entire sequences rather than individual frames [9], [10], [11], [12]. This new paradigm not only intrinsically captures the temporal information in the video stream, but it is also more efficient than individually matching each frame with previous observations [11], [12]. However, the accuracy and robustness achieved by sequence descriptors is bounded by the limited availability of large datasets of sequences. Indeed, for the classic **image-to-image** VPR (**im2im** [8]) the availability of massive datasets has been instrumental in setting the latest state of the art [13], [14], producing descriptors that generalize better and are very compact[1]. Yet, due to difficulties in curating sequences [8], [16], the largest dataset currently available for the seq2seq task (Mapillary Street Level Sequences [8]) is 40 $\times$ smaller than the largest datasets for image-to-image VPR [13], [17].

Given the correlation between the seq2seq and the im2im tasks, we argue that it is possible to produce more effective sequence descriptors by jointly training a model not only on sequences, but also on the readily available massive datasets for image-to-image VPR: on one hand, the im2im training from huge-scale datasets would improve the model's generalizability; on the other, sequence-to-sequence learning would embed the model with robustness to sequentially changing scenes and teach it how to temporally aggregate frame-level information. To this end, we propose a new training methodology that jointly uses images and sequences and exploits a state-of-the-art architecture originally developed for im2im VPR to first extract discriminative embeddings from individual frames and then aggregate them. While this new training method enables the model to effectively learn also from large datasets for the im2im tasks, it does not automatically solve the issue of large-dimensional embeddings required by previous SOTA [12]. To address this issue, in Section III-C we introduce a new aggregation layer called **SeqGeM**, that revisits the popular generalized mean pooling [18] by applying it along the temporal axis, resulting in very compact descriptors and, consequently, speeding-up the matching time (see Fig. 1). The combination of this training method and SeqGeM takes the name of Joint Image and Sequence Training, or **JIST**.

To summarize, we bring the following contributions:

- We propose a novel multi-task training framework to leverage existing large scale datasets of image-to-image VPR and improve upon the seq2seq task [8];

Manuscript received 13 July 2023; accepted 28 November 2023. Date of publication 4 December 2023; date of current version 28 December 2023. This letter was recommended for publication by Associate Editor Y. Liao and Editor S. Behnke upon evaluation of the reviewers' comments. This work was supported in part by Consorzio Interuniversitario Nazionale per l'Informatica (CINI) Roma, Italy. *(Gabriele Berton and Gabriele Trivigno contributed equally to this work.) (Corresponding author: Gabriele Berton.)*

The authors are with the Visual and Multimodal Applied Learning Lab, Department of Control and Computer Engineering, Politecnico di Torino, 10138 Torino, Italy (e-mail: gabriele.berton@polito.it; gabriele.trivigno@polito.it; caputo@dis.uniroma1.it; car.masone@gmail.com).

The code is available at https://github.com/ga1i13o/JIST.

Digital Object Identifier 10.1109/LRA.2023.3339058

[1]This also entails a reduced latency, because the size of descriptors has a linear correlation with the matching time of the retrieval [15].

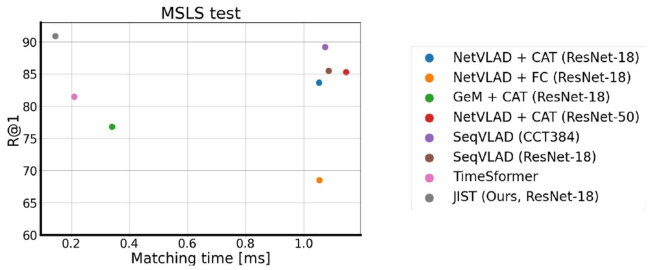

Fig. 1. Our multi-task training framework allows to surpass previous SOTA in performance. Thanks to our novel layer SeqGeM, we are able to cut down the matching time by an order of magnitude.

- We introduce the SeqGeM aggregation layer, which revisits the popular generalized mean pooling [18] by aggregating individual frames descriptors along the temporal axis and resulting in compact and robust descriptors regardless of the input sequence length;
- We show that, compared to previous SOTA, our pipeline achieves better results and faster inference thanks to its reliance on smaller dimensional descriptors.

II. Related Works

Sequence matching: Sequence matching, or frame-by-frame matching, represents an established approach to seq2seq [19], [20], and it operates by building a similarity matrix wherein descriptors of single query frames are compared to database ones. The best match is then determined by aggregating the scores in the matrix under simplifying assumptions, such as constant velocity or no stops [21], which makes it hard to generalize to real-world applications. There is a rich literature on sequence matching that tries to relax these assumptions by exploiting ego-motion information or using complex methods [22], [22] and graph-based frameworks [23], [24], [25]. Recently, SeqMatchNet [26] has also addressed the fact that these methods rely on learned image-to-image descriptors trained without considering the downstream procedure of score aggregation. Despite these improvements, sequence matching can generally be expensive to perform, as it requires each frame from the query to be matched to each frame of all databases sequences, as discussed in [11], [12].

Sequence descriptors: Sequence descriptor methods summarize each sequence with a single embedding which can be used for retrieving the most similar matches. This allows to incorporate temporal clues directly into the descriptors and to perform the similarity search directly on sequences rather than frames, thus greatly reducing the matching time. Facil et al. [9] first introduced the idea of sequence descriptors in VPR using simple aggregation techniques such as concatenation, sum, or processing via a LSTM network. [8] extended their benchmark on the Mapillary Street Level Sequences (MSLS) dataset. A non-learnable aggregation via discrete convolution was explored by Garg et al. [10]. Alternatively, 1D temporal convolutions were employed in SeqNet [11] to obtain a learnable aggregation of frame descriptors. Recently, [27] demonstrated a hyperdimensional computing approach to systematically combine information from multiple single-image descriptors. Considering the architectural differences among these methods, [12] provides a benchmark and taxonomy for seq2seq methods depending on how the frame-level features are fused together, and then it introduces the SeqVLAD aggregation layer that achieves SOTA performance. A follow up work is found in [28].

Image-to-image place recognition on large databases: There is a parallel body of literature in computer vision on image-to-image place recognition, addressing it as a retrieval task using global image descriptors. For years, the de-facto standard method has been NetVLAD [29], that also introduced the training procedure with mining and triplet loss. However, recently [15] has pointed out that the cost of mining triplets is a major bottleneck that prevents these methods from scaling to large datasets. This consideration inspired few recent papers to pursue mining-free methods in order to enable training on massive datasets. Firstly, CosPlace [13] provides a method to split large dense datasets into non-overlapping classes, which then allows for training to be performed with scalable loss functions. Using a different approach, Ali-Bey et al. [30] provides a dataset that is already split into well-defined classes, allowing to use standard retrieval losses without the need for mining. MixVPR [14] uses a similar training approach, and shows that well-designed architectures can provide a boost in recall. Most recently, [31] introduces a novel reward function, named Generalized Contrastive Loss, to dispense from hard-pair mining.

This trend in the literature shows that a method that is able to efficiently leverage large scale datasets can bring great benefits for performances. In seq2seq VPR this has not been possible because it is hard to obtain such large datasets. In this letter we propose a training protocol for sequence descriptors that is able to leverage the large amount of data readily available for the im2im task, even though it does not contain sequences of frames. Moreover, we show that our approach is able to improve upon previous SOTA while reducing the cost of deployment.

III. Method

A. Problem Setting

We tackle the task of *seq2seq VPR* that is formally defined in [8]: given a query sequence the system has to output a sequence from the available database that matches the former. Since the database sequences have GPS labels, this allows to infer an estimate of the query's position. A match is deemed correct if any of the retrieved frames is within 25 meters [8] from any of the query frames. The common recall@N metric [11], [12], [15], [20] is used as an aggregate evaluation, and it represents the percentage of queries that have at least one correct match in the top-N candidates.

Our method builds on the idea that the task of seq2seq VPR can be split into 2 learning objectives: (i) learn to extract features that are distinctive for localization (i.e. ignore transient objects, focus on static components, their style and relative position) and (ii) model the temporal evolution of these salient features

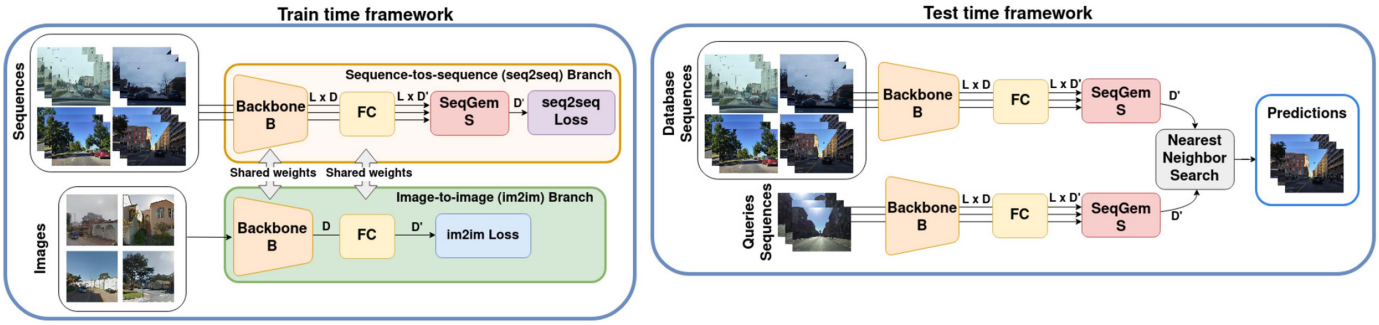

Fig. 2. **Overview of the JIST framework.** At training time (left) we use two branches, one for sequences and one for single-images. Each branch has a separate loss, while sharing part of their weights. The multi-task training allows to obtain discriminative frame-wise embeddings by exploiting the powerful representations learned by the backbone and fully connected from single images. At test time (right) we only use the sequences branch, and we follow the standard image retrieval pipeline: embeddings are extracted for both database and queries sequences, and then a prediction for database sequence that is most similar to the query is computed through a kNN. Note that in a real-world scenario, the potentially expensive embeddings extraction for database sequences can be performed offline, making the framework fast (more information on efficiency in Section IV-C).

within a sequence. In the spirit of deep learning, it is possible to jointly acquire both capabilities in an end-to-end fashion from a dataset of sequences [12]. However, we observe that for the first objective we do not necessarily need sequences, but we can exploit existing large-scale non-sequential VPR datasets to embed into our model robustness to a large variety of scenarios. Following this intuition, we devise our multi-task learning framework.

B. Multi Task Framework: Overview

The typical descriptor extractor architecture for im2im retrieval is composed by a backbone and an aggregator of feature maps (or tokens). For retrieval on sequences, there is an additional step to aggregate frame-level information [12]. In order to leverage both im2im and seq2seq datasets, we need a unified architecture with a frame-level aggregation layer able to process both individual images and sequences. Thus, we propose a novel double-branched architecture: one branch takes sequential data as input, while the other takes single images (see Fig. 2). We iteratively feed each branch with one batch of its corresponding input, compute their respective losses, backpropagate through the entire model and sum the gradients computed for each loss, which are then used for optimization. In doing so, we ensure that both branches share the same gradients and weights: in practice this makes the backbone and fully connected layer (FC) of the two branches identical, and allows joint optimization on both losses in a Siamese-like fashion. Following, we explain how the two branches work at inference and training time.

C. Sequence-to-Sequence Branch

The sequence-to-sequence branch has the objective of exploiting all the frame-wise information extracted by the im2im branch, via the shared backbone and FC layer, while learning to aggregate temporal information from sequences into compact descriptors. The input to this branch is formed by sequences x_{seq} of frames, with $x_{seq} \in \mathbb{R}^{L \times H \times W \times C}$ (where L is the sequence length). The sequences are passed to a backbone B, which extracts $L \times D$ dimensional features where the D depends on the backbone. These features are then passed through an FC layer F, which acts as a whitening transformation [18] and produces L D'-dimensional descriptors (i.e. one descriptor per frame), where D' can be set to a chosen output dimension. At this point, we need two more ingredients: firstly a *sequence aggregation* module that combines these frames into a single vector (the sequence descriptor); secondly, a *loss function* for the seq2seq task.

Sequence aggregation: SeqGeM: The current SOTA sequence descriptor from [12] is built using the SeqVLAD aggregator. This module reinterprets the classic NetVLAD module [29] to make it suitable for sequences. In a nutshell, given a set of D'-dimensional input descriptors SeqVLAD produces a single sequence descriptor vector of size $K \cdot D$, where K is a parameter indicating the number of clusters used to summarize the input vectors. In practice, the implementation from [12] uses $K = 64$, which significantly increases the size of the sequence descriptor and, as a consequence, the matching time of the retrieval. To mitigate this problem, [12] uses a PCA compression operation, which nevertheless adds a post-processing computational overhead.

For all these reasons, we propose a new aggregation module called Sequential Generalized Mean (**SeqGeM**), which revisits the popular GeM layer [18] to operate on sequences, by applying its pooling operation along the temporal axis given a sequence of single-image embeddings (see Fig. 3). Formally, the SeqGeM layer is defined as

$$S : \mathbb{R}^{L \times D'} \rightarrow \mathbb{R}^{D'}$$

$$[d_0, \dots d_{L-1}] \mapsto \left(\frac{1}{L} \sum_{i=0}^{L-1} d_i^p \right)^{\frac{1}{p}} \tag{1}$$

where p is a learnable parameter and d_i is the descriptor of the i^{th} frame. Therefore, the sequence descriptor extraction process is

$$f_{seq} = S(F(B(x_{seq}))) \tag{2}$$

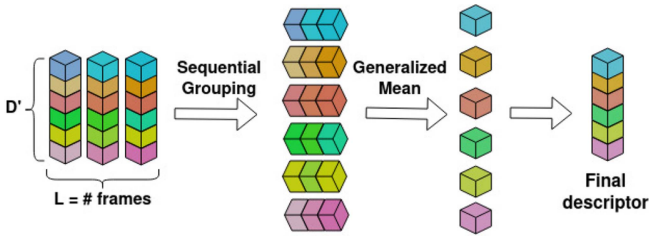

Fig. 3. Sketch of our proposed SeqGeM layer. Given D-dimensional feature vectors from L frames, SeqGeM produces a single descriptor/embedding of dimensionality D, which contains information from the whole sequence.

SeqGeM is implemented with differentiable operations, and it has a few desirable properties: i) it natively produces low-dimensional descriptors without requiring a PCA compression; ii) it is learnable; iii) it has few parameters; iv) it is flexible w.r.t. the length of input sequences, so that sequences of different length can be compared to each other. Finally, SeqGeM is purposefully designed to aggregate only the final descriptors of each frame, instead of the frame's feature maps, as in this way it is able to (i) take advantage of the entire im2im branch, which is trained on large amount of images, and (ii) take as input small descriptors and produce small outputs, whereas usually methods that take as inputs the feature maps (e.g. SeqVLAD) produce large-dimensional sequence descriptors which increases memory and time requirements.

Seq2seq loss: Following best practices from the literature, we use the popular weakly supervised margin triplet loss [29], which takes a query, its positive (a sequence from the same place), and a negative. For best results, negatives need to be *mined*, because selecting random negatives would lead to trivial triplets (i.e., with loss 0), by selecting the negatives closest to the queries in features space. Given triplets of query, positive and negative, the weakly supervised triplet loss, used to train the seq2seq branch, is defined as:

$$\mathcal{L}_{seq2seq} = \sum_i \max(0, d(f^{q_i}_{seq}, f^{p_i}_{seq}) - d(f^{q_i}_{seq}, f^{n_i}_{seq}) + m)) \tag{3}$$

where $f^q_{seq}, f^p_{seq}, f^n_{seq}$ represent the features of a query, its positive and negative, m is the margin of the triplet loss, and $d(\cdot)$ is the euclidean distance between two features.

D. Image-to-Image Branch

The second branch processes single images instead of sequences: given input images $x_{im} \in \mathbb{R}^{H \times W \times C}$ the image branch produces D' dimensional local feature descriptors which can be fed to the image loss. The local feature descriptors are computed as

$$f_{im} = F(B(x_{im})) \tag{4}$$

where the backbone B and fully connected layer F are shared with sequence-to-sequence branch (see Fig. 2). Finally, we attach a loss $\mathcal{L}_{im2im}$ for the image-to-image task, that back-propagates through B and F.

Im2im loss: Since our goal for this branch is to exploit huge datasets of single images to learn robust representations, we resort to the CosPlace training protocol and loss [13] that is the current state-of-the-art for large scale im2im VPR and was designed to be used on the massive San Francisco eXtra Large (SF-XL) dataset. Below we provide a summarized explanation of the CosPlace training protocol, although we note that this is not meant to be a thorough description and we refer the reader to the original CosPlace paper [13] for a more detailed explanation.

The CosPlace training protocol is divided in two steps. In the first step, the SF-XL dataset which contains images labeled with UTM coordinates and heading angles is partitioned into classes based on their position and orientation. This process, that is performed once prior to the actual training, divides the geographical area into small squared cells (10×10 meters) and splits each cell into 12 classes along the orientation/heading (i.e. each class is 30° wide), thus ensuring that all the images in a single class view the same scene (by having similar position and orientation). This division of the continuous label space in a finite number of classes enables the usage of highly scalable losses for large-scale image retrieval, such as the CosFace loss [32].

Therefore, the second step consists in training the model using the CosFace loss on the obtained classes. However, naively using all the classes would be problematic, because images in two adjacent classes may have a very high visual overlap, thus potentially containing the same scene seen from slightly different points of view. Since this would lead to unstable gradients during optimization, the training protocol only considers images from a subset of classes chosen so that no two adjacent classes are used at the same time. This subset is not fixed, but it is changed iteratively during training, to allow the model to see all the images in the dataset.

Summarizing, in this letter we denote as $\mathcal{L}_{im2im}$ the CosFace loss applied according to the CosPlace protocol. However, we want to remark that in principle our multi-task framework is loss-agnostic, so the im2im loss can be easily swapped with another one, for example if a more performing loss becomes available.

E. Total Multi-Task Loss

Overall, the total loss of our multi-task framework is

$$\mathcal{L}_{multi-task} = \lambda_{seq2seq}\mathcal{L}_{seq2seq} + \lambda_{im2im}\mathcal{L}_{im2im} \tag{5}$$

where $\lambda_{seq2seq}$ and λ_{im2im} are hyperparameters. The combination of this multi-task loss with architecture that includes the SeqGeM aggregation makes our multi-task framework, which we name Joint Image and Sequence Training, or JIST.

IV. Experiments

A. Experimental Setup

Datasets: To assess the soundness of the JIST multi-task training framework, we use the following datasets:

• *Mapillary Street-Level Sequences* (MSLS) [8], is built from various cities around world, split in non-overlapping training, validation and test sets, and consisting of 393 k query sequences and 733 k for the database (if we consider 5-frames sequences).

As the original test set labels are not released by the authors, we follow the splits defined in [12].

- *Test set:* Copenhagen, San Francisco
- *Val set:* Amsterdam, Manila
- *Train set:* all remaining cities

Unless otherwise specified, experiments (train/val/test) are performed on MSLS.

- *MSLS Melbourne:* is the subset of MSLS from the city of Melbourne, and it is commonly used [11], [12], [26] to understand the effect of training only on a single city as opposed to the entire MSLS train set. When the model is trained on Melbourne, the validation and testing are performed on the standard MSLS val and test sets.
- *San Francisco eXtra Large* (SF-XL) [13] is a large-scale (41 M images) im2im dataset covering the whole city of San Francisco, and it is used as a training set for the CosPlace component of the loss. Note that CosPlace requires camera heading labels, meaning that most other datasets (MSLS included) can not be used for training CosPlace.
- *Oxford RobotCar* [35] is a small dataset containing roughly 4 k queries and database sequences in each split. It contains multiple traversals of the same path around the city of Oxford. Laps are recorded in different times of the day, year, as well as changing weather conditions, targeting robustness to domain shifts. In the literature there is little consistency upon which splits to adopt [10], [11], [26], thus as with MSLS we follow the proposed one in [12].
  - *RobotCar Test set:* queries: 2014-12-16-18-44-24 (winter night); database: 2014-11-18-13-20-12 (fall day).
  - *RobotCar Validation set:* queries: 2015-02-03-08-45-10 (winter day, snow); database: 2015-11-13-10-28-08 (fall day, overcast).
  - *RobotCar Train set:* queries: 2014-12-17-18-18-43 (winter night, rain); database: 2014-12-16-09-14-09 (winter day, sun).

Training: For training, we set $\lambda_{im2im} = 100$ and $\lambda_{seq2seq} = 10.000$. The learning rate is set to 0.00001 and we use Adam [36] as optimizer. We train our model for a fixed number of iterations, namely 12.5 k. To speed up convergence and reduce carbon footprint of our trainings, we initialize the backbone with the open-source pretrained weights from CosPlace. Regarding our architecture, we use a ResNet-18 [37] backbone which has an output dimensionality $D = 512$. We keep the same dimension after the linear projection $D' = 512$, except for experiments in Table III where we show that our method works well also with smaller descriptors. The parameter p of SeqGeM is initialized to 3.

Evaluation: We use a standard kNN to find the predictions for each query. As metric, we use the Recall@N, defined as the number of queries that have at least one correct positives within the first N predictions. A prediction is deemed correct if at least one of its frames is less than 25 meters away from at least one the query's frames, following [8]'s definition of seq2seq. Unless otherwise specified we use a sequence length of 5 following previous work [12], although in Fig. 5 we show that SeqGeM is able to produce robust descriptors even with different sequence lengths. Given that in VPR it is logical to either train and test on different (non-overlapping) geographical areas [29], or to consider the train and test sets to be geographically overlapping [13], we compute results for both cases: results on MSLS use geographically disjointed sets, whereas results on RobotCar use the same area for training and testing.

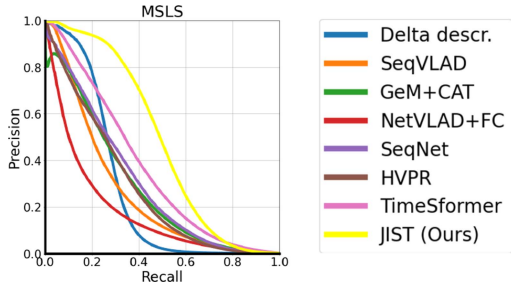

Fig. 4. Precision-Recall curves computed on MSLS test set for the most relevant methods. All models are trained with a ResNet-18 backbone except TimeSformer, which uses a custom backbone.

Methods: We report results from a large number of methods on the task of seq2seq VPR. Wherever available, we made use of the authors official code for our comparisons. For methods based on the traditional *sequence matching*, we compare against three popular implementations: SeqSLAM [20], HVPR [11], and SeqMatchNet [26]. We also compare to existing methods based on *sequence descriptors*. Starting from the work of [8], we test standard concatenation (CAT) of popular im2im descriptors NetVLAD [29] and GeM [18] using different backbones. We also compute results with Delta Decriptors [10], a non-learnt pooling in this category. We compare against Fully-Connected layers on top of flattened frame descriptors [9], varying the feature extractor. Additionally, we test the learnable pooling of SeqNet [11] and the previous SOTA represented by SeqVLAD [12]. Finally, we test a method that processes all frames as a single entity from the first layers, namely the TimeSformer [34].

For methods that produce huge descriptors, mostly due to NetVLAD applied on each frame of the sequences, we followed [12] and applied PCA for dimensionality reduction. It is noteworthy in this sense that our proposed pipeline naturally outputs compact descriptors (512-D) freeing ourselves from the extra cost of applying PCA, while also achieving higher results despite the lower dimensionality.

A few methods (HVPR, SeqMatchNet and SeqNet) could not be trained on the whole MSLS due to large memory requirements of their implementation (more than 256 GB of RAM), hence why some results are missing. Finally, we clarify that official code for Delta Descriptors and SeqSLAM do not train frame-level descriptors and rely on pre-trained networks. In the table they are highlighted with *.

B. Results and Discussion

To empirically assess the effectiveness of our proposed models against previous literature, we report a wide set of experiments in Table I, and precision-recall curves for the most relevant methods in Fig. 4.

TABLE I
EVALUATION OF SEQUENTIAL DESCRIPTORS AND SEQUENCE MATCHING, ON SEQUENCES OF LENGTH 5: RECALL@1 ON VARIOUS DATASETS

Method	Backbone	Descriptor Dimension	GPU Memory Occupation (GB)	Extraction Time (ms)	Matching Time (ms)	Train on Melbourne Test on MSLS	Train on MSLS Test on MSLS	Train on RobotCar Test on RobotCar	Train on MSLS Test on RobotCar
SeqSLAM* [20]	VGG-16	4096 · SL	2.68	39.8	500.1	45.9	45.9	34.7	34.7
HVPR [11]	VGG-16	4096 · SL	2.68	39.8	0.9	51.0	-	56.8	-
SeqMatchNet[27]	VGG-16	4096 · SL	2.68	39.8	500.1	44.8	-	51.9	-
Delta Descriptors*[10]	VGG-16	4096 · SL	2.68	39.8	1.1	43.0	43.0	18.0	18.0
GeM + CAT [8]	ResNet-18	256 · SL	2.04	10.6	0.3	66.7	76.8	75.4	26.4
GeM + CAT [8]	ResNet-50	1024 · SL	2.25	29.5	1.8	63.4	68.6	81.3	14.1
NetVLAD + CAT + PCA [9]	ResNet-18	4096	2.04	10.4	1.1	75.5	83.7	67.8	47.0
NetVLAD + CAT + PCA [9]	ResNet-50	4096	2.26	30.4	1.1	74.9	85.3	89.3	62.4
SeqPool + CAT [34]	CCT384	384 · SL	2.01	15.6	0.5	69.9	77.8	77.4	42.6
SeqNet [11]	VGG-16	4096	2.68	39.8	1.1	50.1	-	60.5	-
SeqNet [11]	ResNet-50	4096	2.26	31.0	1.1	45.6	-	61.3	-
NetVLAD + FC [9]	ResNet-18	4096	3.39	10.5	1.1	55.5	68.5	44.7	19.3
SeqVLAD + PCA [12]	ResNet-18	4096	2.04	10.5	1.1	78.2	85.5	86.5	60.9
SeqVLAD [12]	CCT384	24576	2.02	17.2	6.4	<u>81.7</u>	<u>89.4</u>	<u>92.8</u>	78.7
SeqVLAD + PCA [12]	CCT384	4096	2.02	17.2	1.1	81.4	89.2	**93.3**	76.8
TimeSformer [35]	-	768	2.34	13.5	0.2	73.8	81.5	74.9	46.8
JIST (Ours)	ResNet-18	512	2.04	11.1	0.1	**88.9**	**90.6**	91.5	**79.0**

SL stands for sequence length, CAT indicates concatenation of descriptors, FC stands for fully connected layer. Extraction time is the time to extract descriptors/embeddings, and matching time is the time to find the predictions given the descriptors given the test database of MSLS (with 13584 sequences). Both times refer to a single query. *Denotes a non-trained method.
The best results in bold, second best are underlined.

TABLE II
ABLATION ON THE TWO COMPONENTS OF THE MULTI-TASK LOSS, ON MSLS

$\lambda_{seq2seq}$	0	100	1000	10k	100k	10k	10k	10k	10k	10k
λ_{im2im}	100	100	100	100	100	0	1	10	100	1000
R@1	87.6	88.4	89.4	**90.6**	90.4	89.9	90.2	90.2	**90.6**	89.9

The best results in bold.

TABLE III
EFFECT OF DESCRIPTOR DIMENSIONALITY: JIST VS. SEQVLAD

Descriptor Dim.	64	128	256	512	4096
JIST(ours)	<u>83.6</u>	<u>87.9</u>	<u>89.4</u>	**90.6**	-
SeqVLAD + PCA	83.4	85.6	86.7	87.7	89.2

The best result overall in bold, best per descriptor dimension is underlined.

We summarize the findings from experiments as follows:

- JIST achieves excellent results with small-dimensional descriptors, even when trained on fewer sequential data (i.e. training on Melbourne);
- SeqVLAD achieves overall good results, but its recalls are poor when trained on fewer data;
- Despite its strong results, JIST is extremely fast and uses a simple model for inference;
- Extraction time depends mostly on the backbone, and only slightly depend on the aggregation layer (*e.g.* CAT, SeqGeM, FC);
- On all considered testing datasets, extraction is the bottleneck, although for a bigger dataset matching would be slower, as its speed linearly depends on dataset size;
- We empirically verified that matching time is linearly correlated to descriptors dimension for sequence descriptors (i.e. pure retrieval) methods;

Computational cost: Besides being fast to train (less than 10 hours on a single GPU), JIST provides very efficient inference, due to small descriptors and lightweight architecture. Specifically, we rely on a ResNet-18, which has only 11 M parameters, leading to fast features extraction time.

Matching time is also small (8 times smaller than previous SOTA), due to SeqGeM's compact output: in fact the matching time (i.e. time it takes to find the matching descriptors to the query's through a kNN) depends only on the descriptors' dimension and the size of the database. Note that, as we scale to larger datasets (with more sequences in the database), the bottleneck of a VPR system at inference shifts from the extraction to matching, making compact descriptors and fast matching an important characteristic for large-scale deployment [15].

Ablation on the loss: In this paragraph we aim at understanding how each component of the loss affects results, to justify their use in training. In Table II we report results computed with different weights for $\lambda_{seq2seq}$ and λ_{im2im}, with a ResNet-18 and our proposed SeqGeM layer. We find that when any of the two has a null effect on the back-propagated gradients, the results are evidently lower, proving that both learning objectives are beneficial to the task. Note that using $\lambda_{seq2seq} = 0$ means that only the im2im loss is used (therefore SeqGeM is not trained, but simply initialized to 3). The best results are shown with values of $\lambda_{seq2seq} = 10.000$ and $\lambda_{im2im} = 100$. Finally, we note that the $\mathcal{L}_{seq2seq}$ has a stronger effect than the λ_{im2im}, as not using the $\mathcal{L}_{seq2seq}$ leads to a 3% points in reduction with respect to the best model. This effect proves the fact that while it is possible to learn to extract salient features for localization using only single images, a loss that instructs the model how to aggregate temporal information is necessary.

Effect of descriptor dimensionality: Due to its importance for seq2seq VPR, and for retrieval in general, we perform an ablation on the dimensionality of descriptors. We find that JIST allows trained models to perform astonishingly well even at very low dimensions, reaching an impressive 87.9% with 128 dimensional descriptors, which is only 1.5% lower than previous state of the art while being 192 times smaller. This practically means that the 128D SeqGeM configuration requires only 512 bytes to store each sequence, allowing to store entire cities within a single embedded device. More considerations on this topic of real-world applicability are in Section IV-C.

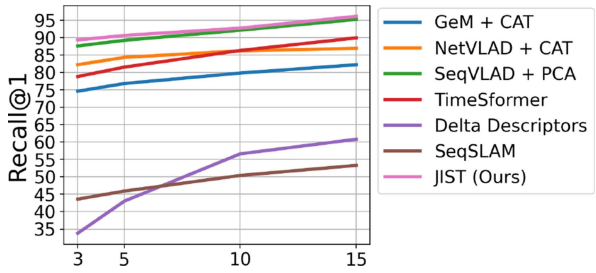

Fig. 5. Plot shows how different methods react to changes in the dimension of test-time sequence length (i.e. number of frames). All methods are trained with fixed sequence length of five.

TABLE IV
ROBUSTNESS TO THE INVERSION OF THE FRAMES

Method	Backbone	Dim.	Forw.	Back.	Diff
SeqSLAM* [20]	VGG-16	4096	45.9	22.9	-50 %
Delta D.*[10]	VGG-16	4096	43.0	11.7	-73 %
HVPR [11]	VGG-16	4096	51.0	28.5	-44 %
SeqMatchNet [27]	VGG-16	4096	44.8	24.2	-46 %
GeM + CAT [8]	ResNet-18	1280	76.8	67.8	-12 %
NetVLAD + CAT + PCA	ResNet-18	4096	83.7	79.9	-5 %
SeqNet [11]	VGG-16	4096	50.1	42.0	-16 %
NetVLAD + FC	ResNet-18	4096	68.5	65.1	-5 %
SeqVLAD + PCA	ResNet-18	4096	85.5	85.2	-0.4 %
SeqVLAD + PCA	CCT384	4096	89.2	89.2	**0.0** %
TimeSformer	-	768	81.5	81.5	**0.0** %
JIST (Ours)	ResNet-18	512	90.6	90.6	**0.0** %

The best (lowest) differences when inverting frames in bold.

Effect of sequence length: In Fig. 5 we investigate the effect of changing the number of frames within sequences (sequence length) at test time, without re-training the model, noting that flexibility on processing sequences of arbitrary lengths (regardless of the length at training time) is a desirable property in practical applications. Firstly, we note that only a small number of methods can be applied to this scenario, i.e. those based on CAT, SeqVLAD, SeqGeM and TimeSformer: others, like those based on a fully connected layer, would need to be trained again from scratch, as their number of parameters depends on the sequence length.

Clearly, all methods benefit from longer sequences: with more frames, descriptors become more informative, limiting perceptual aliasing. Models trained with JIST outperform all competitors, especially with very short sequences: this is expected behaviour, as the image loss allows to extract informative features even from a single frame.

Effect of reversing frames: Robustness to frame ordering is a desirable property in some realistic use-cases, because it allows to reduce the number of sequences stored in the database. Following [9], [11], [12] we assess each model's robustness to reversing the frame ordering for queries sequences, while keeping the database untouched, and report results in Table IV. SeqGeM is inherently robust to frame-ordering, as well as SeqVLAD and TimeSformer which processes the sequence in its entirety. On the other hand, methods based on FC-layers, CAT or sequence matching are the ones that suffer most in this scenario.

Ablation on aggregation layer: Given their importance in aggregating features from multiple frames, in Table V we report experiments performed with a number of pooling/aggregation layers. This shows a number of desirable properties that are satisfied by SeqGem, as well as showing its superiority of results. In particular, we note that the SeqGeM aggregator provides the following characteristics: (i) learnable, (ii) flexible w.r.t. length of input sequences, (iii) invariant to frame ordering, (iv) lightweight, besides producing compact output and having few parameters.

TABLE V
COMPARISON OF AGGREGATION LAYERS. RECALL@1 IS COMPUTED WITH SAME TRAINING CONFIGURATION ON MSLS SPLITS

Aggregation	Learnable	Flexible w.r.t. length input seq.	Invariant to frame order	Output Dimensionality	R@1
Max Pooling	N	Y	Y	512	89.4
Avg Pooling	N	Y	Y	512	89.5
1D-conv	Y	N	N	512	88.7
SeqVLAD	Y	Y	Y	32768	89.7
SeqGeM	Y	Y	Y	512	90.6

Note that the results from Table V are performed within the JIST framework/pipeline, making these aggregations achieve superior recalls w.r.t. most of the baselines from Table I.

C. Considerations for Real-World Deployment

As the use of deep models for seq2seq VPR becomes widespread, we investigate the feasibility of deploying such models in the real world. We perform experiments on a Jetson Nano platform. Considering the scenario of a large city like San Francisco, with 1600 kilometers of road, it would require roughly 800 k sequences to map the whole city. Using the previous state-of-the-art model, namely CCT384 [33] with SeqVLAD [12], it needs ≈ 36 GB (*#sequences * descriptors dimension * #bytes*) of memory to store all the descriptors. More compact representations (commonly compressed with PCA) usually rely on 4096-D features [12], at the cost of a performance penalty. With SeqGeM however, we are able to outperform previous state of the art with 512-D descriptors, which needs only $800\,\text{k} * 512 * 4\text{B} \approx 0.75$ GB, and can be handled by a Jetson Nano.

Given this setting, we analyzed the inference time on a Jetson Nano: we found that extraction time for a sequence takes 276 ms (i.e. with our ResNet-18; does not depend on the size of the database). Matching takes 3.1 seconds with a vanilla kNN (on the whole city of San Francisco). We note that previous works on im2im VPR found that kNN can be sped up by up to 64 times with negligible loss of recall [15] when using approximate/efficient versions of it, like Inverted File Index with Product Quantization [38], [39], leading to a potential processing speed of roughly 3 sequences per second ($276\,\text{ms} + (3100/64)\,\text{ms} = 324\,\text{ms}$), whereas previous SOTA (with descriptors dimension 24576) would process only 0.4 sequences per second. Even with PCA, the throughput would still be limited to 1.4 sequences per second.

V. CONCLUSION

This work proposes a novel training algorithm that efficiently exploits existing data sources to boost performance in sequence-based VPR. We introduce a trainable temporal aggregation layer designed to being flexible to input length and frame ordering, all while guaranteeing compact descriptors. Through extensive experimental evaluation we showcase the improvements that

JIST achieves over previous SOTA, as well as robustness to different conditions such as changes in frame ordering, sequence length and different datasets. We empirically demonstrate that our model is able to not only achieve better results, but also be faster and lighter (in terms of RAM and GPU memory).

Limitations: Although sequence descriptors are a competitive solution to obtain efficiently a coarse global localization estimate even in very large environments, their use is intended when there is the need to search in a large number of sequences (e.g., for loop closure or to bootstrap the localization when lost). Furthermore, we note that a limitation of the current JIST framework is that the two losses require different format of datasets, where the im2im branch is trained on large-scale single-image datasets whereas the seq2seq branch requires continual sequences.

Future works: Possible directions for follow-up works may explore different strategies for extracting knowledge from large pre-trained models (e.g. distillation), generalizing our multi-task framework to other tasks, or using more than two branches to gather knowledge from other data sources.

REFERENCES

[1] C. Masone and B. Caputo, "A survey on deep visual place recognition," *IEEE Access*, vol. 9, pp. 19516–19547, 2021.

[2] K. A. Tsintotas, L. Bampis, and A. Gasteratos, "The revisiting problem in simultaneous localization and mapping: A survey on visual loop closure detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 19929–19953, Nov. 2022.

[3] A. Angeli, S. Doncieux, J.-A. Meyer, and D. Filliat, "Real-time visual loop-closure detection," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2008, pp. 1842–1847.

[4] N. Pion, M. Humenberger, G. Csurka, Y. Cabon, and T. Sattler, "Benchmarking image retrieval for visual localization," in *Proc. Int. Conf. 3D Vis.*, 2020, pp. 483–494.

[5] D. Yudin, Y. Solomentsev, R. Musaev, A. Staroverov, and A. I. Panov, "HPointLoc: Point-based indoor place recognition using synthetic RGB-D images," in *Proc. Int. Conf. Neural Inf. Process.*, 2023, pp. 471–484.

[6] F. Zeng, A. Jacobson, D. W. Smith, N. Boswell, T. Peynot, and M. Milford, "Enhancing underground visual place recognition with Shannon entropy saliency," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 1–10.

[7] C. Toft et al., "Long-term visual localization revisited," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 4, pp. 2074–2088, Apr. 2022.

[8] F. Warburg, S. Hauberg, M. Lopez-Antequera, P. Gargallo, Y. Kuang, and J. Civera, "Mapillary street-level sequences: A dataset for lifelong place recognition," in *Proc. Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 2623–2632.

[9] J. M. Fácil, D. Olid, L. Montesano, and J. Civera, "Condition-invariant multi-view place recognition," *ArXiv*, vol. abs/1902.09516, 2019.

[10] S. Garg, B. Harwood, G. Anand, and M. Milford, "Delta descriptors: Change-based place representation for robust visual localization," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 5120–5127, Oct. 2020.

[11] S. Garg and M. Milford, "SeqNet: Learning descriptors for sequence-based hierarchical place recognition," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 4305–4312, Jul. 2021.

[12] R. Mereu, G. Trivigno, G. Berton, C. Masone, and B. Caputo, "Learning sequential descriptors for sequence-based visual place recognition," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 10383–10390, Oct. 2022.

[13] G. Berton, C. Masone, and B. Caputo, "Rethinking visual geo-localization for large-scale applications," in *Proc. Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 4868–4878.

[14] A. Ali-bey, B. Chaib-draa, and P. Giguère, "MixVPR: Feature mixing for visual place recognition," in *Proc. IEEE Wint. Conf. Appl. Comput. Vis.*, 2023, pp. 2998–3007.

[15] G. Berton et al., "Deep visual geo-localization benchmark," in *Proc. Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 5386–5397.

[16] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *Proc. Conf. Comput. Vis. Pattern Recog.*, 2007, pp. 1–8.

[17] J. Martinez et al., "Pit30m: A benchmark for global localization in the age of self-driving cars," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 4477–4484.

[18] F. Radenović, G. Tolias, and O. Chum, "Fine-tuning CNN image retrieval with no human annotation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 7, pp. 1655–1668, Jul. 2019.

[19] K. L. Ho and P. Newman, "Detecting loop closure with scene sequences," *Int. J. Comput. Vis.*, vol. 74, no. 3, pp. 261–286, 2007.

[20] M. J. Milford and G. F. Wyeth, "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 1643–1649.

[21] S. Schubert and P. Neubert, "What makes visual place recognition easy or hard?," *ArXiv*, vol. abs/2106.12671, 2021.

[22] T. Naseer, W. Burgard, and C. Stachniss, "Robust visual localization across seasons," *IEEE Trans. Robot.*, vol. 34, no. 2, pp. 289–302, Apr. 2018.

[23] O. Vysotska and C. Stachniss, "Lazy data association for image sequences matching under substantial appearance changes," *IEEE Robot. Autom. Lett.*, vol. 1, no. 1, pp. 213–220, Jan. 2016.

[24] S. Schubert, P. Neubert, and P. Protzel, "Graph-based non-linear least squares optimization for visual place recognition in changing environments," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 811–818, Apr. 2021.

[25] S. Schubert, P. Neubert, and Protzel, "Fast and memory efficient graph optimization via ICM for visual place recognition," *Robot. Sci. Syst.*, vol. 73, pp. 1842–1847, 2021.

[26] S. Garg, M. Vankadari, and M. Milford, "SeqMatchNet: Contrastive learning with sequence matching for place recognition & relocalization," in *Proc. Mach. Learn. Res.*, vol. 2022, pp. 429–443.

[27] P. Neubert and S. Schubert, "Hyperdimensional computing as a framework for systematic aggregation of image descriptors," in *Proc. Conf. Comput. Vis. Pattern Recog.*, 2021, pp. 16933–16942.

[28] F. Zhang, J. Zhao, Y. Cai, G. Tian, W. Mu, and C. Ye, "Learning sequence descriptor based on spatiotemporal attention for visual place recognition," Jul. 2023, *arXiv:2305.11467*.

[29] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1437–1451, 2018.

[30] A. Ali-bey, B. Chaib-draa, and P. Giguère, "GSV-cities: Toward appropriate supervised visual place recognition," *Neurocomputing*, vol. 513, pp. 194–203, 2022.

[31] M. Leyva-Vallina, N. Strisciuglio, and N. Petkov, "Data-efficient large scale place recognition with graded similarity supervision," in *Proc. Conf. Comput. Vis. Pattern Recog.*, 2023, pp. 23 487–23 496.

[32] H. Wang et al., "Cosface: Large margin cosine loss for deep face recognition," in *Proc. Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 5265–5274.

[33] A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li, and H. Shi, "Escaping the Big Data paradigm with compact transformers," *CoRR*, vol. abs/2104.05704, 2021.

[34] G. Bertasius, H. Wang, and L. Torresani, "Is space-time attention all you need for video understanding?," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 813–824.

[35] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The Oxford RobotCar dataset," *Int. J. Robot. Res.*, vol. 36, no. 1, pp. 3–15, 2017.

[36] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2014.

[37] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 770–778.

[38] H. Jégou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 117–128, Jan. 2011.

[39] A. Babenko and V. S. Lempitsky, "The inverted multi-index," in *Proc. Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 3069–3076.

Open Access funding provided by 'Politecnico di Torino' within the CRUI CARE Agreement