Robotics: Science and Systems 2021
Held Virtually, July 12–16, 2021

Vector Semantic Representations as Descriptors for Visual Place Recognition

Peer Neubert, Stefan Schubert, Kenny Schlegel and Peter Protzel
Chemnitz University of Technology, Germany
{peer.neubert, stefan.schubert, kenny.schlegel, peter.protzel}@etit.tu-chemnitz.de

***Abstract*—Place recognition is the task of recognizing the current scene from a database of known places. The currently dominant algorithmic paradigm is to use (deep learning based) holistic feature vectors to describe each place and use fast vector query methods to find matchings. We propose a novel type of image descriptor, Vector Semantic Representations (VSR), that encodes the spatial semantic layout from a semantic segmentation together with appearance properties in a, for example, 4,096 dimensional vector for place recognition. We leverage operations from the established class of Vector Symbolic Architectures to combine symbolic (e.g. class label) and numeric (e.g. feature map response) information in a common vector representation. We evaluate the proposed semantic descriptor on 13 standard mobile robotic place recognition datasets and compare to six descriptors from the literature. VSR is on par with the best compared descriptor (NetVLAD) in terms of mean average precision and superior in terms of recall and worst-case average precision. This makes the approach particularly interesting for candidate selection. For a more detailed investigation, we discuss and evaluate recall integrity as additional criterion. Further, we demonstrate that the semantic descriptor is particularly well suited for combination with existing appearance descriptors indicating that semantics provide complementary information for image matching.**

I. Introduction

Visual place recognition is the task of matching a given query image to a potentially large database of known places. It is an important means for loop closure detection in SLAM and for candidate selection for 6-D pose estimation [56]. This task becomes particularly challenging when the environmental condition changes due to changing illumination, weather, or season, and/or when the size of the datasbase becomes very large. Intuitively, information about the *semantic* content of the image can help in both directions. On one hand, semantic is largely invariant of appearance chances. A snow covered tree is still a tree. Here, recent and future developments from (deep) learned models to capture semantics can be leveraged. On the other hand, to address a large-scale database, one can use the semantic gist of a scene for a coarse categorization, e.g. into urban or rural scenes (think of the seminal GIST [48] paper). After such coarse categorizations, e.g. into an urban scene, one can conduct more fine grained semantic categorization using salient semantic landmarks (e.g the Eiffel tower) or other semantic features like the architectonic style of the buildings (think of the "What makes Paris look like Paris?" paper [13]). However, an largely open question is, how can we further exploit semantics together with fine-grained appearance properties for fast image matching, e.g. how to distinguish individual urban street scenes with a high proportion of similarly looking Victorian style buildings?

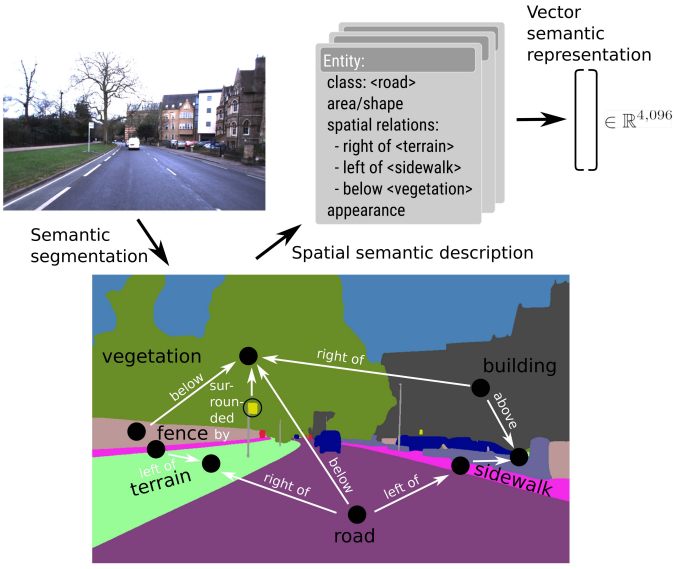

Fig. 1. A Vector Semantic Representation (VSR) is a single high-dimensional vector that combines information of semantic entities, their spatial layout, and appearance. We describe how a semantic segmentation can be used to create the entities and how operations from Vector Symbolic Architecture (VSA) can be used to encode this mixed symbolic-numeric information in a single vector that can serve as descriptor for place recognition.

In this paper we propose a novel approach to encode the spatial semantic layout of images for place recognition. An example is shown in Fig 1. The key idea is to describe the shown street scene by the semantic information that there is a sidewalk right to the street and grass terrain to the left, which in turn is followed by another sidewalk and a fence. We use a deep learning based semantic segmentation model to extract a list of semantic entities. This list includes objects with well defined shape and boundary ("things" [5], e.g. a sign), as well as amorphous background regions ("stuff" [5], e.g. terrain). Each entity is described by its semantic class, a coarse representation of its shape and location, and a list of its spatial semantic relations, e.g. "left-of <sidewalk>". This is complemented by an appearance descriptor based on salient feature map responses.

Given this list of entities, each with combined symbolic (e.g. class) and numeric (e.g. feature map response) information, the major challenge becomes to generate a descriptor that allows fast matching of these image representations. We propose

to leverage operations from the well established class of Vector Symbolic Architectures (VSA) [51, 19, 57] to generate a novel type of image descriptor named Vector Semantic Representation (VSR). Vector Symbolic Architectures allow the systematic processing and representation of symbolic and numeric information based on well defined operations on high-dimensional vectors (e.g. with 4,096 dimensions). We will describe how each piece of information from the above entity list is encoded in an individual high-dimensional vector and how we leverage the VSA principles to combine these vectors in a single 4,096 dimensional vector that captures all information from the spatial semantic description and that can serve as an image descriptor in the same fashion as, e.g., a NetVLAD [2] descriptor.

We will evaluate this novel holistic image descriptor on 13 sequence comparison from standard mobile robot place recognition datasets and compare against six descriptor approaches from the literature. The experiments show improved recall@k and average precision performance compared to the best compared approach (on par mean average precision, considerably improved worst-case performance). This makes the approach particularly promising for candidate selection, which is further investigated using a introduced recall integrity criterion. Based on the assumption that the encoded semantic information in the VSR descriptor is complementary to typically used appearance descriptors (e.g. NetVLAD [2]), we also evaluate combinations of different descriptors. The combination with VSR turns out to considerably improve all evaluated descriptors by a large margin and to be better suited than combinations of existing descriptors. Code is available.[1]

II. Related Work

A. Descriptors for place recognition

Visual place recognition [39] is an important task in mobile robotics and used for loop closure detection in SLAM or candidate selection for visual localization [56]. [60] discusses various aspects of the visual place recognition problem. Different to 6-DOF pose estimation that often uses local features (e.g. keypoints [38, 12, 47, 14]), place recognition typically builds upon holistic image descriptors that compute a single descriptor vector for a whole image [2, 72, 66, 42, 46]. Important reasons are the memory consumption and the required time for exhaustively comparing local features.

We will use operations from Vector Symbolic Architetctures to aggregate information of local entities in a holistic descriptor. There are several approaches available to create holistic descriptors from ordered or unordered sets of local features, including BoW [64, 10], Fisher vectors [50], and VLAD [23, 1]. Aggregated selective match kernels [71] aim at unifying aggregation-based techniques with matching-based approaches like Hamming Embedding [24]. VLAD in combination with soft-assignment is fully differentiable and seamlessly integrates in deep learning approaches, e.g. NetVLAD [2]. Other deep learning variants of local feature aggregation for image matching include sum-pooling [70], max-pooling [22], and mean-pooling [7]. The latter also outputs global and local descriptors.

The (relative) spatial location of local features can provide important information, e.g. for geometric verification [55]. Regarding holistic descriptors, BoW can integrate spatial information via voting [62]. Pyramid match kernels [21] can evaluate matchings at multiple resolutions. Based on this, spatial pyramid matching [36] can approximate global geometric correspondence between sets of local features. Multi-VLAD [1] extends this idea to VLAD, Pyramid-Enhanced NetVLAD [74] extends it to deep learning. The typical usage of flattened AlexNet-conv3 [35] descriptors (or similar) as in [66][61] is an implicit encoding of local features (i.e. feature map vectors) together with their image location (encoded by the position in the concatenated output vector). Similar encodings can be applied to other local features. In [44], we used hyperdimensional computing to encode DELF [47] descriptors together with their spatial location in a single holistic descriptor based on a MAP [18] architecture.

To reduce memory consumption and runtime for comparisons, descriptors are often combined with dimensionality reduction approaches like PCA [47] or Gaussian random projections [66], or compression techniques like product quantization [25]. Approximate nearest neighbor and inverted indexes play an important role for large-scale image matching [43, 37, 64]. Please refer to [60] for discussion of further aspects of descriptor-based visual place recognition.

B. Exploiting semantic for localization

There is an intuitive benefit from using semantic for localization resulting in an accordingly rich variety or approaches in the literature. A particular hope is that despite severe appearance changes due to changing illumination, weather, or seasons, the semantics of observed scenes can be robustly matched [56, 67]. We will provide a short list of related approaches that exploit semantic, however, a full survey is beyond the capabilities of this paper.

Semantic information can be exploited at different parts of a localization pipeline. Assigned semantic meaning can be used to amplify [33] or inhibit [32] image features of particular classes, or during training of descriptor models [58]. [33] captures semantic scene context of local features in descriptors and use semantics to improve the matching procedure. [54] proposes a full SLAM system on the level of objects. For specific localization problems, specific semantic features of the environment can be used, e.g. pose estimation on roads with lane markings [59]. [65] uses semantic segmentations for long-term localization against semantic 3-D maps. [3] uses object detection and random finite set representations to localize against a map of semantic objects.

Semantic segmentations are for example used in [69] to evaluate the consistency of feature matches for localization, in [63] for geo-location including discovery of commonly occurring scene layouts, and in [75] to obtain semantic edges that can be used for localization. [17] extract semantic graphs

[1] https://www.tu-chemnitz.de/etit/proaut/VSR

from semantic segmentation and use them to compute random walk descriptors for fast matching against a database graph.

We will evaluate our proposed approach in combination with other holistic descriptors. This type of complementary usage of semantic is related to semantic feature reweighting [33, 32, 27] or semantic verification [67]. A strongly related approach is the Local Semantic Tensor (LoST) [16] that aggregates feature descriptors over semantic classes into a holistic descriptor. This approach was also extended with a local feature matching pipeline particularly designed for matching opposite views of a scene.

C. Vector Symbolic Architectures

Vector Symbolic Architectures (VSA) (also known as Hyperdimensional Computing or computing with large random vectors) is an established class of approaches to solve *symbolic* computational problems using mathematical operations on large numerical vectors with thousands of dimensions [26, 51, 19, 57]. Using embeddings in high-dimensional vector spaces to deal with ambiguities is well established in natural language processing [6]. VSAs make use of additional operations on high-dimensional vectors. So far, VSAs have been applied in various fields including robotics [45], addressing catastrophic forgetting in deep neural networks [9], medical diagnosis [73], fault detection [29], analogy mapping [52], reinforcement learning [30], long-short term memory [11], text classification [31], and synthesis of finite state automata [49]. They have been used in combination with deep-learned descriptors before, e.g. for sequence encoding [45] and local feature aggregation [44]. A particularly related VSA are spatial semantic pointers [34], a variant of the Semantic Pointer Architecture [15], that processes vector encodings of symbols with positions in images using a complex vector space and fractional binding [34]. The following Sec. III includes a short introduction to VSA principles.

III. Algorithmic Approach

As illustrated in Fig. 1, the input to our approach is an image, the output is a single numeric vector that encodes the spatial layout and appearance of semantic entities in this image. We build on available, readily trained neural networks to generate a semantic segmentation and to obtain feature map activations for appearance description. The semantic segmentation is parsed into a set of discrete entities, each with a set of properties (semantic class, spatial relations to other classes, ...). To be able to efficiently match this type of structure, we encode each piece of information in an individual high-dimensional vector and use operations from Vector Symbolic Architectures (VSA) to combine all the vectors from all entities of an image into a single vector (e.g. with 4,096 dimensions) that can then serve as a holistic descriptor for place recognition.

A. Preleminaries on Vector Symbolic Architectures

The operations from Vector Symbolic Architectures (VSA) [51, 26, 19] allow the structured combination of multiple vectors from the same high-dimensional vector space into a single vector from the same space. "Structured" means that we control the similarity of vectors. For example, we can encode the assignment of a value to a variable (aka. role-filler pairs) by encoding the "variable" and the "value" each in a d-dimensional vector, and then use the VSA's binding operation (explained below) to combine both in a single d-dimensional vector. The output vector will be dissimilar to each input vector but allows to later recover a vector that is very similar to the value-vector if we query with the variable-vector. This type of VSA operation build on the sometimes surprising geometric properties of high-dimensional vector spaces, e.g., that high-dimensional iid. *random* vectors are almost sure pairwise very dissimilar (quasi-orthogonal). VSAs and the following algorithmic description heavily rely on such random vectors to represent unrelated symbols (e.g. different semantic classes). Although this type of computation is very common in the VSAs literature, it is significantly different to conventional algorithmic descriptions. Please refer to [26, 45, 51] for general introductions to Vector Symbolic Architectures.

B. Vector space and operations

We adopt the frequency-domain Holographic Reduced Representation (FHRR) framework of [51] (which is one particular VSA). Each element is a d-dimensional complex valued vector, we will use $d = 4,096$ and $\mathbb{C}^{4,096}$.[2] This choice is based on the experimental comparison from [57] and the compatibility of FHRR vectors with fractional binding [34] for systematically encoding scalar values to vectors.

Each piece of information will be stored in a high-dimensional distributed vector representation, in the sense that information is encoded over all dimensions of the vector instead of one single number. This allows to encode both symbolic information (e.g. class labels) and numeric information (e.g. a vector of feature map activations) in a unified representational substrate. Relations between vectors are evaluated using their cosine similarity. Since in high-dimensional spaces, random vectors are very likely almost orthogonal (quasi-orthogonal) [26], we can use random vectors to encode unrelated symbols (e.g. different variable names or classes) with only a very small chance of ever confusing them based on their vector similarity. We will use the algebraic operations *bundling* $\oplus$ and *binding* $\otimes$ to combine vector information.

- **Bundling** $\oplus$ is used to store multiple input vectors in a set-like representation, where the result is a vector that is similar to each input vector. The implementation of the bundling $\oplus$ operator is an *element-wise addition* of the complex vector-values.
- **Binding** $\otimes$ is used to store variable-value pairs. The result of binding is dissimilar to each input vector, but each of the input vectors can be (approximately) restored. Binding is similarity preserving, that means that $\forall A, B, C, D \in \mathbb{C}^d : A \otimes B$ is similar to $C \otimes D$ iff A is

[2]The output VSR will only contain the angles of the complex number and thus will be from $\mathbb{R}^{4,096}$.

similar to C *and* B is similar to D (or A to D and B to C). In case of the complex FHRR, the binding $\otimes$ operation is implemented as an *element-wise multiplication* of the complex values.

An important property of these operations is that the output is a vector from the same vector space as the input vectors. This allows to combine these simple operations to encode complex structured information.

To also include information from structured numbers (e.g. the x-coordinate of an object in an image), we use the **fractional binding** mechanism proposed by Komer et al. [34] to systematically encode scalar values in vectors. "Systematically" means that similar scalar values (small euclidean distance) are encoded to similar vectors (small angular distance). Fractional binding encodes a real scalar value x in a complex vector from $\mathbb{C}^d$ by

$$fracBind_B(x) := B^{\lambda \cdot x} \tag{1}$$

where $B \in \mathbb{C}^d$ is a fixed random vector and λ is a scaling factor that controls how fast the vector similarity changes with changes of the encoded scalar x (illustrated in Fig. 2). For encoding scalars with different meaning (e.g. x and y coordinates), different random base vectors B_x and B_y can be used. Independent of the similarities between the scalars x and y, $fracBind_{B_x}(x)$ and $fracBind_{B_y}(y)$ will have a low similarity.

As proposed in [51], we will restrict each complex number to magnitude 1. This simplifies some of the required computations and allows to store each vector element by a single scalar, which is the phase angle of the complex number (instead of storing phase and magnitude or real and imaginary parts of a general complex number). In particular, the output vector will only contain the angles and can thus be stored as a real-valued vector (with the same memory footprint as, e.g., a 4,096 dimensional NetVLAD descriptor). When storing this angle-representation the following (mathematical equivalent) simplifications arise: Random vectors are created by iid. sampling each dimension uniformly from $[-\pi, \pi]$. Vector similarity is evaluated as average cosine of the angle differences. Binding $\otimes$ simplifies to element-wise addition and fractional binding to element-wise multiplication. However, for bundling, the angles have to be converted into complex numbers before addition. For consecutive bundle operations, we can stay in the full complex representation. After bundling, we convert back to the angle representation for further processing or storing.

C. Vector semantic representation of images

We will use the above vector operations to generate a $d = 4,096$ dimensional image descriptor coined Vector Semantic Representation (VSR) for each input image. A VSR is a vector from $\mathbb{R}^d$, however, please keep in mind that the elements actually represent angles of complex numbers.

A note on notation: We will use small letters to refer to scalar numbers and capital letters to refer to vectors. We will use the summation symbol $\sum_i X_i$ to refer to bundling elements X_i, even if X_i is an angle-representation and thus the summation involves conversion to a full complex number (i.e., $\sum_i X_i := X_1 \oplus X_2 \oplus X_3 ...$).

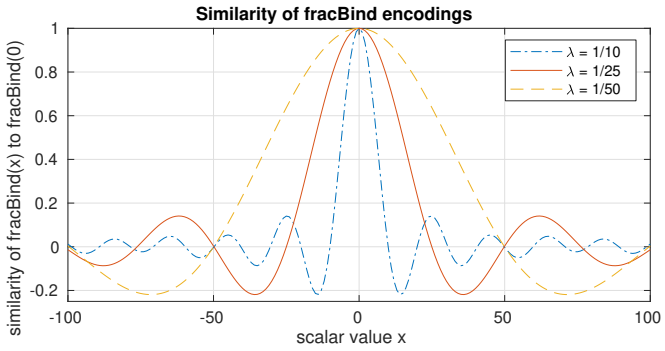

Fig. 2. Encoding similar scalar values with fractional binding results in high-dimensional vectors with high similarity. A scaling parameter λ can be used to influence the decay of the vector similarities. This resembles the sinc() function, the visible oscillation is a result of the periodic behavior of exponentiation in the complex domain, see [34] for more details.

The VSR of an image is the combination of all of its semantic entities $i = 1...k$:

$$VSR = \sum_i \omega_i \cdot E_i \tag{2}$$

ω_i is a weighting factor computed from the size of the i-th entity (it is the square root of the entity's area in pixels).

1) Finding semantic entities: We use the term "semantic entity" to refer to "things" and "stuff" [5]. Object detection algorithms can (primarily) find image objects with well-defined shape ("things") (e.g. traffic lights, signs, or poles). However, for tasks like place recognition we want to additionally use semantic information of amorphous background regions like "vegetation" or "terrain". Therefore, we use the connected components of a standard semantic segmentation approach as semantic entities. A possible future extension of this simple approach could use recent developments from panoptic segmentations [28] that combine both approaches.

We use Hierarchical Multi-Scale Attention [68] to assign a semantic class label from all non-dynamic Cityscapes classes to each pixel. For connected components, we use 8-neighborhood and create an entity boundary wherever the class label changes. The result is a list of entities $e_i : i = \{1, 2, ..., k\}$

Each entity consists of its spatial semantic information SSI_i including the relation to other entities and information about its appearance encoded in the vector A_i:

$$E_i = SSI_i \oplus A_i \tag{3}$$

2) Spatial Semantic Information SSI_i: The SSI_i is a single d-dimensional vector that comprises information about the semantic class of the i-th entity, its location in the image, and the classes of neighbored entities. It is computed by

$$SSI_i = C_i \otimes N_i \otimes S_i \tag{4}$$

Fig. 3 illustrates how each of these three vectors is created. C_i encodes the class c_i of entity e_i. We create a fixed random

vector C_{class} for each of the 11 non-dynamic Cityscapes classes $\mathcal{C} = \{C_{class} : class \in \{$*road, sidewalk, building, wall, fence, pole, traffic light, traffic sign, vegetation, terrain, sky*$\}\}$. The vector C_i is simply the corresponding vector from $\mathcal{C}$ for the entity's class c_i:

$$C_i = C_{c_i} \in \mathcal{C} \tag{5}$$

It is essential to keep these random vectors fixed for each class across all entities and across all images. Based on the quasi-orthogonality property of high-dimensional spaces, these random vectors are pairwise non-similar. The fact that the angle between the vectors $C_{building}$ and $C_{sidewalk}$ is roughly 90 degree is the vector-geometric way to express that these are two semantically different classes.[3]

To encode the semantic neighborhood relations of an entity, we accumulate the boundary length to entities of other classes for each direction.[4] Again, we use a random (but fixed) vector for each of the 8 canonical directions (top-of, top-left-of, left-of, ...). We can refer to a class-direction combination by binding the two corresponding vectors. The resulting vector will be non-similar to any other class-direction combination. The neighborhood vector is then bundled over all appearing direction-class combinations, each weighted by the length of the boundary $\beta_{C,D}$:

$$N_i = \sum_{D \in \mathcal{D}} \sum_{C \in \mathcal{C}} \beta_{C,D} \cdot (C \otimes D) \tag{6}$$

The (rough) shape and location are encoded using a $n_x \times n_y$ grid over the image as illustrated in the bottom-right part of Fig. 3. We count for each grid cell the proportion $p_{x,y}$ of the grid cell area that is covered by entity e_i. To encode this in a vector, we use a fixed random vector $G_{x,y}$ for each grid cell and compute the weighted bundle over all covered grid cells:

$$S_i = \sum_{x=1}^{n_x} \sum_{y=1}^{n_y} p_{x,y} \cdot G_{x,y} \tag{7}$$

The similarity of the resulting vector S_i will be the more similar to each vector $G_{x,y}$, the higher the proportion $p_{x,y}$ is. Moreover, the S vectors of two entities will be the more similar, the higher the amount of shared grid proportions.

In particular for this shape/location encoding, there is a variety of alternative approaches to encode this information in a distributed vector. We do not claim, that these encodings are the best possible (which is very likely not the case), however, they are sufficiently good to create a useful descriptor using VSA principles that can compete with the state of the art. A straight-forward alternative approach would be to use fractional binding to systematically encode the locations. So far, we did not yet evaluate this or other approaches. However, below, we will use fractional binding for a second procedure

[3]If there were related classes (e.g. different types of trees), one could use more similar vectors for these related classes, e.g. based on PSI [73].

[4]Since we do this for all entities, we represent a neighborhood relation of two entities in both directions, once in each entity.

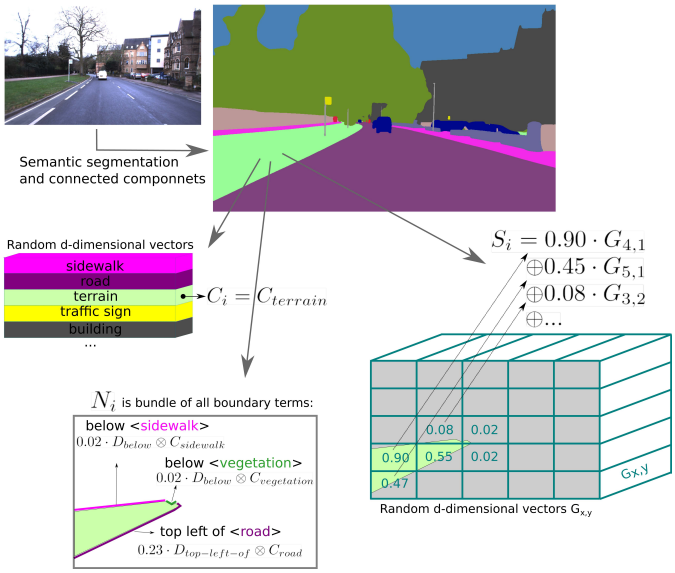

Fig. 3. Illustration of the spatial semantic information part of an image entity e_i consisting of class vector C_i, neighborhood encoding N_i, and shape/location encoding S_i. Together with the appearance encoding A_i, these vectors are the basis for the proposed vector semantic representation (VSR).

where we encode location: the systematic position encoding in the appearance vector.

3) Appearance vector A_i: Feature map responses from convolutional network layers are an established way to create descriptors [35, 16, 2]. To describe the appearance of an entity, we use the DELF [47] approach to detect a set of salient regions in this entity and use their feature map responses f_j and positions x_j, y_j. A_i is computed by:

$$A_i = \sum_j fft(\hat{f}_j) \otimes fracBind_x(x_j) \otimes fracBind_y(y_j) \tag{8}$$

We fist convert the feature map response vector f_j to a distributed d-dimensional vector $\hat{f}_j$ using a Gaussian random projection followed by mean normalization of all features of one image. A Fast Fourier Transform *fft()* is used to create a complex representation. Fractional binding *fracBind()* is used to encode the position, we use different random basis vectors for x and y directions. The resulting vector A_i is a single complex valued vector that describes the appearance of entity e_i. As said before, the vectors A_i and SSI_i of all entities are combined to the VSR descriptor using eq. 2.

IV. Experiments

A. Experimental setup

We will evaluate the VSR approach on standard place recognition datasets from mobile robotics. We use 13 sequence comparisons from three urban driving datasets with different characteristics regarding urban or suburban environment, appearance changes, single or multiple visits of places, possible stops, or viewpoint changes: **OxfordRobotCar** [40], **CMU** Visual Localization [4], and **StLucia** Various Times of the

TABLE I

AVERAGE PRECISION OF THE PROPOSED VSR APPROACH, OTHER DESCRIPTORS, AND THE COMBINATIONS OF DESCRIPTORS ON ALL DATASETS. THE FIRST TABLE COMPARES DESCRIPTORS AND SHOWS IMPROVEMENT BY COMBINATION WITH VSR. THE SECOND TABLE COMPARES DIFFERENT ALTERNATIVE COMBINATIONS WITH THE BEST PERFORMING DESCRIPTOR NV. IN EACH TABLE, THE BEST RESULT PER DATASET IS HIGHLIGHTED (EXCLUDING NV+DV+VSR). FOR COMBINED APPROACHES, THE COLORED ARROWS INDICATE LARGE ($\geq$25% BETTER/WORSE) OR MODERATE ($\geq$5%) DEVIATION COMPARED TO THE DESCRIPTOR THAT IS NAMED FIRST (EXCEPT FOR NV+DV+VSR WHICH COMPARES AGAINST NV+DV).

	Database	**Query**	**VSR** ours	**NV** [2]	**NV+VSR** [2]+ours	**DV** [72]	**DV+VSR** [72]+ours	**AN** [35]	**AN+VSR** [35]+ours	**HN** [8]	**HN+VSR** [8]+ours	**DELG** [7]	**DELG+VSR** [7] + ours
OxfordRobotCar	2014-12-09-13-21-02	2015-05-19-14-06-38	0.84	0.78	0.89 ↗	0.61	0.83 ↑	0.24	0.68 ↑	0.25	0.61 ↑	0.86	**0.93** ↗
	2014-12-09-13-21-02	2015-08-28-09-50-22	0.59	0.60	**0.70** ↗	0.43	0.63 ↑	0.11	0.36 ↑	0.09	0.30 ↑	0.17	0.44 ↑
	2014-12-09-13-21-02	2014-11-25-09-18-32	0.78	0.87	0.89 →	0.87	**0.90** →	0.42	0.70 ↑	0.41	0.69 ↑	0.69	0.83 ↗
	2014-12-09-13-21-02	2014-12-16-18-44-24	0.17	0.55	**0.67** ↗	0.11	0.33 ↑	0.07	0.25 ↑	0.08	0.31 ↑	0.10	0.44 ↑
	2015-05-19-14-06-38	2015-02-03-08-45-10	0.88	0.92	**0.95** →	0.25	0.53 ↑	0.36	0.84 ↑	0.42	0.83 ↑	0.78	0.91 ↗
	2015-08-28-09-50-22	2014-11-25-09-18-32	0.61	0.61	**0.70** ↗	0.38	0.54 ↑	0.09	0.39 ↑	0.11	0.44 ↑	0.35	0.59 ↑
CMU	20110421	20100901	0.61	0.73	0.75 →	0.66	0.74 ↗	0.44	0.61 ↑	0.55	0.63 ↗	**0.81**	0.78 →
	20110421	20100915	0.71	0.77	0.78 →	0.75	0.77 →	0.59	0.71 ↗	0.67	0.72 ↗	**0.79**	0.78 →
	20110421	20101221	0.56	0.56	0.61 ↗	0.49	0.59 ↗	0.34	0.56 ↑	0.40	0.57 ↑	0.61	**0.63** →
	20110421	20110202	0.47	0.61	**0.66** ↗	0.49	0.55 ↗	0.33	0.48 ↑	0.37	0.48 ↑	0.54	0.60 ↗
StLucia	100909_0845	180809_1545	0.35	0.02	0.19 ↑	0.22	0.33 ↑	0.36	0.38 →	**0.43**	0.42 →	0.03	0.29 ↑
	100909_1000	190809_1410	0.46	0.07	0.36 ↑	0.44	**0.56** ↑	0.47	0.50 ↗	0.52	0.52 →	0.13	0.48 ↑
	100909_1210	210809_1210	0.53	0.51	0.64 ↑	**0.78**	0.76 →	0.54	0.56 →	0.59	0.59 →	0.59	0.63 ↗
Worst case			0.17	0.02	0.19 ↑	0.11	**0.33** ↑	0.07	0.25 ↑	0.08	0.30 ↑	0.03	0.29 ↑
Best case			0.88	0.92	**0.95** →	0.87	0.90 →	0.59	0.84 ↑	0.67	0.83 ↗	0.86	0.93 ↗
Average case (mAP)			0.58	0.58	**0.68** ↗	0.50	0.62 ↗	0.34	0.54 ↑	0.38	0.55 ↑	0.50	0.64 ↑

	Database	**Query**	**NV** [2]	**NV + VSR** [2] + ours	**NV + LoST** [2] + [16]	**NV + DV** [2] + [72]	**NV + DELG** [2] + [7]	**NV + AN** [2] + [35]	**NV + HN** [2] + [8]	**NV + DV + VSR** [2] + [72] + ours
OxfordRobotCar	2014-12-09-13-21-02	2015-05-19-14-06-38	0.78	**0.89** ↗	0.85 ↗	0.78 →	0.86 ↗	0.77 →	0.74 ↘	0.86 ↗
	2014-12-09-13-21-02	2015-08-28-09-50-22	0.60	**0.70** ↗	0.64 ↗	0.62 →	0.45 ↘	0.50 ↘	0.47 ↘	0.69 ↗
	2014-12-09-13-21-02	2014-11-25-09-18-32	0.87	0.89 →	0.89 →	**0.90** →	0.85 →	0.85 →	0.84 →	0.91 →
	2014-12-09-13-21-02	2014-12-16-18-44-24	0.55	**0.67** ↗	0.55 →	0.48 ↘	0.44 ↘	0.59 ↗	0.61 ↗	0.61 ↑
	2015-05-19-14-06-38	2015-02-03-08-45-10	0.92	**0.95** →	0.92 →	0.72 ↘	0.93 →	0.93 →	0.92 →	0.84 ↗
	2015-08-28-09-50-22	2014-11-25-09-18-32	0.61	**0.70** ↗	0.64 ↗	0.58 →	0.58 →	0.56 ↘	0.58 ↘	0.65 ↗
CMU	20110421	20100901	0.73	0.75 →	0.76 →	0.77 ↗	**0.79** ↗	0.73 →	0.74 →	0.78 →
	20110421	20100915	0.77	0.78 →	0.78 →	**0.80** →	**0.80** →	0.77 →	0.78 →	0.80 →
	20110421	20101221	0.56	0.61 ↗	0.60 ↗	0.58 →	**0.62** ↗	0.60 ↗	0.60 ↗	0.63 ↗
	20110421	20110202	0.61	**0.66** ↗	0.62 →	0.61 →	**0.66** ↗	0.62 →	0.63 →	0.64 →
StLucia	100909_0845	180809_1545	0.02	0.19 ↑	0.03 ↑	**0.20** ↑	0.04 ↑	0.16 ↑	0.19 ↑	0.30 ↑
	100909_1000	190809_1410	0.07	0.36 ↑	0.10 ↑	**0.44** ↑	0.14 ↑	0.34 ↑	0.35 ↑	0.54 ↗
	100909_1210	210809_1210	0.51	0.64 ↑	0.59 ↗	**0.79** ↑	0.62 ↗	0.65 ↑	0.65 ↑	0.78 →
Worst case			0.02	0.19 ↑	0.03 ↑	**0.20** ↑	0.04 ↑	0.16 ↑	0.19 ↑	0.30 ↑
Best case			0.92	**0.95** →	0.92 →	0.90 →	0.93 →	0.93 →	0.92 →	0.91 →
Average case (mAP)			0.58	**0.68** ↗	0.61 ↗	0.64 ↗	0.60 →	0.62 ↗	0.62 ↗	0.69 ↗

Day [20]. For OxfordRobotCar, we sampled sequences at 1Hz with the recently published accurate ground truth data [41].

We compare to the following descriptors: NetVLAD **NV** [2]: We use the authors' VGG-16 version[5] with whitening trained on the Pitts30k dataset (4,096-D). DenseVLAD **DV** [72]: We use the authors' version[6] with 128-dimensional SIFT descriptors and 128 words trained on 24/7 Tokyo dataset, as well as PCA projection to 4,096-D. AlexNet **AN** [35]: We use the conv3 output of Matlab's ImageNet model and the full 65k dimensional descriptor. HybridNet **HN** [8]: We use the authors' version[7] and the full 43k dimensional descriptor. **DELG** [7]: We use the implementation from TensorFlow models with ResNet101 trained on a subset of the Google Landmarks Dataset v2 (GLDv2-clean) which was amongst best in [7]. **LoST** [16]: We use the authors' LoST[8] version (not LoSTX, which includes additional keypoint matching and was designed for matching opposing views, not for aligned views).

To generate semantic segmentations, we use the Cityscapes model from the authors' version[9] of Hierarchical Multi-Scale Attention [68]. To extract local features on entities, we use the TensorFlow Hub implementation[10] of DELF [47] and extract a maximum of 200 features per image. We use all entities with a minimum size of 10 pixels and default scales $\lambda_x = 4/w$ and $\lambda_y = 6/h$ for fractional binding in x and y direction for images of size $w \times h$. The number of dimensions in VSR is 4,096.

For evaluation, we compute pairwise similarity matrices between database and query image sets and compare them to ground-truth knowledge about place matchings using a series of thresholds. We report average precision (AP) computed as area under the resulting precision-recall curve, as well as achieved recall using the best k matchings. To combine an existing descriptor with VSR (e.g. **NV+VSR**, or two existing descriptors), we simply perform an elementwise multiplication of their pairwise image similarity matrices. We do not apply pre- or postprocessing steps like dataset standardization [61] or sequence evaluation (e.g. [42]). Of course, all evaluated approaches can be combined with such additional techniques.

B. Place recognition performance

The upper part of table I shows the place recognition performance of the proposed VSR approach and the other descriptors. The average case performance of VSR is on par with the best performing other descriptor (NetVLAD, NV). The worst case performance indicates that for each descriptor,

[5]https://github.com/Relja/netvlad

[6]http://www.ok.ctrl.titech.ac.jp/~torii/project/247/

[7]https://github.com/scutzetao/DLfeature_PlaceRecog_icra2017

[8]https://github.com/oravus/lostX

[9]https://github.com/NVIDIA/semantic-segmentation

[10]https://tfhub.dev/google/delf/1

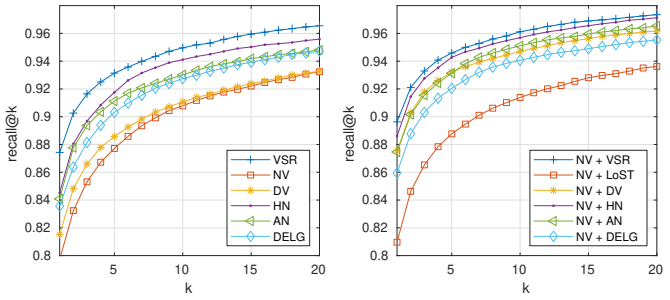

Fig. 4. Achieved recall when using the best k matching candidates per query image, averaged over all datasets. *(left)* Single descriptors (the curve of LoST would be below the shown part). *(right)* Combinations with NetVLAD.

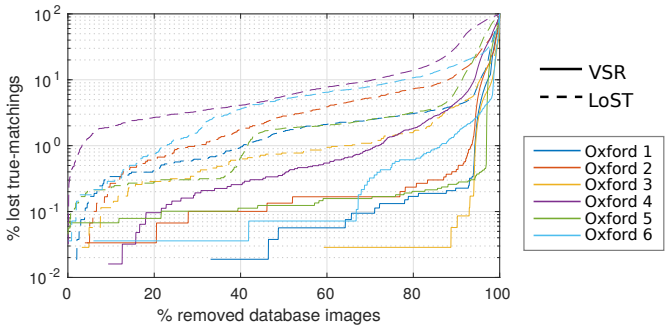

Fig. 5. Evaluation of recall integrity on the six OxfordRobotCar comparisons (the numbers correspond to appearance in table I). Lower is better.

there are one or multiple datasets where the performance significantly drops. Although VSR provides the best worst case performance, the achieved average recall on the fourth OxfordRobotCar dataset is only 0.17. This is largely due to the problems of the semantic segmentation algorithm with the low-illumination conditions in the query sequence. However, these conditions are very challenging for all descriptors. Appropriate pre- or postprocessing steps like dataset standardization [61] or sequence evaluation (e.g. [42]) can presumably improve the performance of all descriptors.

A significant improvement of the worst-case performance (of at least 25%) for all descriptors is achieved by combination with VSR (as said before, the combination is a simple element-wise multiplication of the pairwise similarity matrix, computational effort is discussed below). For all evaluated descriptors, also the average case performance is improved by combination with VSR. This demonstrates that the semantic information from VSR can complement existing descriptors. The combination with NetVLAD (NV+VSR) provides the best average case results.

This type of combination tends to improve results also for other combinations of existing descriptors. The lower part of table I shows results of different combinations of existing descriptors (we selected NV as base since it is the best performing descriptor from the upper table). Most prominent, the worst case performance improves significantly for all combinations. However, only the two approaches that include semantic information (VSR and LoST) are able to never decrease the place recognition performance compared to the stand-alone NetVLAD. The additional performance gain by LoST is particularly interesting since its stand-alone performance (mAP=0.44, not shown in table I) is considerably worse than e.g. of DenseVLAD or DELG. The best combination that does not include semantic information is NetVLAD with DenseVLAD (NV+DV). The last column shows that additionally including semantic information using VSR to this combination can further improve the results (mAP moderately increases from 0.64 to 0.69).

C. Recall@k and semantic similarity as a necessary criterion

Global image descriptors (e.g. each of the above evaluated) are often used to select a small set of matching candidates from a large dataset for further validation [56]. Fig. 4 shows the achieved recall when selecting the k most similar database images per query (recall@k). The changed ranking of descriptors compared to the average precision measure from table I indicates that different qualities of the descriptor are evaluated. Most noticeably, the proposed VSR provides best results of all stand-alone descriptors (left plot) and of all combinations with NetVLAD (right plot).

The good performance of flattened feature map descriptors like AlexNet (AN) and HybridNet (HN) compared to the two VLAD descriptors NetVLAD (NV) and DenseVLAD (DV) are due to the moderate (but realistic for driving scenarios) viewpoint changes in these datasets. In particular LoST was designed to handle large viewpoint changes (including opposing views). For a more suitable comparison of VSR and LoST, we also evaluate a slightly different criterion: Instead of selecting a small number of candidates for a query, we remove an increasing fraction of low-similarity images from the database and measure the percentage of lost true matchings (we call this recall integrity). The goal is to treat semantic similarity as a necessary criterion for image matchings, but not require it to be a sufficient criterion. Recall@k returns the k most similar database images. As long as k is much smaller than the database size, then high descriptor similarity has properties of a necessary *and* sufficient criterion for candidate selection, since each of the k candidates has to be more similar than the waste majority of all database images.

To accommodate for the large spatial invariance of LoST, Fig. 5 evaluates this recall integrity for VSR and LoST on all sequence comparisons from OxfordRobotCar. It can be seen that based on the LoST descriptor, it is possible to remove about 60% of all database images from the list of potential matchings at a cost of 1-8 % of the true matchings (i.e., 1-8 % of the true matchings for this query image were also removed). With VSR, it is in turn possible to remove the same amount of potential matchings at a cost of 0.03 - 0.55 % of all true-matchings, or remove 92 % of the potential matchings at the same cost of 9 % loss. Again, this worst case scenario (from the evaluated datasets) is the particularly challenging forth Oxford sequence comparison (the purple curves). For all other comparisons, the loss is significantly smaller (please

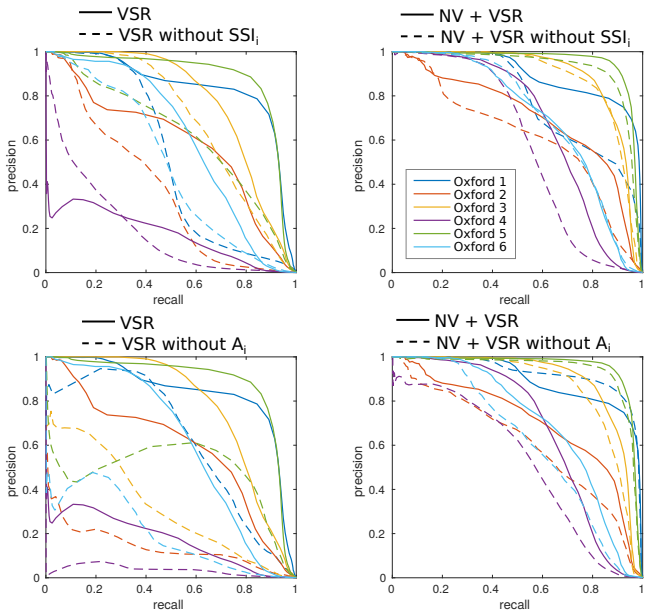

Fig. 6. Evaluation of the importance of SSI_i and A_i on the six OxfordRobotCar comparison. *(left column:)* Stand-alone VSR. *(right column:)* VSR in combination with NetVLAD. *(top row:)* VSR with and w/o SSI_i (using only A_i). *(bottom row:)* VSR with and w/o A_i (using only SSI_i).

note the logarithmic scale).

D. Ablation study

VSR combines all entities of an image, each entity is the bundle of a spatial semantic vector SSI_i (see eq. 4) and an appearance vector A_i (eq. 8). Fig. 6 evaluates the importance of each of the two parts. Solid lines show results of the full VSR as defined in eq. 2, the dashed lines of the same color indicate performance variation when removing either SSI_i or A_i. In particular the challenging fourth Oxford comparison (purple curves) shows a large degradation when only using the spatial semantic information (bottom-left plot). In this particular case, adding SSI_i even degrades performance in the high-precision regime (top-left plot). However, adding SSI_i can still increase the achieved recall at low precision values. In general, the large margin between the curves of the full VSR and the reduced versions demonstrate the importance of both components.

E. Computational effort

The runtime of the deep learning models heavily depends on the actual hardware and model choice. Using an Intel Core i7-7700K CPU and a NVIDIA GTX 1080Ti GPU, we can run our setup in less than a second per image. Our (completely) unoptimized Matlab implementation requires about 350ms to create a VSR descriptor of an image (90ms to find entities, 260ms for encoding). Although the encoding requires several operations on high dimensional vectors, they can be easily parallelized. Moreover, VSAs in general have very promising properties to run on very power-efficient hardware [53] which can be crucial for mobile application.

For place recognition, the runtime for image description is typically much less important than the time for matching against a large database. Since the VSR descriptor is a single vector of the same size as, e.g. a NetVLAD descriptor, matching is accordingly fast. In particular, we see no reason why VSR should not be compatible with existing approximate nearest neighbor matching techniques like product quantization [25], however, we did not yet test this.

For combining descriptors, comparison runtime becomes particularly important if the simple technique of element-wise multiplication of pairwise similarity matrices is applied, since this requires computation of multiple different descriptor distances for each image comparison. However, the good recall@k performance (Fig. 4) and the high recall integrity (Fig. 5) suggest that VSR has also potential to considerably reduce the number of matching candidates *before* computation of NetVLAD (or other additional) descriptor distances (provided a reasonable semantic segmentation is available).

V. Conclusion

We proposed Vector Semantic Representation (VSR) as descriptor for place recognition. It implements the high-level concept of encoding an image by the spatial layout of its semantic entities. We create semantic entities using semantic segmentations and encode for each entity the spatial semantic information and its appearance. The evaluation showed, that both components contribute to the place recognition performance, which is on par or better than the compared existing approaches.

Of course, the VSR relies on the quality of the semantic segmentation. If the segmentation is bad, the VSR performance will also degrade. This also limits the application to environments for which a semantic segmentation model is available. We restricted our evaluation to urban street scenes since we used a segmentation model for the Cityscapes classes. Although the viewpoint changes in the evaluated datasets are realistic for driving scenarios, their overall amount is limited. The application to hand-held camera images would very likely require a different encoding of entity locations than the grid approach, e.g. using fractional binding as well. Also, currently we rely on a direct neighborhood (a common boundary) between entities to establish a spatial relation. The VSA operations can very likely be used accordingly to encode other (more distant) relations. However, such extensions are left for future work.

In general, we consider Vector Symbolic Architectures (VSA) as a promising tool to encode diverse information (symbolic and numeric) in descriptors. The presented approach is just one (rather simple) way of creating semantic entities and then using the power of VSAs to create a descriptor for fast matching. This general concept is expected to be applicable to other tasks and to also benefit from future developments on extracting semantic (and other) information from images.

References

[1] R. Arandjelovic and A. Zisserman. All about vlad. In *Conf. on Computer Vision and Pattern Recognition*, 2013. doi: 10.1109/CVPR.2013.207.

[2] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. *Trans. on Pattern Analysis and Machine Intelligence*, 40(6), 2018. ISSN 0162-8828. doi: 10.1109/TPAMI.2017.2711011.

[3] Nikolay Atanasov, Menglong Zhu, Kostas Daniilidis, and George J. Pappas. Localization from semantic observations via the matrix permanent. *IJRR*, 35(1-3):73–99, 2016.

[4] H. Badino, D. Huber, and T. Kanade. Visual topometric localization. In *Intelligent Vehicles Symposium (IV)*, 2011. doi: 10.1109/IVS.211.5940504.

[5] Holger Caesar, Jasper R. R. Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In *CVPR*, pages 1209–1218. IEEE Computer Society, 2018.

[6] José Camacho-Collados and Mohammad Taher Pilehvar. From word to sense embeddings: A survey on vector representations of meaning. *J. Artif. Intell. Res.*, 63:743–788, 2018.

[7] Bingyi Cao, André Araujo, and Jack Sim. Unifying deep local and global features for image search. In *European Conference on Computer Vision (ECCV)*, 2020. ISBN 978-3-030-58565-5.

[8] Zetao Chen, Adam Jacobson, Niko Sünderhauf, Ben Upcroft, Lingqiao Liu, Chunhua Shen, Ian D. Reid, and Michael Milford. Deep learning features at scale for visual place recognition. In *ICRA*, pages 3223–3230. IEEE, 2017. ISBN 978-1-5090-4633-1.

[9] Brian Cheung, Alexander Terekhov, Yubei Chen, Pulkit Agrawal, and Bruno A. Olshausen. Superposition of many models into one. In *NeurIPS*, 2019.

[10] Mark Cummins and Paul M. Newman. Appearance-only slam at large scale with fab-map 2.0. *Int. J. Robotics Res.*, 30(9):1100–1123, 2011.

[11] Ivo Danihelka, Greg Wayne, Benigno Uria, Nal Kalchbrenner, and Alex Graves. Associative Long Short-Term Memory. In *Int. Conf. on Machine Learning*, 2016.

[12] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPR Workshops*, 2018.

[13] Carl Doersch, Saurabh Singh, Abhinav Gupta, Josef Sivic, and Alexei A. Efros. What makes paris look like paris? *ACM Trans. Graph.*, 31(4):101:1–101:9, 2012.

[14] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torii, and T. Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. doi: 10.1109/CVPR.2019.00828.

[15] Chris Eliasmith. How to build a brain: from function to implementation. *Synthese*, 159(3):373–388, 2007.

[16] Sourav Garg, Niko Sünderhauf, and Michael Milford. Lost? appearance-invariant place recognition for opposite viewpoints using visual semantics. In *Robotics: Science and Systems*, 2018.

[17] Abel Gawel, Carlo Del Don, Roland Siegwart, Juan Nieto, and Cesar Cadena. X-view: Graph-based semantic multi-view localization. In *IEEE Robotics and Automation Letters (RA-L)*, 2018.

[18] Ross W. Gayler. Multiplicative binding, representation operators, and analogy. In *Advances in analogy research: Integr. of theory and data from the cogn., comp., and neural sciences*, Bulgaria, 1998.

[19] Ross W. Gayler. Vector Symbolic Architectures answer Jackendoff's challenges for cognitive neuroscience. In *Int. Conf. on Cognitive Science*, 2003.

[20] A. J. Glover, W. P. Maddern, M. J. Milford, and G. F. Wyeth. Fab-map + ratslam: Appearance-based slam for multiple times of day. In *Int. Conf. on Robotics and Automation (ICRA)*, 2010. doi: 10.1109/ROBOT.2010.5509547.

[21] Kristen Grauman and Trevor Darrell. The pyramid match kernel: Efficient learning with sets of features. *Journal of Machine Learning Research*, 8(26):725–760, 2007.

[22] Syed Sameed Husain and Miroslaw Bober. Remap: Multi-layer entropy-guided pooling of dense cnn features for image retrieval. *IEEE Trans. Image Process.*, 28(10):5201–5213, 2019.

[23] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *Conference on Computer Vision and Pattern Recognition*, 2010.

[24] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Improving bag-of-features for large scale image search. *Int. J. Comput. Vis.*, 87(3):316–336, 2010.

[25] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(1):117–128, 2011.

[26] Pentti Kanerva. Hyperdimensional Computing: An Introduction to Computing in Distributed Representation with High-Dimensional Random Vectors. *Cognitive Computation*, 1(2):139–159, 2009.

[27] Hyo Jin Kim, Enrique Dunn, and Jan-Michael Frahm. Learned contextual feature reweighting for image geo-localization. In *CVPR*, 2017.

[28] Alexander Kirillov, Kaiming He, Ross B. Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *CVPR*, pages 9404–9413, 2019.

[29] D. Kleyko, E. Osipov, N. Papakonstantinou, V. Vyatkin, and A. Mousavi. Fault detection in the hyperspace: Towards intelligent automation systems. In *International Conference on Industrial Informatics (INDIN)*, 2015. doi: 10.1109/INDIN.2015.7281909.

[30] Denis Kleyko, Evgeny Osipov, Ross W. Gayler, Asad I. Khan, and Adrian G. Dyer. Imitation of honey bees' concept learning processes using Vector Symbolic Architectures. *Biologically Inspired Cognitive Architectures*, 14:57 – 72, 2015. ISSN 2212-683X. doi: https://doi.org/

10.1016/j.bica.2015.09.002.

[31] Denis Kleyko, Abbas Rahimi, Dmitri A. Rachkovskij, Evgeny Osipov, and Jan M. Rabaey. Classification and Recall With Binary Hyperdimensional Computing: Tradeoffs in Choice of Density and Mapping Characteristics. *IEEE Transactions on Neural Networks and Learning Systems*, 29(12):5880–5898, 2018. doi: 10.1109/TNNLS.2018.2814400.

[32] Jan Knopp, Josef Sivic, and Tomás Pajdla. Avoiding confusing features in place recognition. In *ECCV (1)*, volume 6311 of *Lecture Notes in Computer Science*, pages 748–761. Springer, 2010. ISBN 978-3-642-15548-2.

[33] Nikolay Kobyshev, Hayko Riemenschneider, and Luc Van Gool. Matching features correctly through semantic understanding. In *3DV*, pages 472–479. IEEE Computer Society, 2014. ISBN 978-1-4799-7000-1.

[34] Brent Komer, Terrence C. Stewart, Aaron Voelker, and Chris Eliasmith. A neural representation of continuous space using fractional binding. In *CogSci*, pages 2038–2043, 2019. ISBN 0-9911967-7-5.

[35] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[36] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[37] W. Li, Y. Zhang, Y. Sun, W. Wang, M. Li, W. Zhang, and X. Lin. Approximate nearest neighbor search on high dimensional data — experiments, analyses, and improvement. *IEEE Transactions on Knowledge and Data Engineering*, 32(8):1475–1488, 2020.

[38] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, November 2004. ISSN 0920-5691.

[39] S. Lowry, N. Sünderhauf, P. Newman, John J. Leonard, David Cox, Peter Corke, and Michael J. Milford. Visual place recognition: A survey. *Trans. Rob.*, 32(1), 2016. ISSN 1552-3098. doi: 10.1109/TRO.2015.2496823.

[40] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The oxford robotcar dataset. *The Int. Journal of Robotics Research*, 36(1):3–15, 2017.

[41] Will Maddern, Geoffrey Pascoe, Matthew Gadd, Dan Barnes, Brian Yeomans, and Paul Newman. Real-time Kinematic Ground Truth for the Oxford RobotCar Dataset. *CoRR*, abs/2002.10152, 2020.

[42] M. Milford and G. F. Wyeth. SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights. In *Int. Conf. on Robotics and Automation*, 2012. ISBN 978-1-4673-1403-9.

[43] Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *Int. Conf. on Computer Vision Theory and Applications*, 2009.

[44] Peer Neubert and Stefan Schubert. Hyperdimensional computing as a framework for systematic aggregation of image descriptors. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[45] Peer Neubert, Stefan Schubert, and Peter Protzel. An introduction to hyperdimensional computing for robotics. *Künstliche Intell.*, 33(4):319–330, 2019.

[46] Peer Neubert, Stefan Schubert, and Peter Protzel. A neurologically inspired sequence processing model for mobile robot place recognition. *IEEE Robotics and Automation Letters*, 4(4):3200–3207, 2019.

[47] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han. Large-scale image retrieval with attentive deep local features. In *Int. Conf. on Computer Vision (ICCV)*, 2017. doi: 10.1109/ICCV.2017.374.

[48] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *Int'l Journal of Computer Vision*, 42(3):145–175, 2001.

[49] E. Osipov, D. Kleyko, and A. Legalov. Associative synthesis of finite state automata model of a controlled object with hyperdimensional computing. In *Conference of the IEEE Industrial Electronics Society (IECON)*, 2017. doi: 10.1109/IECON.2017.8216554.

[50] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *Conference on Computer Vision and Pattern Recognition*, 2007. doi: 10.1109/CVPR.2007.383266.

[51] Tony Alexander Plate. *Distributed Representations and Nested Compositional Structure*. PhD thesis, Toronto, Ont., Canada, Canada, 1994.

[52] Dmitri A. Rachkovskij and Serge V. Slipchenko. Similarity-based retrieval with structure-sensitive sparse binary distributed representations. *Computational Intelligence*, 28(1):106–129, 2012.

[53] A. Rahimi, S. Datta, D. Kleyko, E. P. Frady, B. Olshausen, P. Kanerva, and J. M. Rabaey. High-Dimensional Computing as a Nanoscalable Paradigm. *IEEE Transactions on Circuits and Systems*, 64(9):2508–2521, Sep. 2017.

[54] Renato F. Salas-Moreno, Richard A. Newcombe, Hauke Strasdat, Paul H. J. Kelly, and Andrew J. Davison. SLAM++: Simultaneous Localisation and Mapping at the Level of Objects. In *CVPR*, 2013.

[55] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Improving image-based localization by active correspondence search. In *European Conf. on Computer Vision (ECCV)*, 2012.

[56] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomas Pajdla. Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[57] Kenny Schlegel, Peer Neubert, and Peter Protzel. A comparison of vector symbolic architectures. *CoRR*,

abs/2001.11797, 2020.

[58] Johannes Lutz Schönberger, Marc Pollefeys, Andreas Geiger, and Torsten Sattler. Semantic Visual Localization. In *CVPR*, 2018.

[59] Markus Schreiber, Carsten Knöppel, and Uwe Franke. LaneLoc: Lane marking based localization using highly accurate maps. In *Proc. IV*, 2013.

[60] Stefan Schubert and Peer Neubert. What makes visual place recognition easy or hard? *CoRR*, abs/2106.12671, 2021.

[61] Stefan Schubert, Peer Neubert, and Peter Protzel. Unsupervised learning methods for visual place recognition in discretely and continuously changing environments. 2020.

[62] X. Shen, Z. Lin, J. Brandt, S. Avidan, and Y. Wu. Object retrieval and localization with spatially-constrained similarity measure and k-nn re-ranking. In *Conference on Computer Vision and Pattern Recognition*, 2012. doi: 10.1109/CVPR.2012.6248031.

[63] Gautam Singh and Jana Košecká. Semantically Guided Geo-location and Modeling in Urban Environments. In *Large-Scale Visual Geo-Localization*, 2016.

[64] Josef Sivic and Andrew Zisserman. Video google: Efficient visual search of videos. In *Toward Category-Level Object Recognition*, volume 4170 of *Lecture Notes in Computer Science*, pages 127–144. Springer, 2006. ISBN 3-540-68794-7.

[65] E. Stenborg, C. Toft, and L. Hammarstrand. Long-term Visual Localization using Semantically Segmented Images. In *ICRA*, 2018.

[66] N. Sünderhauf, F. Dayoub, S. Shirazi, B. Upcroft, and M. Milford. On the Performance of ConvNet Features for Place Recognition. *CoRR*, abs/1501.04158, 2015.

[67] Hajime Taira, Ignacio Rocco, Jirí Sedlár, Masatoshi Okutomi, Josef Sivic, Tomás Pajdla, Torsten Sattler, and Akihiko Torii. Is this the right place? geometric-semantic pose verification for indoor visual localization. In *ICCV*, 2019.

[68] Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical multi-scale attention for semantic segmentation, 2020.

[69] Carl Toft, Erik Stenborg, Lars Hammarstrand, Lucas Brynte, Marc Pollefeys, Torsten Sattler, and Fredrik Kahl. Semantic Match Consistency for Long-Term Visual Localization. In *ECCV*, 2018.

[70] G. Tolias, T. Jenícek, and O. Chum. Learning and aggregating deep local descriptors for instance-level recognition. In *European Conf. on Computer Vision*, 2020.

[71] Giorgos Tolias, Yannis Avrithis, and Hervé Jégou. Image search with selective match kernels: Aggregation across single and multiple images. *Int. J. Comput. Vis.*, 116(3): 247–261, 2016.

[72] A. Torii, R. Arandjelović, J. Sivic, M. Okutomi, and T. Pajdla. 24/7 place recognition by view synthesis. In *Conf. on Computer Vision and Pattern Recognition*, 2015.

[73] Dominic Widdows and Trevor Cohen. Reasoning with Vectors: A Continuous Model for Fast Robust Inference. *Logic journal of the IGPL / Interest Group in Pure and Applied Logics*, (2):141–173, 2015.

[74] J. Yu, C. Zhu, J. Zhang, Q. Huang, and D. Tao. Spatial pyramid-enhanced netvlad with weighted triplet loss for place recognition. *IEEE Transactions on Neural Networks and Learning Systems*, 31(2):661–674, 2020.

[75] X. Yu, S. Chaturvedi, C. Feng, Y. Taguchi, T.-Y. Lee, C. Fernandes, and S. Ramalingam. VLASE: Vehicle Localization by Aggregating Semantic Edges. In *IROS*, 2018.