Dimensionality reduction to improve search time and memory footprint in content-retrieval tasks: Application to semiconductor inspection images

Article *in* Advances in Industrial and Manufacturing Engineering · September 2022

DOI: 10.1016/j.aime.2022.100097

CITATIONS
0

READS
45

7 authors, including:

Marisa Noemi Faraggi
OCTO
17 PUBLICATIONS **370** CITATIONS

SEE PROFILE

Advances in Industrial and Manufacturing Engineering 5 (2022) 100097

Contents lists available at ScienceDirect

Advances in Industrial and Manufacturing Engineering

journal homepage: www.elsevier.com/locate/aime

Regular articles

Dimensionality reduction to improve search time and memory footprint in content-retrieval tasks: Application to semiconductor inspection images

Thomas Vial *, Farah Dhouib, Louison Roger, Annabelle Blangero, Frédéric Duvivier, Karim Sayadi, Marisa N. Faraggi

OCTO Technology, 34, avenue de l'Opéra, 75002 Paris, France

ARTICLE INFO

Keywords:
Semiconductor manufacturing
Defectivity analysis
Image retrieval
Similarity search
Dimensionality Reduction
Principal Component Analysis (PCA)
Sparse Random Projection (SRP)
Isomap
Local Linear Embedding (LLE)
Deep learning (DL)
Computer vision

ABSTRACT

Quality control in semiconductors is a crucial step to produce high quality microchips. During the last years, advances in artificial vision have significantly improved image quality control techniques. In the semiconductor industry, automated visual inspection is fundamental to avoid human intervention and keep the pipeline sanitized. Different types of images are collected during this process, feeding image databases that continually grow and cannot be labelled by humans in an exhaustive manner. Advances in image retrieval search methods are fundamental to develop more efficient techniques that meet user requirements.

In this work we propose a dimensionality reduction approach on the feature vectors computed by a classifying deep learning model, while keeping a high retrieval performance. To validate this technique, we evaluate four well-known reduction algorithms on a subset of the full database: Principal Component Analysis (PCA), Sparse Random Projection (SRP), Isomap, Locally Linear Embedding (LLE), in combination with three similarity metrics: Euclidian (L_2), cosine and inner product. As the number of components of the vectors is reduced, the performance of the image retrieval is measured by recall, time to search, and memory footprint of the database.

PCA offers the best results, allowing a significant reduction in search time and memory usage, while SRP becomes an option only when the cosine distance is used. With PCA, we were able to divide the memory footprint by a factor of 16, the search time by 6, while maintaining an average recall of 0.96.

1. Introduction

Defect identification on production lines is a key task for many industries. A defect, by definition, is a failure or a set of failures that by nature or cumulative effect cause a part of a product to fail to meet applicable minimum acceptance standards or specifications. We can count on many different techniques to detect defects in the industrial domain (laser scanners (Hong-Seok and Mani, 2014), electric current (Brauer et al., 2014), weight, human inspection, etc.), but with the advances made in machine vision in recent years, the combination of cameras and artificial intelligence becomes the preferred method in different sectors (ElMaraghy and Bullis, 1989; Miao et al., 2019; Zhang et al., 2020; Haleem et al., 2021). The last 10 years have shown continuous development in computer vision thanks to neural networks (Murthy et al., 2020; Tulbure et al., 2022). These advances have allowed the development of highly accurate defect detection algorithms like *Region Based Convolutional Neural Networks* (R-CNNs) (Girshick et al., 2014; Girshick, 2015; He et al., 2016), *You Only Look Once* (YOLO) (Redmon et al., 2016) or *Single Shot Detectors* (SSD) (Liu et al., 2016). Those algorithms are at the core of applications that allow operators to inspect an increasing number of products at a high frequency.

In this article, we focus on the identification of defects in silicon wafers during the production of microchips. This type of product is subject to numerous quality controls using images at different stages in the production line. During the quality control pipeline, three types of images are collected, such as those shown in Fig. 1: (a) wafer maps, (b) optical images and (c) scanning electronic microscope (SEM).

In this case, the quality control process must tell whether the defect patterns detected in real time were already encountered in the past. The process involves querying a huge database of images, which can take a long time to search, making the task of the quality controller less efficient. In a highly technical and high-throughput industry such as semiconductor manufacturing, users expect performant tools. The

* Corresponding author.
E-mail addresses: thomas.vial@octo.com (T. Vial), farah.dhouib@octo.com (F. Dhouib), louison.roger@octo.com (L. Roger), frederic.duvivier@octo.com (F. Duvivier), karim.sayadi@octo.com (K. Sayadi), marisa.faraggi@octo.com (M.N. Faraggi).
URL: http://www.octo.com/ (T. Vial).

https://doi.org/10.1016/j.aime.2022.100097
Received 12 May 2022; Received in revised form 18 July 2022; Accepted 29 August 2022
Available online 5 September 2022

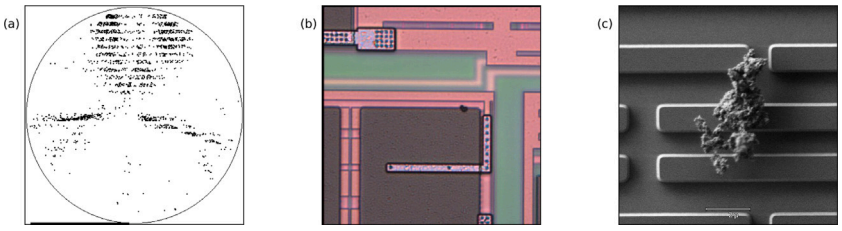

Fig. 1. Images collected in quality control pipelines. (a) Wafermaps, (b) optical and (c) scanning electronic microscope (SEM).

objective of this article is to show that it is possible to provide assistance to defectivity engineers by reducing the time of the search step, without degrading the relevance of its results.

The research is carried out in the context of a European best-in-industry project called MADEin4 (MADEin4, 2019). Within this project, OCTO Technology and STMicroelectronics Rousset have jointly developed an application named RETINA (*Research of Twin Images with Neural Algorithm*). The tool offers said assistance to engineers in their inspection routine. Faced with a newly detected defect, they are now able to quickly find similar ones that occurred in the past, and take insight from the associated process data.

Regarding the identification of defects, two parts are at play. The first part is a Deep Learning (DL) model that detects defects on inspection images, and infers a defect class from the features of the image. The DL model is built by STMicroelectronics and it is not the object of this work; the model is applied automatically on every inspection image, feeding an image database of currently ≈ 20 millions elements, this number continuously increasing.

The second part corresponds to the search engine, RETINA, which operates on abstract vectors encoded by the DL model, called *fingerprints* or *feature vectors*. When a RETINA user searches for past images similar to the new one, the application actually looks for feature vectors that are "close" to that of the reference image in the feature space. As the number of images in the historical database continually increases, search time becomes a key player in performing an efficient quality control task. The process described above is named as Content-based image retrieval (CBIR) (Jenni et al., 2015; Vita and Kamboj, 2016), and the time to search is a known weak point of the technique (Hameed et al., 2021).

This article aims to evaluate the impact of dimensionality reduction on retrieval performance, characterized by the time to search and the memory footprint, while maintaining high relevance of the results. A particular aspect of our situation is the rate of production of the images (and thus vectors). With thousands of inspections occurring every day from all fabs, there is no possibility for humans to manually assign a defect class to each and every image — an information that would be important to evaluate the relevance of search results before and after reducing the number of dimensions. Instead, we "trust" the result of classification by the DL model in our evaluation of dimensionality reduction operators.

The paper is organized as follows: in Section 2, we provide a summary of previous studies that applied the concept of dimensionality reduction to CBIR tasks. In Section 3, a detailed description of the problem is presented, and definitions are formulated. Section 4 corresponds to the Experimental Protocol where we present the dataset, the components of the experiment and the methodology applied to evaluate our assumptions. In Section 5, Results are presented analysing the choice of distance metrics and key parameters like the number of components. Conclusions are presented in Section 6.

2. Related work

Many studies are exploring the possibility of reducing the dimensionality of feature vectors to alleviate pressure on CBIR systems. The selection below offers an overview of the directions taken by those studies.

Some focus on benchmarking PCA, Isomap, LLE (Cheng et al., 2013) and other several supervised and unsupervised reduction algorithms on explicit visual characteristics of images, such as semantic features or colour histograms. Semantic features require substantial pre-processing of the images, based on human expertise. On the other hand, DL networks combine the pre-processing and the learning phases in one single step. Contrary to colour histograms, those DL features also encode visual patterns containing rich information.

Other works aim to quantify the impact of a PCA transformation on feature vectors produced by a DL model (Babenko et al., 2014; Fleet et al., 2014; Yandex and Lempitsky, 2015). They explore different algorithms to produce those vectors from activations within the network. *Babenko et al.*. Babenko et al. (2014) prove the superiority of vectors output by a model that was trained on a specific dataset, over one that came pre-trained on generic image datasets.

The works mentioned above are directed towards generic image retrieval, as shown by the datasets they founded their experiments on, e.g. ImageNet. It must be noted that the applications of deep-learning based CBIR is not limited to generic search engines. For example, Arai et al. (2018) use convolutional autoencoders to compress 3D brain MRI into compact vectors suitable for CBIR. Cao et al. (2018) apply dimensionality reduction on feature vectors obtained from pictures of clothes, for e-commerce applications. Closer to our own application, (Zhang et al., 2022) rely on mixed DL and hand-crafted features to enable visual inspection in the context of wool manufacturing — but they do not seem to take a step further towards CBIR.

Our work lies in the continuity of those studies. In our case, the DL network is given *a priori* and has its own training procedure, because it serves another purpose of classifying inspection images. To avoid impacting the online classification step with our own design, we take the feature vectors for granted and look for a way to optimize them for our specialized image-retrieval task.

3. Statements

In RETINA, the search activity is the identification of the K nearest neighbours of a given image, which we assume the user has selected among a total of N images. The neighbour search does not happen in the RGB (Red, Green, Blue) or HSV (Hue, Saturation, Value) space of the raw image, but rather in the abstract vector space computed by the DL network mentioned earlier. While the DL model's primary function is the online classification of defects, RETINA uses the activations of the last fully-connected layer as a fingerprint of each image, thus obtaining a unique vector of features per image.

The assumption is that inspection images from the same defect class will exhibit similar visual features, and that the DL network encodes the statistical distribution of those features in its weights during training. After training, any new image is feed-forwarded into the network, and the resulting activations form a vector that encodes the features of the image.

In our case, the vectors are produced by a Residual network, namely a ResNet-18 network (He et al., 2016), trained on historical inspection images, and are of dimension $P = 512$. The N images thus make

up a matrix M of size $N \times P$, each row of which representing the fingerprint of an image from the database. Given a reference image of index i_0, the objective is to find the K closest rows in M (excluding i_0), which optimize some metric d. We allow d to be the Euclidian distance (L_2, to be minimized), the inner-product pseudo-distance (IP, to be maximized) or the cosine pseudo-distance (cos, to be maximized).

Here we focus on the techniques that allow an improvement of *search runtime* and the *memory footprint* of the vectors containing the fingerprints[1].

Search runtime: RETINA is an interactive application that boosts quality control, and as such, one fundamental need is to deliver results in a few seconds. RETINA currently performs exact calculations of the (pseudo-)distances, $d(i, i_0)$, which are calculated for every candidate row $i \neq i_0$. We use the efficient CPU implementation from FAISS (Johnson et al., 2019), still the runtime is proportional to the number of elements in M, namely $N \cdot P$, in approximation at least[2].

Memory footprint: To allow fast calculations with FAISS, M is held fully in computer memory, avoiding comparatively slow disk accesses for the search. The total memory footprint of M is $4 \cdot N \cdot P$ bytes, 4 being the size of a single floating point variable. At the time of writing this document, N is approximately 20 millions, thus the footprint for such a matrix M is roughly $4 \times 20 \cdot 10^6 \times 512 = 40\,960\,000\,000$ bytes (38 GiB).

N is constantly growing as new inspections are carried out, so we are looking for a way to temporarily relieve the pressure on runtime and memory by decreasing the key figure $N \cdot P$. N is imposed by the STMicroelectronics' manufacturing activity; our approach here is explore the possibility of decreasing the value of P to some number $P' < P$, without impairing the relevance of the search results.

The approach to validate in this paper is the dimensionality reduction of M.

4. Experimental protocol

The context and challenge were stated in previous section. Here we focus on the description of the methodology that allows us to validate the following assumption: some dimensionality reduction is possible without sacrificing the relevance of the search results.

We propose to run different dimensionality reduction algorithms on an extract of the total database. This allows us to obtain smaller vectors (P' between 10 and 510 components), then we analyze the ability of those reduced vectors to perform a similarity search, and evaluate the results. For this we will consider three similarity metrics: L_2, IP and cos.

4.1. Experiments

In order to validate our approach of reducing the feature vector component size without loosing quality in the image identification, we define an *experiment* as a collection of parameters from the list below:

- a dimensionality reduction algorithm A — one of *PCA* (Pearson, 1901), *SRP* (Bingham and Mannila, 2001), *Isomap* (Tenenbaum et al., 2000), and *LLE* (Roweis and Saul, 2000). We use the implementations from Scikit-Learn v1.0 (Pedregosa et al., 2011). Their exact configuration is detailed in the appendix;
- a number P' of components after reduction;
- a random subset of vectors T used for fitting the algorithms that work on the statistical distribution of the vectors (PCA, Isomap and LLE[3]) — the factor of interest is $|T|$, the size of the training set or *training size*;
- a random set of vectors E used for measuring the effect of the reduction — concretely, we sample 10 images from each defect class C_i, so $|E|$ is $10 \times 10 = 100$ for all experiments. We make sure that T and E do not overlap, to avoid training the algorithm A on evaluation images;
- a (pseudo-)distance metric d used to search for neighbours, as explained above — so d is one of L_2, IP and cos;
- a fold number $f \in [\![1, 5]\!]$ — each canonical experiment is repeated 5 times, to account for variations caused by the randomness in choosing T and E, and possibly in the training code of A.

The number of neighbours searched is kept constant: $K = 100$, since a search for any number $K' < K$ of neighbours would yield the same first K' elements from the result set — once A is fitted, the search procedure we use is deterministic.

With a set of experiments defined, the loop is described in Algorithm 1.

Algorithm 1: Run set of experiments.

foreach *experiment* $e \leftarrow (f, A, P', T, E, d)$ **do**
  $I, D \leftarrow search(E, N, K, D)$;
  Fit A with P' components using T as training set;
  $M' \leftarrow A(M)$;
  $I', D' \leftarrow search(E, M', K, d)$;
  Append (e, I, D, I', D') to experiment log;
end

$search(E, M, K, d)$ searches M for the closest K neighbours of each row in E, according to some (pseudo-)distance metric d. It returns a pair (I, D), where I contains the *image IDs* of the rows in M that were identified as neighbours, and D are the *(pseudo-)distances* between the reference vectors from E and those neigbours. Thus both I and D are matrices of dimension $|E| \times K$.

As in the description of the experiment loop, we refer to the IDs and distances *before* applying any algorithm A as I and D, and *after* applying A as I' and D'.

4.2. Dataset

Our database is composed of $N = 27\,695$ Scanning Electronic Microscope (SEM) images extracted from a bigger database by STMicroelectronics. For each image we have computed the associated feature vector using the DL model above mentioned.

Each image is tagged with a *defect class*, resulting from the classification by the DL model. The class is thus ultimately determined by the visual features of the image, as learnt by the neural network. For example, one class represents scratches on die surface, another one particles adhering to the surface, etc. (see Fig. 2). There are 10 classes, labelled C_0 to C_9 in the following. The images are not equally distributed among classes; the smallest class has 415 images while the biggest one has 5 555. Our experimental protocol ensures that all classes are equally sampled in each training set T.

The defect class does not convey a particular meaning for RETINA. We use it in this work to select the most meaningful similarity metrics, as explained in Section 5.1.

[1] Note that the constraints below refer to the current technical design of the RETINA application. Future versions will question those choices by allowing non-exhaustive search through indexing, slower disk access for less frequent queries, and distribution of the workload across several servers.

[2] Strictly speaking, the relationship is not completely linear because FAISS exploits Single Instruction Multiple Data (SIMD) capabilities of the processor, introducing a more complex dependency on P.

[3] When the target number of components P' is imposed, the SRP algorithm generates a random projection matrix that depends only on the initial dimension P, not on the vectors T or their number $|T|$.

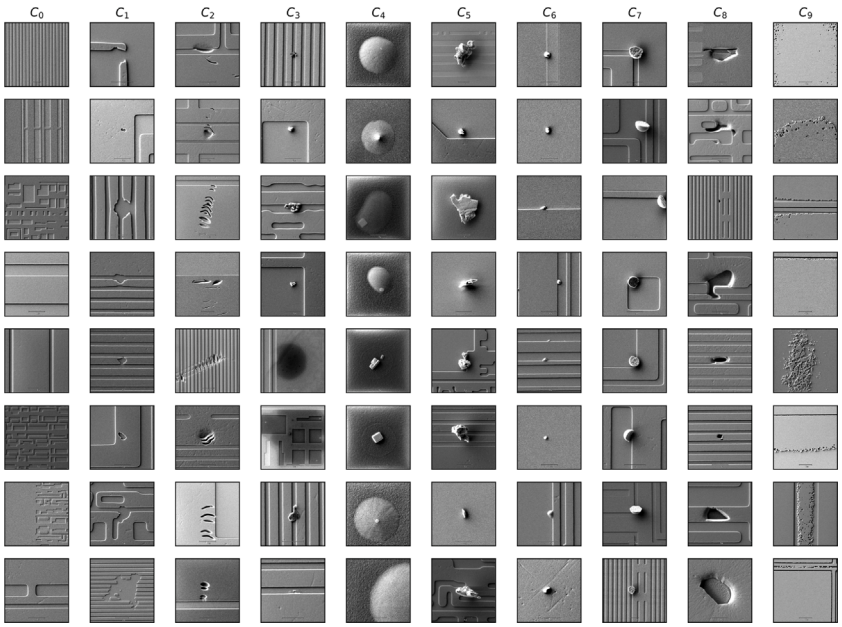

Fig. 2. Random samples from defect classes C_0 to C_9. Note that the classification being done by a neural network, inaccuracies are possible.

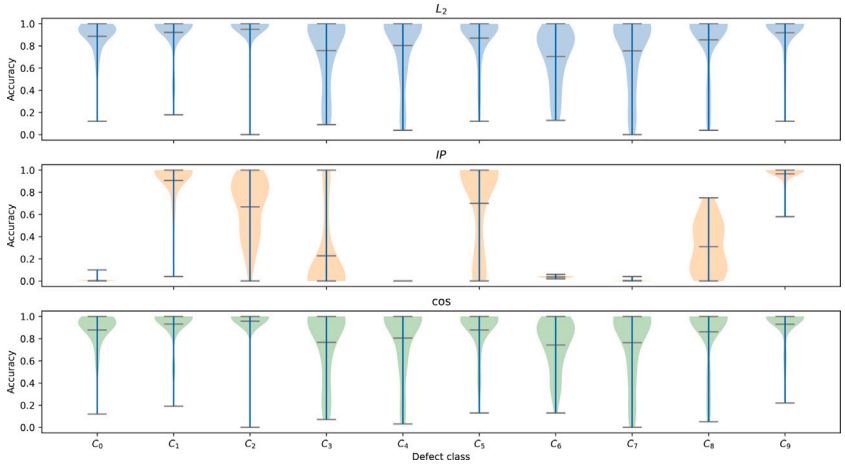

Fig. 3. Distributions of accuracies per metric and defect class.

4.3. Evaluation

4.3.1. Evaluating the choice of the (pseudo-)distance metric

While the (pseudo-)distance metric d is one attribute of the experiment among others, it plays a special role. In any typical application, it will be chosen by its ability to retrieve relevant results for the user, whether the search operates on full or reduced vectors.

To avoid misinterpreting results based on irrelevant metrics, we first evaluate the 3 candidates L_2, IP and cos for themselves, using the defect classes. If $C(i)$ is the defect class associated with any vector i from the database, we define the accuracy of d for a reference vector i_0 as:

$$accuracy(d, i_0) = \frac{\left|\left\{i \in I, C(i) = C(i_0)\right\}\right|}{K}$$

where I is the set of image IDs retrieved by the search operation using d as the metric.

The accuracy is a number between 0 and 1. 1 means that all neighbours retrieved by the search operation belong to the same class as i_0, that is, that the results are consistent with the classification by the DL network.

4.3.2. Evaluating the impact of dimensionality reduction

In all generality, the loss of precision incurred by the reduction in dimension is determined by evaluating some metric $m(I, D, I', D')$. In this work, the evaluation metric chosen is the *recall*. To evaluate an experiment, we compute:

$$recall(I, D, I', D') = \frac{|I \cap I'|}{K}$$

Note that this metric does not use D and D', only the indices I and I' are considered. What appears at first to be a limitation is rather a thought decision — the vectors represent images in an abstract space, in which distances or angles are not easily interpretable. Besides, recall has a simple meaning: it is the proportion of relevant neighbours that were correctly retrieved after reducing the dimension of M, regardless of the actual relevance of the original results. As such, it eases the choice of the critical parameter P' by end users and business stakeholders.

5. Results and discussion

In this section, we describe the results of the experiments that we conducted in order to validate our approach. This approach consists in

Table 1
Comparison of mean and standard deviations of accuracies per class, for L_2 and cos metrics, showing in particular the similarity between L_2 and cos. The best means are shown in bold.

	L_2		cos		IP	
Defect class	mean	std dev.	mean	std dev.	mean	std dev.
C_0	**0.886**	0.154	0.878	0.161	0.003	0.014
C_1	0.923	0.185	**0.932**	0.169	0.906	0.209
C_2	0.951	0.169	**0.956**	0.164	0.670	0.269
C_3	0.759	0.307	**0.767**	0.306	0.227	0.361
C_4	0.805	0.301	**0.806**	0.300	0.000	0.000
C_5	0.870	0.216	**0.879**	0.207	0.701	0.407
C_6	0.705	0.259	**0.743**	0.237	0.036	0.009
C_7	0.756	0.315	**0.764**	0.313	0.004	0.010
C_8	0.855	0.284	**0.863**	0.279	0.310	0.230
C_9	0.919	0.192	**0.931**	0.167	0.967	0.090

reducing the size of feature vectors, in order to improve the memory footprint and search time, without losing quality in the image retrieval process. First, we narrow the choice of the (pseudo-)distance metric d to those that are the most appropriate for our image retrieval problem. Then, and for each (pseudo-)distance metric, we identify the most appropriate dimensionality reduction algorithm, that is, the one that will have the least impact in terms of recall and the best memory footprint and search time. In Fig. 7, we summarize a decision making process grounded in the experimental results.

5.1. Distance metric analysis

In order to identify the distance metrics that better performs for this kind of problem, we run the following evaluation. We compute the accuracy of every proposed metric (L_2, IP and cos) for each class. For this step, we run 15 experiments — one per metric (3 metrics) and per fold (5 folds), without applying any reduction, on 10 evaluation images per class. This gives a sample of 1 500 accuracies, 500 accuracies per metric. The distributions of the accuracies by defect class of the reference image are plotted on Fig. 3. This result informs us on the ability of the metric to retrieve consistent results.

IP shows heterogeneous distributions across defect classes, with disastrous performance (very low accuracies) on 6 of the 10 classes.

On the other hand, L_2 and cos exhibit very similar distributions, but with minor differences in favour of cos, as shown in Table 1. The differences between L_2 and cos are explained by the vectors not being of unit norm. For those metrics, most accuracy values are encountered in the higher portion of the $[0, 1]$ interval for all defect classes. Following that observation, we deem them equally apt to be used in search operations, contrary to IP. The current version of RETINA is indeed configured for the L_2 metric. L_2 has the slight advantage over cos to not require prior normalization of the vectors for the calculations.

While this evaluation relies on the defect classes only, a closer inspection of actual search results indicates that L_2 and cos do retrieve similar sets of neighbours for a given reference image. The results slightly differ sometimes, or are retrieved in a different order, but the images themselves are often the same.

5.2. Dimensionality reduction

After the previous result, we know that IP is not performing well on this problem, so with the intention of simplifying the analysis, we decided not to take it into account in the following experiments. The problem then boils down to the educated choice of the algorithm A, size of training set $|T|$ and new dimension P', for either of L_2 or cos as (pseudo-)distance metric d. This section examines the influence of each parameter, then proposes a decision process in the form of a flowchart.

In Fig. 4, we show results of recall for L_2(upper row) and cos (bottom row), for all the four reduction algorithms proposed: linear projections (PCA, SRP), and manifold learning (Isomap, LLE), at different training sizes (except for SRP, which does not depend on $|T|$). Lines with different colours represent the reduced number of components, i.e. P'.

5.2.1. The role of training size $|T|$

Results in Fig. 4 give an overview of how algorithms behave with respect to the training size $|T|$. As SRP does not depend on $|T|$, it is excluded from this discussion.

For PCA ((a) and (e)), the size of the training set, $|T|$, has little impact on the convergence of the recall. The PCA algorithm reaches a plateau at $|T| \geq 250$, where the overall distribution of the vectors is captured, and further training will not improve the results. In our dataset, 250 elements represent 1% of the database.

For Isomap ((c), (g)), recalls appear to be increasing slowly with $|T|$.

For LLE ((d), (h)), it is difficult to tell from the chart if the algorithm reaches some kind of convergence as $|T|$ increases.

As a consequence, we have run a few more experiments, with higher training sizes, for Isomap and LLE. The results are reported in Fig. 5. The augmented charts show a slight improvement for Isomap, which eventually reaches a maximum performance. LLE, on the other hand, does not improve significantly with $|T|$.

We have not explored training sizes above 10 000 for Isomap and LLE, because we did not expect better results beyond those already high values of $|T|$. Besides, the complexity of those algorithms increases with $|T|$, making them significantly slower and more memory demanding at training time than PCA or SRP.

We conclude that, as $|T|$ increases, more information from the statistical distribution of the feature vectors is captured by the PCA and Isomap algorithms than by Isomap and LLE. This is especially true with PCA, for which only a small number is needed. LLE, on the other hand, seems insensitive to the training size $|T|$, as is SRP by design.

5.2.2. The role of the number of components P'

This value is the most important in the analysis, because our goal is to produce feature vectors that are small enough and still encode maximum information from the images. Reducing as much as possible the size of the vectors, while keeping acceptable recalls, allows an optimization in both search runtime and memory consumption.

In Figs. 4 and 5, the number of components varies from 10 to 400 for the four different algorithms and the two (pseudo-)distance metrics. As we can expect, higher values of P' yield better recalls. For algorithms other than PCA, the standard deviation of recalls also decreases with P', as shown by the shaded areas around the average lines. This is further exhibited in Table 2. Interestingly, the gain in recall decreases as more components are kept. This shows that increasing P' beyond some limit value (which depends on the algorithm and metric) will only bring marginal benefit.

Another salient observation is the similarity between the two (pseudo-)distance metrics L_2 and cos. Even if the recalls can be different (especially with PCA), the overall behaviour does not depend on the metric.

Here are the other insights from the analysis of the charts.

For PCA ((a) and (e) in Fig. 4), the minimum number $P' = 10$ already brings excellent scores, especially with the L_2 metric ($recall \approx 0.8$). Higher values ($P' \geq 50$) are visually undistinguishable from each other. The recalls are also very close to 1 for those higher values with the L_2 metric.

For SRP ((b) and (f) in Fig. 4), the algorithm eventually reaches good performance ($recall \geq 0.9$), although not as fast as with PCA. SRP is very sensitive to P'. In particular, the performance is not very good for the lowest value of P' ($recall < 0.5$).

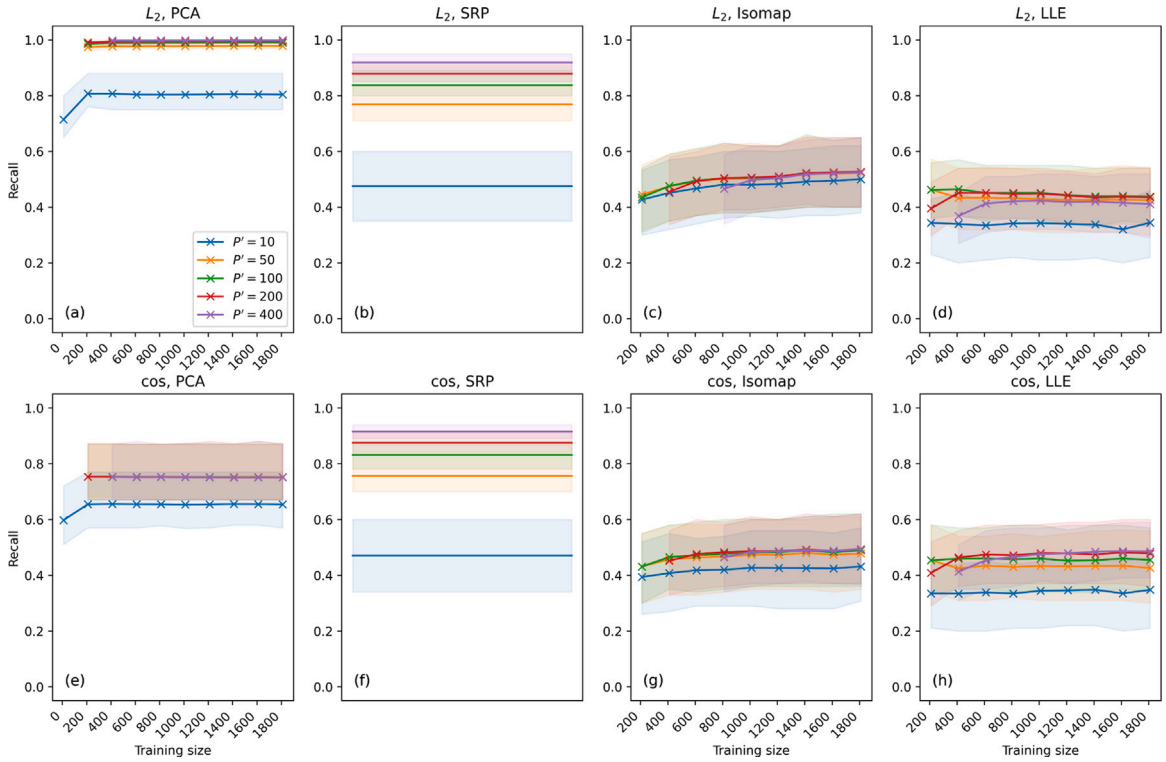

Fig. 4. Recall for PCA, Isomap, LLE and SRP algorithms, with L_2 and cos (pseudo-)distance metrics. The x-axis is the training size, except for SRP in (b) and (f), where it is irrelevant, and the colours represent the number of reduced components $P' \in \{10, 50, 100, 200, 400\}$. Solid lines represent the average recalls, while shaded areas cover the intervals between the 25th and 75th percentiles.

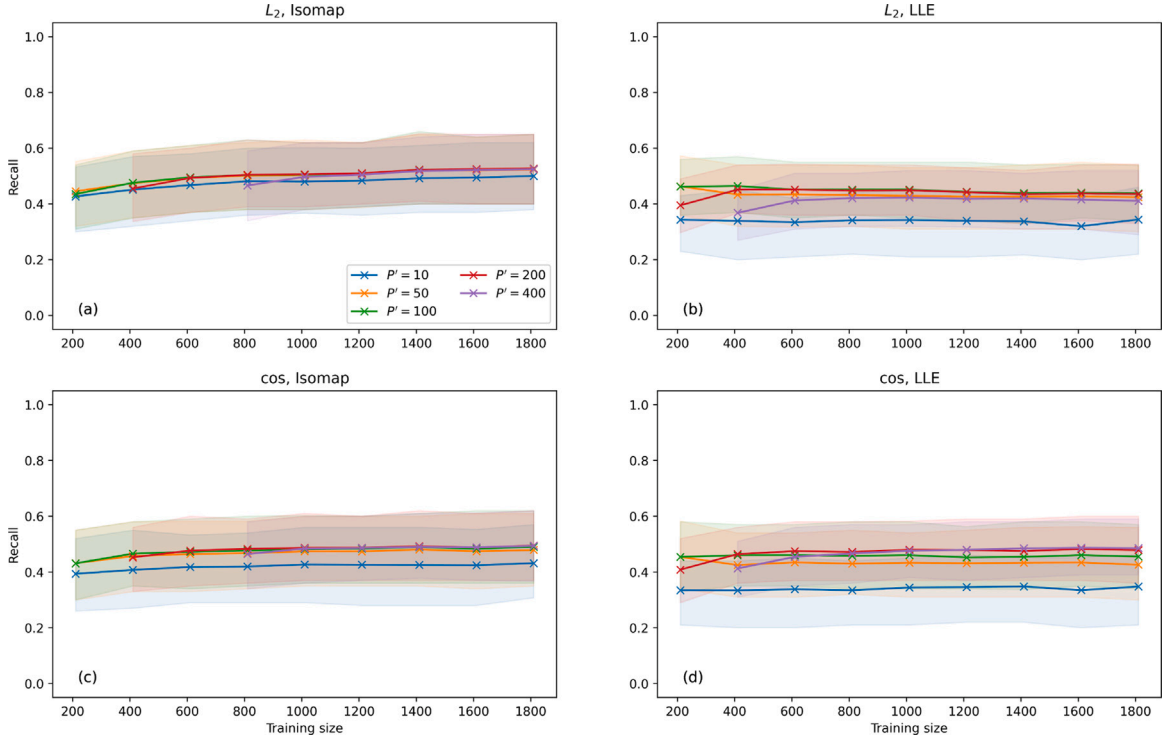

Fig. 5. Exploration of higher values of training size $|T|$, for Isomap and LLE algorithms. The experiments from Fig. 4 are reported, with two additional samples for $|T| \in \{5\,000, 10\,000\}$.

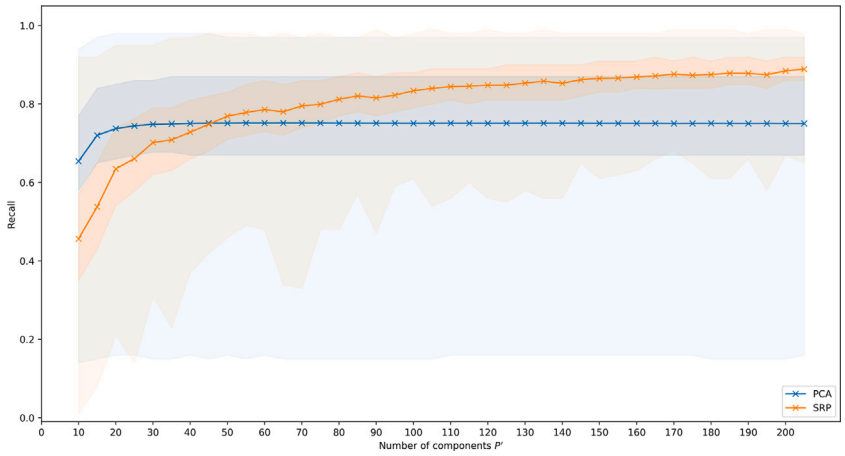

Fig. 6. Mean recalls (solid lines) for PCA and SRP for cos metric. Dark shaded areas cover the 25th-to-75th percentile intervals; light shaded areas the complete intervals. Training size for PCA is set to 2 000.

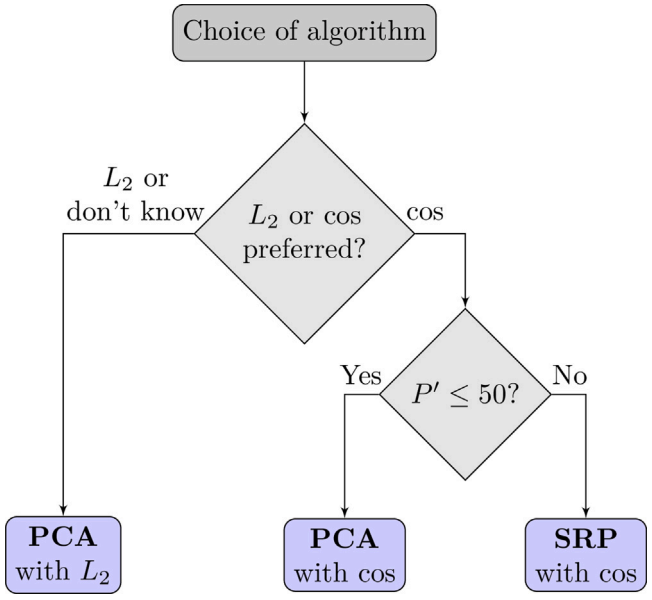

Fig. 7. Flowchart for the decision of the algorithm (PCA or SRP) depending on constraints with the metric or number of components.

For Isomap and LLE ((c), (g) and (d), (h) in Fig. 4, (a), (c) and (b), (d) in Fig. 5), the dependency on P' is small, even for the low values. The benefit of increasing P' is always marginal, but the algorithm never reaches good scores in our experiments ($recall < 0.65$).

5.2.3. The role of the (pseudo-)distance metric d

In Section 5.1, we eliminated the IP metric because of its unability to return search results consistent with the semantic grouping of the images by defect class. The remaining metrics are L_2 and cos, as reported in Figs. 4 and 5. From these figures we observe that algorithms perform differently with respect to the metric.

For PCA ((a) and (e) in Fig. 4), performance is clearly higher for L_2.

For all the other algorithms (other subplots from Figs. 4 and 5), the differences between L_2 and cos are minor and much smaller than the interquartile distances (shaded areas).

Table 2
Aggregated recalls across experiments for given pairs of algorithm A, number of components P'. In each subtable, the first column **mean** is the average recall for the biggest training size $|T|$ experimented for the algorithm (except for SRP). The second column **mean gain** is the difference in recall with the previous row, divided by the difference in P', showing that the marginal gain brought by new components decreases as P' increases. The last column **std dev.** is the standard deviation of recalls; it decreases with P', except for PCA where it slightly increases.

	PCA		
P'	mean	mean gain	std dev.
10	0.72924		0.15284
50	0.86513	0.00340	0.15833
100	0.87090	0.00012	0.16314
200	0.87350	0.00003	0.16552
400	0.87484	0.00001	0.16724
	SRP		
P'	mean	mean gain	std dev.
10	0.47356		0.17264
50	0.76249	0.00722	0.09412
100	0.83406	0.00143	0.07037
200	0.87714	0.00043	0.05275
400	0.91712	0.00020	0.03807
	Isomap		
P'	mean	mean gain	std dev.
10	0.50405		0.16915
50	0.54687	0.00107	0.14903
100	0.55478	0.00016	0.14610
200	0.55782	0.00003	0.14403
400	0.55624	−0.00001	0.14240
	LLE		
P'	mean	mean gain	std dev.
10	0.35636		0.18378
50	0.42254	0.00165	0.17724
100	0.44866	0.00052	0.16518
200	0.45751	0.00009	0.15419
400	0.43733	−0.00010	0.15484

5.2.4. The role of the algorithm A: PCA, srp, isomap and LLE

We have discussed the behaviour of the four algorithms with varying training sizes, numbers of components and (pseudo-)distance metrics. Now we summarize those insights to guide the choice of the most appropriate algorithm for the application.

First, we eliminate Isomap and LLE, since there is no situation where either of them is the best performer.

The choice between PCA and SRP, the two remaining candidates, depends on the metric used. If, as in RETINA, the metric is not dictated

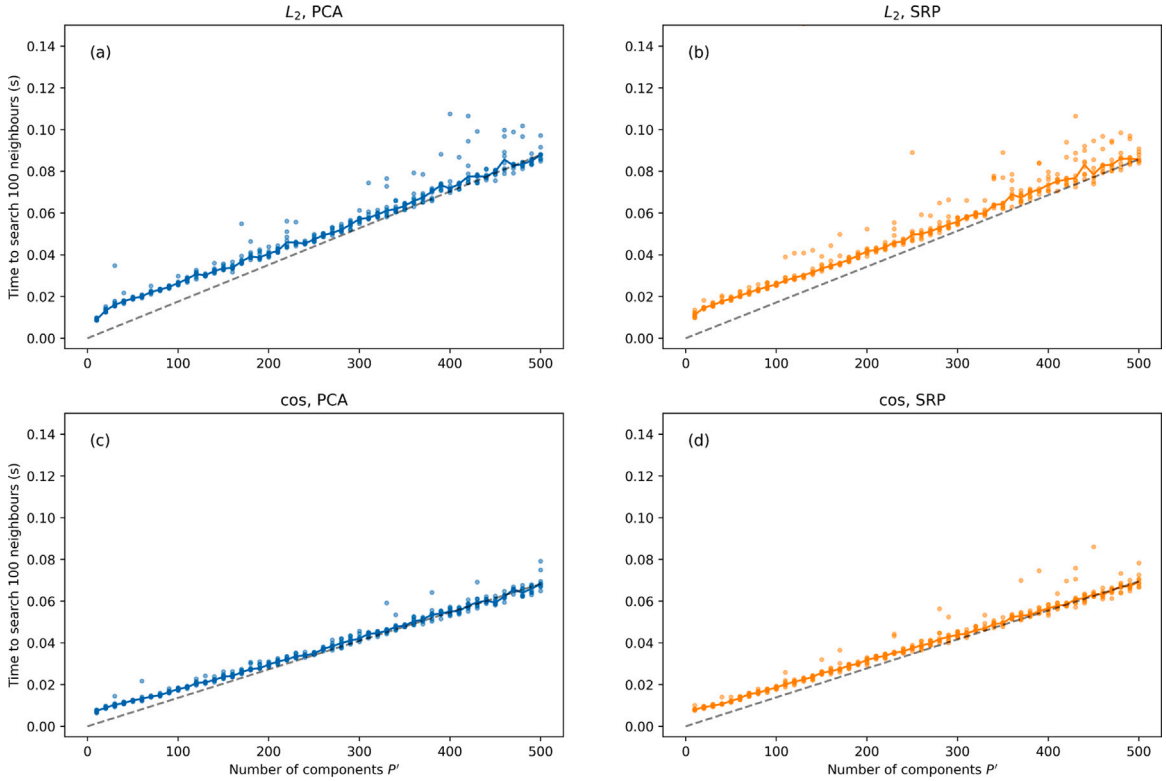

Fig. 8. Reported times to search for 100 closest neighbours of 100 evaluation vectors, for algorithms PCA ((a), (c)) and SRP ((b), (d)), and for (pseudo-)distance metrics L_2 ((a), (b)) and cos ((c), (d)), as P' varies. The dots represent one measure, the coloured lines the medians by P'. The dashed grey lines are the "ideal" response times that would be perfectly proportional to P', interpolated from 0 to the latest sample median.

by business considerations, then PCA with L_2 is the best choice. Its average recall is always higher than for the other algorithms, and its distribution is tighter, with less severe outliers. Besides, for the smallest value of $P' = 10$ (a 88% reduction of the size of the database), the average recall of PCA is already around 70%. This confirms our assumption that dimensionality reduction is a valid approach to improving the performance of image similarity search.

Should cos be preferred over L_2, then the choice of the algorithm depends on the number of components P' that one can manage in the application. Fig. 6 shows PCA (blue line) and SRP (orange line) recalls with P' values sampled at a higher rate than before. Dark shaded areas in the figure represent the 25th and 75th percentiles, light shaded ones the full range of the values encountered. This result shows that the decision threshold between the two algorithms is for $P' \gtrsim 30$. Above that value, the recalls for SRP are higher and less widespread, without extremely bad outliers. Below that threshold, PCA is slightly better on average. For our minimal number $P' = 10$, the average recall is 65% for PCA, clearly above the 46% for SRP.

Fig. 7 recapitulates the algorithm that would guide the choice between PCA and SRP.

5.3. Impact on search runtime and memory footprint

As stated in Section 3, dimensionality reduction aims to improve the time to search for images, and the memory footprint of the application. As explained previously, both quantities are expected to be proportional to the reduced dimension, P'. In this section, we report the measurements made in our experimental setup.

5.3.1. Impact on memory footprint

In order to quantify the gain in memory after reduction of the feature vector, we monitor the memory taken by the vector elements in computer memory. RETINA and the experiments are coded in *Python,* where vectors are represented as dense NumPy (Harris et al., 2020) arrays whose elements are 4-byte floating point numbers (`float32`). The byte size of an array `a` can be obtained by using the `sys.getsizeof(a)` function call. The return value of the call is the total size occupied by the array, including its elements and the metadata maintained by NumPy's internal working.

The relationship between the byte size of the full database against the number of components P' is linear, meaning that the array metadata is constant and independent of the shape of the array. The byte size is indeed given by the following formula: $byte\ size = 110\,780 \times P' + 120$. 11780 is the size occupied by $N = 27\,695$ floats, and 120 is the constant overhead of metadata. It makes the memory footprint fully predictable for a given (N, P') pair.

5.3.2. Impact on search runtime

Concerning the search runtime, we measure the time to perform the nearest-neighbour search, with the function described in Section 4 (Algorithm 1). It is worth mentioning that we do not consider the time to apply trained algorithm A on evaluation vectors E, or the database M — in RETINA, search acts on vectors that were previously reduced by an offline operation. The time measured reflects the minimal cost of a search in the conditions of our experiments, namely: $K = 100$ nearest neighbours of $|E| = 100$ evaluation vectors.

With the aims to imitate the application in production, we did not follow the same experimental loop as in Algorithm 1. We applied the dimensionality reduction algorithms in a preliminary step, before running all the searches on already reduced vectors[4]. Also, we used more folds to iron out possible variations incurred by background

[4] This is not fully representative of the production workload of RETINA, though. The search engine does more than just matching vectors: data management, filtering by process data, user interaction, etc. Pre- and post-processing add their own overhead, and can have negative side-effects on the search

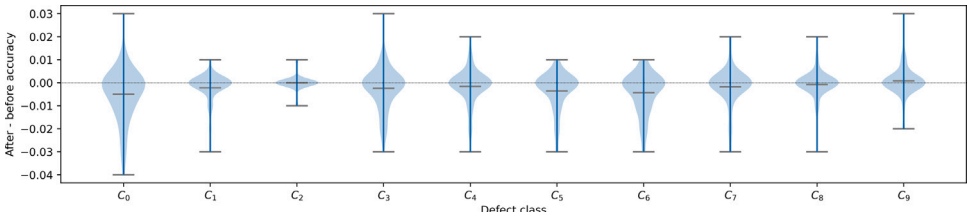

Fig. 9. Difference in accuracies (*after reduction* - *before reduction*) for all defect classes.

activity of the operating system. We focused on vectors reduced by PCA and SRP only, expecting no dependency on the algorithm. The experiments were conducted on a virtual server with four 2.3 GHz virtual cores (CPU only) and 15 GB of RAM.

In Fig. 8, we present the search runtime measured for PCA ((a), (c)) and SRP ((b), (d)). The choice of the (pseudo-)distance metric L_2 or cos is also taken into account, L_2 being on the top row and cos on the bottom row. As expected, the runtime is independent from the type of algorithm reduction. Overall, we find a proportionality between the search time and P' for both algorithms. At low component numbers, a deviation from a pure linear relationship is visible in both metrics, being more significant for L_2 than cos. However, we estimate that this deviation does not significantly affect the overall performance of the reduction method.

We also see that calculations with the cos metric are slightly faster than those with L_2. This fact is not surprising, because L_2 involves more arithmetic operations for each distance evaluated[5].

5.4. Results for current RETINA configuration

After having validated in the previous sections that PCA with L_2 similarity is the best overall combination, in this part we focus on current results for RETINA running on PCA with L_2 and $P' = 32$. The training size is set to $|T| = 2\ 000$ (7.2% of the database), well above the beginning of the plateau from Fig. 4.

5.4.1. Accuracy and recall

In Fig. 9, we plot the difference in the accuracy before and after applying the reduction method on the feature vector for all the defect classes. As we expected, the difference in accuracy lies around zero in the $[-0.08, +0.04]$ range for the ten defect classes. No particular class stands out as being more impacted than the others on average by the reduction.

In terms of *recall*, the distribution of the values in Fig. 10 shows that despite the significant reduction in dimensions, from 512 to 32, the impact on retrieved neighbours is very small. The average recall is over 0.96, meaning that 96 of the $K = 100$ images that would have been retrieved without reduction are present in the actual results with a feature vector size of 32 components.

5.4.2. Memory and search runtime

The initial memory size, corresponding to the raw vectors, was estimated as 38 GiB in Section 3. With $P' = 32$ and $N = 2 \times 10^6$, we obtain a revised size of 2.4 GiB, to be compared with the initial 38 GiB of the raw vectors. This is a 16-fold improvement.

Regarding the search time, whose measurements are sensitive to the infrastructure being used, we only report the theoretical improvement that would be observed on our test server (described in Section 5.3).

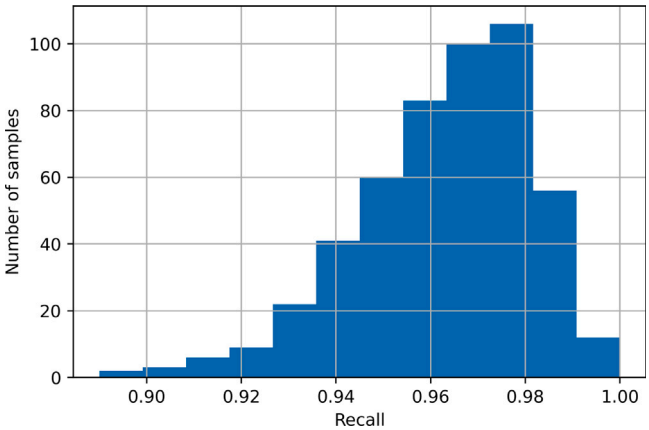

Fig. 10. Distribution of recalls for PCA with $P' = 32$ and the L_2 metric, which is the RETINA configuration.

The initial cost ($P' = P = 512$) of searching for 100 neighbours with L_2 would be around 0.091 s, and that figure would decrease to 0.015 s with $P' = 32$. It is a 6-times improvement only, less than the 16-fold gain in memory consumption. The offset of the curves in Fig. 8, especially significant for low values of P', explain this discrepancy.

6. Conclusions

In this paper, we formulated the performance of a CBIR system as a trade-off between relevance of results, search time and memory footprint. In the absence of definitive classification labels, and using classes from the upstream DL model as reference, we first used accuracy to select the similarity metrics that are most likely to identify nearest neighbours from the same class as the reference image.

Then, we envisioned dimensionality reduction of the feature vectors as a means to improve search time and memory footprint at once. We proposed to use recall as an interpretable measure for relevance of search results after applying the reduction.

Through a systematic exploration of reduction algorithms and their parameters (similarity metric d, number of components P', training size $|T|$), we eventually identified our trade-off for RETINA, i.e. a PCA algorithm with a reduced number of components $P' = 32$, using L_2 as the metric for searching the neighbours. That combination leads to significant improvements: after training the PCA with 7.2% of the images, the memory footprint is divided by a factor of 16, the search time by 6, while maintaining an average recall of 0.96 for 100 neighbours. In the case of RETINA, given the volume of images managed online by the application, those improvements let us relieve the pressure on the hardware infrastructure considerably.

In this first study, we experimented with conventional reduction algorithms that showed good performances in the semiconductor domain. However, the protocol applied could include other algorithms that were not considered in this work, or other vector compression techniques such as quantization (implemented by FAISS), which have similar

itself *via* their effect on the CPU caches. Those effects are not evaluated in this study. They could be avoided by loading the database into GPU memory, where calculations are fast and not impaired by the side activity of the CPU.

[5] FAISS does not compute the square roots, the neighbours are actually searched by *squared* distances.

benefits as the dimensionality reduction. By acting on the vectors and not on the network, the classification task would remain unaffected.

Although our approach explicitly avoids tampering with the network, which supports a critical online defect classification task, the 16-fold reduction in number of dimensions suggests that its weights are redundant in some way. Another approach could be to optimize it while maintaining its accuracy as a classifier.

CRediT authorship contribution statement

Thomas Vial: Conceptualization, Methodology, Software, Formal analysis, Investigation, Writing – original draft, Visualization. **Farah Dhouib:** Conceptualization, Methodology, Software, Investigation. **Louison Roger:** Software. **Annabelle Blangero:** Conceptualization, Methodology, Supervision, Review & editing. **Frédéric Duvivier:** Funding acquisition, Review & editing. **Karim Sayadi:** Supervision, Project administration, Funding acquisition, Review & editing. **Marisa N. Faraggi:** Conceptualization, Methodology, Validation, Formal analysis, Writing – original draft, Visualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors wish to thank STMicroelectronics Rousset for providing the images and the DL model used in this study, and for reviewing the work.

This project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement no. 826589. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Netherlands, Belgium, Germany, France, Italy, Austria, Hungary, Romania, Sweden, and Israel.

Appendix. Configuration of the algorithms

Here we detail how we have chosen the hyperparameters of each dimensionality reduction algorithm A. In addition to the desired number of components, each algorithm has its own set of parameters that control its training procedure. We used Scikit-Learn version 1.0[6].

For PCA, the default parameters of class `sklearn.decomposition.PCA` were used.

For SRP, the default parameters of class `sklearn.random_projection.SparseRandomProjection` were also used.

For Isomap, the class is `sklearn.manifold.Isomap`. We set the number of neighbours (`n_neighbors`) to 7 for the training step, instead of the default 5, to match the setting used in Cheng et al. (2013). Also, we set the metric for the nearest neighbour-search (`metric`) according to the (pseudo-)distance metric d of the experiment.

For LLE, the base class is `sklearn.manifold.LocallyLinearEmbedding`. We extended it to allow changing the metric for the nearest-neighbour search, and used the same parameters as for Isomap (i.e. `n_neighbors` = 7 and `metric` = d); for the other parameters the default values are used.

[6] The full API, showing the default values of parameters, is available at the following address: https://scikit-learn.org/1.0/modules/classes.html (accessed 10 May 2022).

References

Arai, H., Chayama, Y., Iyatomi, H., Oishi, K., 2018. Significant dimension reduction of 3D brain MRI using 3D convolutional autoencoders. In: 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society. EMBC, IEEE, Honolulu, HI, pp. 5162–5165. http://dx.doi.org/10.1109/EMBC.2018.8513469.

Babenko, A., Slesarev, A., Chigorin, A., Lempitsky, V., 2014. Neural codes for image retrieval. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (Eds.), Computer Vision – ECCV 2014. In: Lecture Notes in Computer Science, Springer International Publishing, Cham, pp. 584–599. http://dx.doi.org/10.1007/978-3-319-10590-1_38.

Bingham, E., Mannila, H., 2001. Random projection in dimensionality reduction: applications to image and text data. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '01, Association for Computing Machinery, New York, NY, USA, pp. 245–250. http://dx.doi.org/10.1145/502512.502546.

Brauer, H., Ziolkowski, M., Toepfer, H., 2014. Defect detection in conducting materials using eddy current testing techniques. Serbian J. Electr. Eng. 11 (4), 535–549. http://dx.doi.org/10.2298/SJEE1404535B.

Cao, Z., Mu, S., Xu, Y., Dong, M., 2018. Image retrieval method based on CNN and dimension reduction. In: 2018 International Conference on Security, Pattern Analysis, and Cybernetics. SPAC, IEEE, pp. 441–445. http://dx.doi.org/10.1109/SPAC46244.2018.8965601.

Cheng, B., Zhuo, L., Zhang, J., 2013. Comparative study on dimensionality reduction in large-scale image retrieval. In: 2013 IEEE International Symposium on Multimedia. pp. 445–450. http://dx.doi.org/10.1109/ISM.2013.86.

ElMaraghy, H.A., Bullis, D.J., 1989. Expert inspector of surface defects. Comput. Ind. 11 (4), 321–331. http://dx.doi.org/10.1016/0166-3615(89)90131-0.

Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., 2014. Computer Vision – ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I. Springer, Google-Books-ID l4BHBAAAQBAJ.

Girshick, R., 2015. Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1440–1448.

Girshick, R., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 580–587.

Haleem, N., Bustreo, M., Del Bue, A., 2021. A computer vision based online quality control system for textile yarns. Comput. Ind. 133, 103550. http://dx.doi.org/10.1016/j.compind.2021.103550.

Hameed, I.M., Abdulhussain, S.H., Mahmmod, B.M., 2021. Content-based image retrieval: A review of recent trends. In: Pham, D.T. (Ed.), Cogent Eng. 8 (1), 1927469. http://dx.doi.org/10.1080/23311916.2021.1927469, Publisher: Cogent OA.

Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N.J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M.H., Brett, M., Haldane, A., del Río, J.F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., Oliphant, T.E., 2020. Array programming with NumPy. Nature 585 (7825), 357–362. http://dx.doi.org/10.1038/s41586-020-2649-2, Number: 7825, Publisher: Nature Publishing Group.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778.

Hong-Seok, P., Mani, T.U., 2014. Development of an inspection system for defect detection in pressed parts using laser scanned data. 24th DAAAM International Symposium on Intelligent Manufacturing and Automation, 2013, Procedia Eng. 24th DAAAM International Symposium on Intelligent Manufacturing and Automation, 2013, 69, 931–936.http://dx.doi.org/10.1016/j.proeng.2014.03.072,

Jenni, K., Mandala, S., Sunar, M.S., 2015. Content based image retrieval using colour strings comparison. Big Data, Cloud and Computing Challenges, Procedia Comput. Sci. Big Data, Cloud and Computing Challenges, 50, 374–379.http://dx.doi.org/10.1016/j.procs.2015.04.032,

Johnson, J., Douze, M., Jégou, H., 2019. Billion-scale similarity search with GPUs. IEEE Trans. Big Data 7 (3), 535–547.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A.C., 2016. SSD: Single shot MultiBox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (Eds.), Computer Vision – ECCV 2016. In: Lecture Notes in Computer Science, Springer International Publishing, Cham, pp. 21–37. http://dx.doi.org/10.1007/978-3-319-46448-0_2.

MADEin4, 2019. MADEin4 - metrology advances for digitized electronic components and systems (ECS) industry 4.0. URL https://madein4.eu/.

Miao, R., Gao, Y., Ge, L., Jiang, Z., Zhang, J., 2019. Online defect recognition of narrow overlap weld based on two-stage recognition model combining continuous wavelet transform and convolutional neural network. Comput. Ind. 112, 103115. http://dx.doi.org/10.1016/j.compind.2019.07.005.

Murthy, C.B., Hashmi, M.F., Bokde, N.D., Geem, Z.W., 2020. Investigations of object detection in images/videos using various deep learning techniques and embedded platforms—A comprehensive review. Appl. Sci. 10 (9), 3280. http://dx.doi.org/10.3390/app10093280, Number: 9, Publisher: Multidisciplinary Digital Publishing Institute.

Pearson, K., 1901. LIII. On lines and planes of closest fit to systems of points in space. Lond. Edinb. Dublin Philos. Mag. J. Sci. 2 (11), 559–572.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in python. J. Mach. Learn. Res. 12, 2825–2830.

Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 779–788.

Roweis, S.T., Saul, L.K., 2000. Nonlinear dimensionality reduction by locally linear embedding. Science 290 (5500), 2323–2326.

Tenenbaum, J.B., Silva, V.d., Langford, J.C., 2000. A global geometric framework for nonlinear dimensionality reduction. Science 290 (5500), 2319–2323.

Tulbure, A.-A., Tulbure, A.-A., Dulf, E.-H., 2022. A review on modern defect detection models using DCNNs – deep convolutional neural networks. J. Adv. Res. 35, 33–48. http://dx.doi.org/10.1016/j.jare.2021.03.015.

Vita, G., Kamboj, P., 2016. Content based image retrieval system: Review. Int. J. Comput. Trends Technol. 34, 129–133. http://dx.doi.org/10.14445/22312803/IJCTT-V34P123.

Yandex, A.B., Lempitsky, V., 2015. Aggregating local deep features for image retrieval. In: 2015 IEEE International Conference on Computer Vision. ICCV, IEEE, Santiago, Chile, pp. 1269–1277. http://dx.doi.org/10.1109/ICCV.2015.150.

Zhang, N., Shamey, R., Xiang, J., Pan, R., Gao, W., 2022. A novel image retrieval strategy based on transfer learning and hand-crafted features for wool fabric. Expert Syst. Appl. 191, 116229. http://dx.doi.org/10.1016/j.eswa.2021.116229.

Zhang, J., Wang, H., Tian, Y., Liu, K., 2020. An accurate fuzzy measure-based detection method for various types of defects on strip steel surfaces. Comput. Ind. 122, 103231. http://dx.doi.org/10.1016/j.compind.2020.103231.