EUROGRAPHICS Workshop on Graphics and Cultural Heritage (2021)
A. Chalmers and V. Hulusic (Editors)

Automatic segmentation of archaeological fragments with relief patterns using convolutional neural networks

*Elia Moscoso Thompson[1], *Andrea Ranieri[1] and Silvia Biasotti[1]

[1]Istituto di Matematica Applicata e Tecnologie Informatiche 'E. Magenes' - CNR
*These authors contributed equally to this work

Abstract

The recent commodification of high-quality 3D scanners is leading to the possibility of capturing models of archaeological finds and automatically recognizing their surface reliefs. We present our advancements in this field using Convolutional Neural Networks (CNNs) to segment and classify the region around a vertex in a robust way. The network is trained with high-resolution views of the 3D models captured at different angles. The views represent both the model with its original textures and a colorization of the patches according to the value of the Shape Index (SI) in their vertices. The SI encodes local surface variations and we exploit the colorization of the model driven by the SI to generate other view and enrich the dataset. Our method has been validated on a relief recognition benchmark on archaeological fragments proposed within the SHape REtrieval Contest (SHREC) 2018.

CCS Concepts

• ***Computer systems organization*** → ***Neural networks;*** • ***Computing methodologies*** → ***Shape analysis;***

1. Introduction

Many manufactured objects possess repeating elements [WLKT09] that significantly affect their type, material, and style. For instance, the recognition of motifs and decorations on the surface of an artifact can contribute significantly to the work of the archaeologist by automatically supporting the classification and annotation of archaeological fragments [ZPS*16, DTC*17], even if scattered in various collections [MTBS*18]. Tackling this problem is challenging because this task demands the analysis of details that are present only on high-resolution models. Since this kind of models can become very large, local and very efficient algorithms are needed to efficiently perform this analysis. In our terminology, we distinguish the two concepts of retrieval and recognition. Given a query model and a collection of surfaces, with the term *relief retrieval* we are interested to find the surfaces in the dataset characterized by the same relief of the query, while with the term *relief recognition* we are interested to identify known reliefs on each surface and to localize them on the surface. The two concepts are linked, and relief retrieval can be viewed as a step within a recognition framework. However, while relief retrieval has been object of interest for a while and counts multiple contributions (for instance [WTBdB15, WBB15, MB18, Gia18]) and there are some recent benchmarks, like [BMTA*17, MBG*20] less has been done for recognition [BMTB*18].

Indeed, previous works on local similarity analysis [DCV14, GK15, RBFB13] focus primarily on the concept of self-similarity rather than pattern recognition or segment the surface by looking primarily at the structure of the model rather than the reliefs imprinted on its surface [HSL*17, WJHM15].

In this paper we are exploring the possibility of segmenting reliefs of repeated motifs and decorations on the surface of an archaeological artefact using supervised learning to obtain the classification of the individual pixels of images extrapolated from 3D models. While most of the methods in the literature tend to isolate a single pattern from the rest of the model and then to recognize it, our goal is to segment the whole surface *at once*, thus also tackling composite patterns, without the need for any prior knowledge of the artifact. Figure 1 shows examples of fragments whose surface is covered by a single pattern or a composite one.

Our approach is based on sampling a surface model with a sufficient number of disk-like patches, then convert these patches into images, use these images to feed a Convolutional Neural Network (CNN) that robustly segments and classifies these images, and finally, to bring back on the surface the segmented images (see the graphical summary in Figure 2). Both image segmentation and classification are done by training a DeepLabv3+ [CZP*18] network with a pre-trained ResNet-152 [HZRS15] as backbone.

To enrich the dataset, in addition to images of the textured object, we create synthetic images of the patches. These images are derived from the so-called Shape Index [Kv92], which locally characterizes the surface variation in each point with a scale-invariant scalar value that ranges from -1 to 1. It is possible to think of this

DOI: 10.2312/gch.20211411

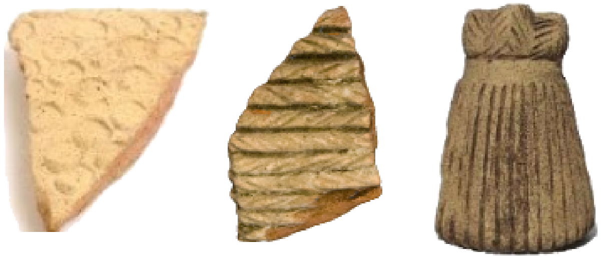

Figure 1: *From left to right: two examples of fragments characterized by a single pattern and an example of composite relief pattern (rightmost model).*

new set of images as an "offline and geometry-aware data augmentation" of the original images. All these images (those with real and synthetic textures) make up the overall dataset and are subject to normal data augmentation during training.

We validate our method on the benchmark of the SHREC'18 contest on surface relief (or pattern) recognition [BMTB*18]. Initial results show that approaching the relief recognition problem in this way is feasible and already generates promising results.

This paper is organized as follows. Section 2 briefly overviews the works relevant to frame our work in the current state of art. Section 3 introduces our method, describing both the image extraction and the learning setup and training, and the dataset on which our method is tested. Experimental settings and results are detailed in Section 4. Finally, the conclusive remarks and future perspectives are given in Section 5.

2. Related work

When dealing with the recognition of relief variations on a surface, a common strategy is to reduce the data dimension by projecting the 3D data onto a proper plane (thus creating an image) and apply an image pattern recognition algorithm to the projected data. For instance, we mention [OVSP13] for tree species classification and its adoption to the classification of relief and engraves of rock artifacts [ZPS*16] and the classification of archaeological fragments [DTC*17]. In [OVSP13] the geometric variations of the surfaces are represented by a 3D deviation map over a cylinder, which is flattened on a plane using the Principal Component Analysis (PCA) technique. In [ZPS*16] a height map directly projects the relief onto the plane. Then, the resulting images are compared using the complex wavelet technique on images. Similarly [DTC*17] used a depth map of the face on which the motif is engraved to highlight the local variance of the relief patterns transformed the variance map into a binary image that is classified using a multi-class SVM model. The projection on a plane or a cylinder can be too restrictive if the curvature of the surface underlying the relief is bumpy and irregularly embedded. To address the characterization of relief applied to bumpy surfaces, [Gia18] proposed to re-parameterize a surface using geodesic disks and then to embed the disk into an image. Then, the problem is addressed with a statistical texture classification method, using SIFT descriptors and Fisher Vectors classifiers.

A common strategy of these approaches is to relate the relief pattern recognition to a texture classification problem: indeed, second order statistics, filter banks, local binary patterns, pooling of point descriptors, e.g. SIFT+Fisher Vectors [CMK*14], etc. are some of the techniques successfully developed in the pre Deep Learning era of Computer Vision to tackle this kind of problems. With the advent of CNNs, the use of data-driven features extractors outperformed hand-crafted features on classic benchmarks [KSH12, CMKV16]. Also in the context of Cultural Heritage, image-based techniques have been recently used to automatically classify ceramic potsherds according suggested matches from a comparative collection with typical pottery types and characteristics [ADD*21]. This led us to consider CNNs, and in this case semantic segmentation architectures such as DeepLabv3+ [CZP*18], to address this type of problem in order to overcome the limitation of the existing approaches of being able to deal only with single patterns.

3. Method and data description

We divide our method into two steps: the creation of a collection of images (Section 3.1) and a brief description of the architecture we use (Section 3.2). In the following we detail both these steps and then, in Section 3.3, we briefly describe the data we used to test our approach.

3.1. Local feature characterization and extractor

Our strategy for the generation of the surface images is to get a sufficient and complete set of images of the object surface. For this task, we consider a surface covering made of N_S disks of radius R. The choice of the radius R determines the size of the lens we use to analyse the surface. As a general criterion, we choose R large enough to allow N_s to encapsulate at least a significant portion of the decoration we are interested in. To keep our pipeline as simple as possible, we expect the models in the dataset to be all in the same scale. We show our choice of R for our use-case in Section 3.3.

Then, the number of disks N_S is determined as $N_S = \lfloor 1.2 \frac{A}{\pi R^2} \rceil$, where A is the area of the model surface. Note that we admit (possible) disk overlaps in order to be reasonably sure to cover the whole surface. Moreover, we use the amplification factor 1.2 in the estimation of N_S. Given a model M represented with a triangle mesh T, the centers of the N_S disks are determined using a Poisson-disk sampling [CCS12] (Figure 3(a)). Given a sample s, the kd-tree algorithm [FBF77] is used to compute the neighbor of s of radius R (see Figure 3(b)), defined as $N(s) = \{v \in V | d(s,v) \leq R\}$, where d is the Euclidean distance. We then consider the triangles that have at least one of the vertices in $N(s)$ and we keep only the connected component of triangles that contain the sample s. We name such a region of triangles (or rather, such a patch) $\mathcal{P}$. Then, we center $\mathcal{P}$ in the origin of the Cartesian coordinate system and we align it to the xy-plane. This is done by computing the best fitting plane π for the points in $N(s)$, computed via Least-Squares w.r.t. the family of planes in $\mathbb{R}^3$. In Figure 3(c) we show a profile with a patch and its best fitting plane.

The patch $\mathcal{P}$ is then colorized. We consider two different types of colorization: one derived from the real texture and obtained from

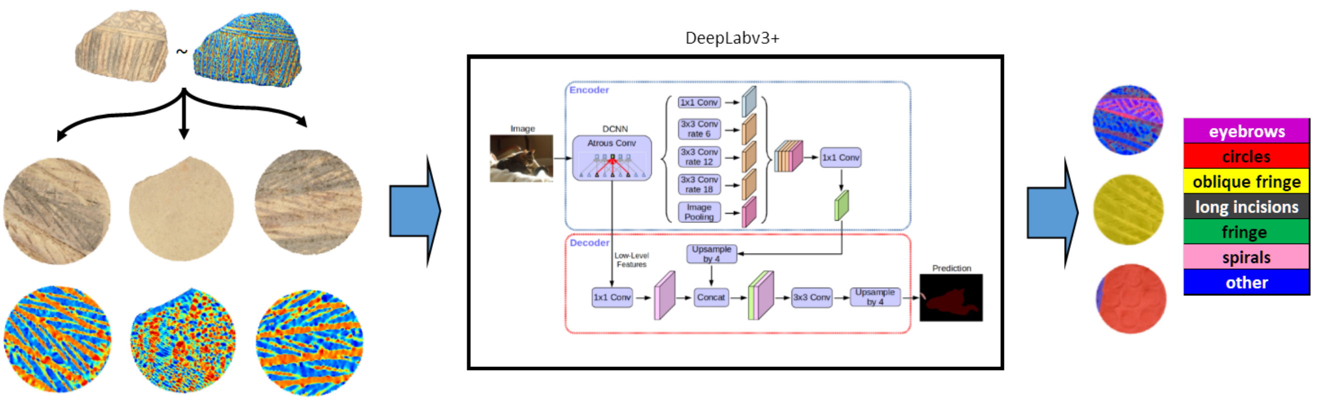

Figure 2: *Method overview: we extract disk-like patches from the surface model, then we use these images to feed a CNN that robustly segments and classifies them. Both segmentation and classification are done by training a DeepLabv3+ network. The DeepLabv3+ representation is taken from [CZP*18].*

the RGB values of the scattered data, the other based on a synthetic colorization of the surface according to a geometric property. As the surface property, we have selected the Shape Index [Kv92], which is a single-value local property derived from the surface principal curvatures and is strictly related to the perception [TP97] of scale-independent surface characteristics. The Shape Index was already used in [MTBS*18] to locally characterize the surface patterns and in [BMS19] for similarity reasoning based on local surface variation. Principal curvatures are estimated on T via the Matlab Toolbox Graph [Pey] that in [MTB19] was experimentally shown to be a curvature estimation tool well suited for characterizing local relief variations. This process consists in taking a picture of $\mathcal{P}$ as follows: we flatten the patch onto the xy-plane and then, we place the camera point of view along the positive z-axis, pointed toward the origin of the Cartesian coordinate system. The surface colourizations with respect to RGB and SI values are shown in Figure 3(d) and Figure 3(e), respectively. Lighting, in the latter case, is turned off to avoid noises and/or uncertainty caused by shadows or highlights, while the values are represented via jet color-map.

3.2. Deep learning environment

To evaluate network performance under different conditions, we train the same architecture with three different patch radii ($R = 1.25, 2, 3$) and with three different data augmentation regimes: 1) no data augmentation, 2) basic data augmentation, 3) strong data augmentation. With $R = 2$, we also train the network with only half of the dataset (i.e. RGB images only or SI images only) as a further ablation study. Except for the "no data augmentation" regime, all images, both those with "natural" textures and those with Shape Index textures, are subjected to data augmentation to prevent overfitting on an architecture with such a high model capacity. See Section 4 for details on the data augmentation setups used.

When performing transfer learning from a pre-trained network, the training of a segmentation network is usually divided into two phases, *a - frozen* = backbone layers are frozen, only the head (i.e. decoder/upscaling network) is trained and *b - unfrozen* = all layers are trained, layer groups are trained with learning rate increasing with the depth of the network. For each epoch, all the training set is fed into the network, one batch at a time, with each image in the batch being altered by a random number of data augmentation transformations of different intensity or magnitude.

3.3. Dataset description

In our experiments we used models and the ground-truth available in the benchmark of the SHREC'18 contest on relief recognition [BMTB*18]. These models were derived from one of the use cases of the GRAVITATE EU Project [UUTCIC*] and correspond to objects in attributed to the Neo-Cypriote style, dating from the second half of the VII century BC to the early VI century BC [MTW91, Kar93]. Each model is represented as a triangle mesh, equipped with colorimetric information on the vertices. The set of models contains 30 models, characterized by none, one, or more than one reliefs. The triangle meshes possess a very high precision over the small details (which is a key factor for the analysis of the decorative elements) and their number of vertices range from 150K to 6.8M. Figure 4 (top) shows examples of the models. The models are annotated with labels on the mesh triangles. For each model, triangles characterized by the same pattern have same label. Triangles that do not belong to any pattern (like eyes, mouth or smooth areas) are labeled as *no pattern* (or *NoPatt* for short). This is additional label is mandatory because in our environment the concept of "unlabeled" is not defined, thus we use one label to identify everything on the surface but patterns.

Figure 4 (bottom) shows examples of the segmentation and classification provided by the benchmark. Different colors represent different patterns.

Given the size of models, the parameter R for our sampling method was initially set to 2 as a compromise between number of sampling images and and size of the disk. As a further study, we also train the network with sampled patches with a radius of $R = 1.25$ and $R = 3$ to observe any performance improvements or degradation by providing more or less "context" to the network. A

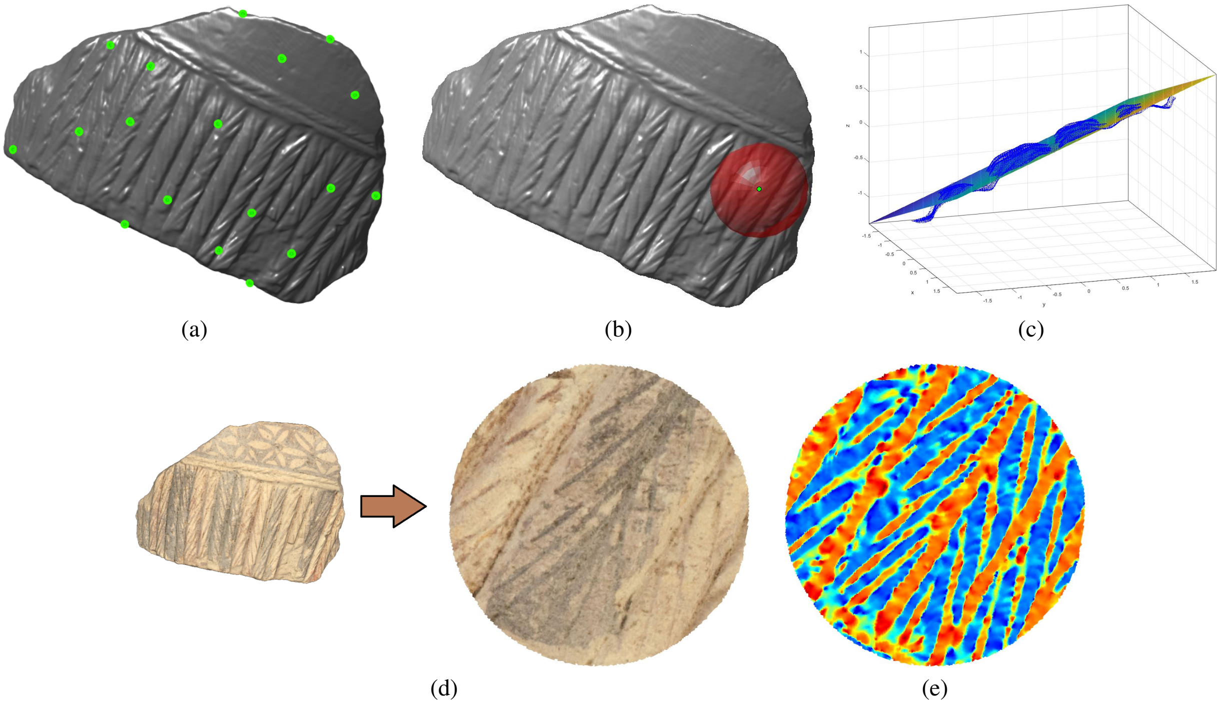

Figure 3: *Overview of the image sampling part of the method. (a): a model and its sampling $s_1, \dots, s_{N_s}$ done with the Poisson-disk sampling method; (b): the neighborhood (in red) of a sample s_i (in green); (c): the best fitting plane to the vertices in the neighborhood which determines the rotation of the patch; (d): the patch colorized based on the texture of the original object (in the left); (e): the same patch colorized using the shape index estimated with [Pey].*

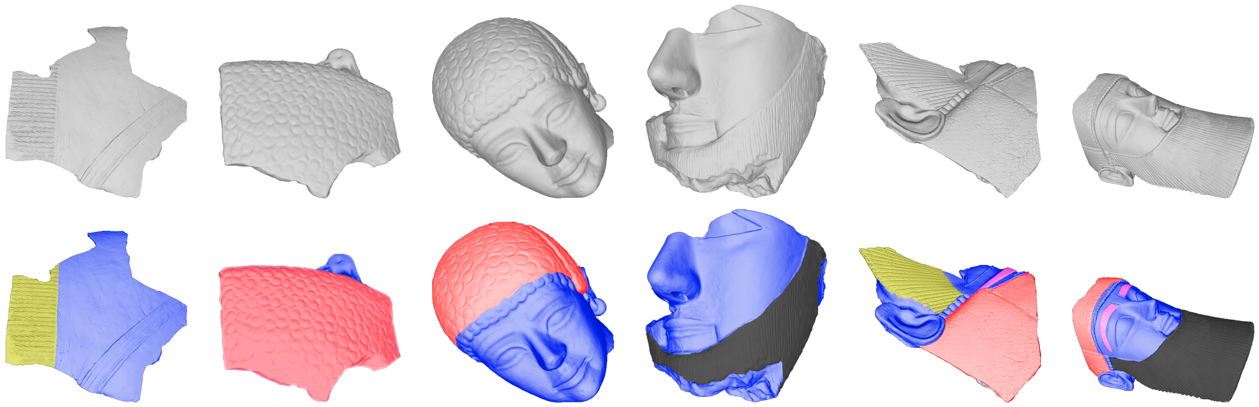

Figure 4: *Top: A sub-sample of the Model Set of the SHREC'18 benchmark on pattern recognition. For an overview of all the models, please refer to [BMTB*18]. Bottom: the same sub-sample provided with their ground-truth. Different colors represent different patterns. Yellow is* oblique fringe, *red is* circlets, *magenta is* eyebrows, *black is* long incisions, *green is* fringe *and pink is* spirals. *Blue is used to cover those part of the patches that are not patterns (*NoPatt, *see Section 3.3).*

visual representation of a sphere of radius 2 (in red) over a model is shown in 3(b).

4. Experimental setup and results

We briefly describe our learning setup in Section 4.1. We trained the same architecture with images obtained with different settings: three different patch radii: 1.25, 2 and 3 and using only patches with RGB textures or only colored with the Shape Index. Each of these architectures was trained in three different data augmentation regimes as described in Section 4.2.

4.1. Setup

The training was performed using Python code inside Jupyter Notebook, leveraging the popular Fast.ai [HG20] deep learning library built on PyTorch. The hardware used was a GPU node of the high-performance EOS cluster located within the University of Pavia. Each of those GPU nodes has a dual Intel Xeon Gold 6130 processor (16 cores, 32 threads each, for a total of 64 virtual processors per node) with 128 GB RAM and 2 Nvidia V100 GPUs with 32 GB of video RAM.

From the dataset described in Section 3.3 we extracted 1821 images with real texture and as many images with the synthetic ones. The training was performed starting from these 3642 high resolution RGB+SI images, the latter rendered by our local feature extractor, at 1585x1585 px. We chose to split the dataset randomly into training and validation sets using a 70/30 split. We perform the training using a batch size equal to 8.

Fast.ai convenient APIs allow to download the pre-trained backbone and weights from the Torchvision model zoo in a very simple and automatic way. *Fast.ai* also automatically modifies the DeepLabv3+ architecture so that the number of neurons in the output layer of the decoder network corresponds to the number of classes of the current problem, initializing the new layers with random weights.

In order to maximize the level of automation during the training of the network, a few Fast.ai callbacks were used to perform the early stopping of the training (with $patience = 10$, i.e. 10 epochs without improving the validation loss of the network) and to save the best model of the training round and then reload it for validation. Learning rates has been set to *slice(1e-04, 1e-03)* and *slice(1e-07, 1e-06)* for the frozen and unfrozen steps respectively.

4.2. Data Augmentation & Training

Except for the "no data augmentation" regime, the other two regimes both use this set of data augmentations: *Blur, CLAHE, GridDistortion, OpticalDistortion, RandomRotate90, ShiftScaleRotate, Transpose*. Below is a more detailed description of the data augmentation setups used:

- **No data augmentation** - images are left undistorted and are fed to the network "as-is" (*no-data-aug* in Table 1).
- **Basic data augmentation** - in addition to the transformations listed above, images can also receive the following transformations: *ElasticTransform, HorizontalFlip, HueSaturationValue* (*basic-data-aug* in Table 1)
- **Strong data augmentation** - in addition to the transformations listed above, images can also receive the following transformations: *Emboss, Flip, GaussNoise, MedianBlur, MotionBlur, PiecewiseAffine, RandomBrightness, RandomContrast, Sharpen* (*strong-data-aug* in Table 1)

For each of the scenarios (*no-data-aug*, *basic-data-aug*, *strong-data-aug*), the training was limited to 100+100 (freeze+unfreeze) epochs, but ended for early stopping after the number of epochs shown in Table 1.

The number of epochs in which the network learns (i.e. the validation loss steadily decreases) is a further indicator of the quality of the chosen data augmentation transformations: if the data augmentation is sub-optimal, the network will stop learning before the same network trained on the same data with the same hyperparameters but in a scenario with a better data augmentation.

Once all the training rounds were completed, the model with the lower validation loss was reloaded to produce the images in Figure 5.

4.3. Metrics

To evaluate the network performance, we use two metrics: the first is the so-called F1-Score (or DiceMulti metric), an extension of the concept of Søresen-Dice coefficient [Sør48] from the simple binary classification to the multiclass problem, see [OB19] for more details. Such a metric is already available in popular deep learning libraries, such as *Fast.ai* [HG20] and scikit-learn. The second measure is called *Surface Pattern Metric* (or *SurfPatt metric* for short) and it is inspired by [BFC08, BSFC08]. In short, it checks how many pixels of a predicted segmentation class are correctly labeled with respect to the ground-truth, without considering the pixels marked as background. In our problem, we consider as background pixels both the white pixels that does not belong to the patch (the actual background) and the areas of the models without any meaningful pattern (class label: *NoPatt*). Thanks to this metric, since the background pixels are dominant in this dataset, we can evaluate the actual improvement of the training in segmenting only the pixels belonging to the "real" classes of the problem, ignoring the performance in segmenting the background and no pattern classes. We also consider the standard Precision and Recall scores used for CNN classification. Briefly, the Precision score is the number of true positive (pixels) over the number of positive. It evaluates the performance based on how many false positives are produced. The Recall score is the number of true positives over the number of true positives plus the number of false negatives. It evaluates the performances based on how much of the total elements are correctly classified. To calculate *F1-Score*, *Precision* and *Recall* as efficiently as possible, we modified the corresponding *scikit-learn* [PVG*11] function to make it possible to calculate the Multiclass Confusion Matrix (MCM) one batch at a time, without having to pass the entire *y_pred* and *y_true* vectors. This would have been extremely inefficient since, in the case of semantic segmentation, the size of these two vectors is equal to the double of the entire dataset in terms of number of pixels and therefore of bytes allocated in RAM (hundreds of gigabytes).

4.4. Results

Table 1 shows the results of the training sessions with the network fed with $R = 2$ patches in the three different data augmentation regimes. In this setup, the best training epoch occurred using only images generated with the Shape Index colorization and achieved 0.9108 of F1-Score and 0.9354 of Precision in the *basic-data-aug* data augmentation regime. The best score of *SurfPatt* metric, on the other hand, was 0.7634 and was obtained by training the network only with RGB images in the *strong-data-aug* data augmentation regime.

As easily predictable, training without data augmentation almost

RGB+SI $R = 1.25$	Tr. Loss	Val. Loss	*SurfPatt m.*	F1-Score	Precision	Recall	Epochs
no-data-aug	0.0594	0.1317	0.5992	0.8002	0.8098	0.7934	50+28
basic-data-aug	0.1325	0.1300	0.5930	0.7896	0.8319	0.7696	66+21
strong-data-aug	0.1547	0.1203	0.5930	0.7798	0.8643	0.7494	48+37
RGB+SI $R = 2$	Tr. Loss	Val. Loss	*SurfPatt m.*	F1-Score	Precision	Recall	Epochs
no-data-aug	0.0348	0.0853	0.6390	0.8569	0.8685	0.8473	50+26
basic-data-aug	0.0652	0.0586	0.6610	0.8997	0.9147	0.8861	100+12
strong-data-aug	0.0653	0.0739	0.6173	0.8667	0.8784	0.8603	65+21
RGB+SI $R = 3$	Tr. Loss	Val. Loss	*SurfPatt m.*	F1-Score	Precision	Recall	Epochs
no-data-aug	0.0789	0.0824	0.7224	0.8742	0.8917	0.8611	98+16
basic-data-aug	0.0448	0.0439	0.7909	**0.9388**	**0.9481**	**0.9308**	100+15
strong-data-aug	0.1986	0.0458	**0.7938**	0.9358	0.9430	0.9299	65+23
RGB $R = 2$	Tr. Loss	Val. Loss	*SurfPatt m.*	F1-Score	Precision	Recall	Epochs
no-data-aug	0.0252	0.0787	0.7388	0.8926	0.8985	0.8882	65+24
basic-data-aug	0.0695	0.0578	0.7497	0.9012	0.8975	0.9071	81+13
strong-data-aug	0.0598	0.0592	0.7634	0.9010	0.8965	0.9060	80+16
SI $R = 2$	Tr. Loss	Val. Loss	*SurfPatt m.*	F1-Score	Precision	Recall	Epochs
no-data-aug	0.0169	0.0808	0.7486	0.8767	0.8924	0.8639	55+39
basic-data-aug	0.0581	0.0514	0.7508	0.9108	0.9354	0.8893	100+14
strong-data-aug	0.0605	0.0579	0.7330	0.8966	0.9284	0.8732	58+25

Table 1: *The metrics of the DeepLabv3+ networks trained with different parameters and setups.*

always has lower performance than training with data augmentation. When the training is performed in the *basic-data-aug* regime, on the contrary, it often reaches a higher F1-Score than that obtained in the *strong-data-aug* regime, a sign that the extra transformations of the latter are not advantageous for training with this type of data.

As can be seen in Figure 5, the network classifies the type of pattern very well, almost always attributing the right color to the corresponding area. However, the segmentation is not pixel perfect, sometimes the network does not perfectly follow the contours of the most complex patterns. This highlight that the network needs further training (therefore with more images, a greater data augmentation or both). However, the results are very promising, suggesting that this approach is on the right path and the task is manageable.

In Figure 5 it is also possible to see an interesting detail that showcases "qualitatively" the idea of the network's ability to generalize on new data. No dataset has a pixel-perfect ground-truth and not even ours can escape this "law": in the fourth *GT/Prediction* pair of images of the first column, it can be seen how the ground-truth does not give the correct label to one of the circlets in the lower part of the image. Nevertheless, the network predicts sufficiently well which class that pattern belongs to, an evidence that the network is capable of generalizing to previously unseen data. It is also possible to observe how the network performance increases by providing more context with patches whose radius is greater ($R = 3$) and decreases by reducing the context ($R = 1.25$). In particular, with patch radius $R = 3$ and the *basic-data-aug* regime, the network reaches an F-Score of 0.9388 and a Precision of 0.9481, while the network trained with a patch radius of $R = 1.25$ has the worst performance of all training runs, including networks trained without data augmentation. This is not surprising as smaller patches lead to smaller images, with less data from which the net can learn from.

All the code and data used to run these experiments (except for the code of the *reverse mesh labeling* tool, still under development) is available in the following GitHub repository: https://github.com/surface-pattern-recognition/surface-pattern-recognition

Below, we briefly discuss how to bring image segments back onto a fragment surface. First, for each sampled disk, we consider the mask derived from the segmentation of the RGB image and we use its colors to classify the vertices of the patch. More precisely, we first project all the vertices of a patch on the $xy-$plane (let us call them v'_i). Note that the disk was already previously aligned with respect to the same plane. Then we convert the image into a set of 2D points by creating a square grid of 2D points based on the image size (let us call them p'_j), so that the pixel labeling induces a labeling onto the point p'_j. Both point sets are then aligned so that the bounding boxes of points p'_j coincides with the bounding boxes of the points v'_i. Finally, each v'_i takes the label of the closest p'_j. Figure 6 shows an overview of this process.

A sample of the results we obtain on some of the models (at least one for each relief class) are shown in Figure 7. Compared to the previous color scheme, we add cyan to represent the areas of each model that are not covered (i.e. have not been sampled) by any patch. Note that most of both the ground-truth and predictions share the same set of labels, thus indicating good accuracy in semantic segmentation. Even labels with a very limited number of samples, such as the eyebrow (pink), are segmented well in the latest model of Figure 7. We can also see that some reliefs are better predicted than others. For example, the long incisions (black) line

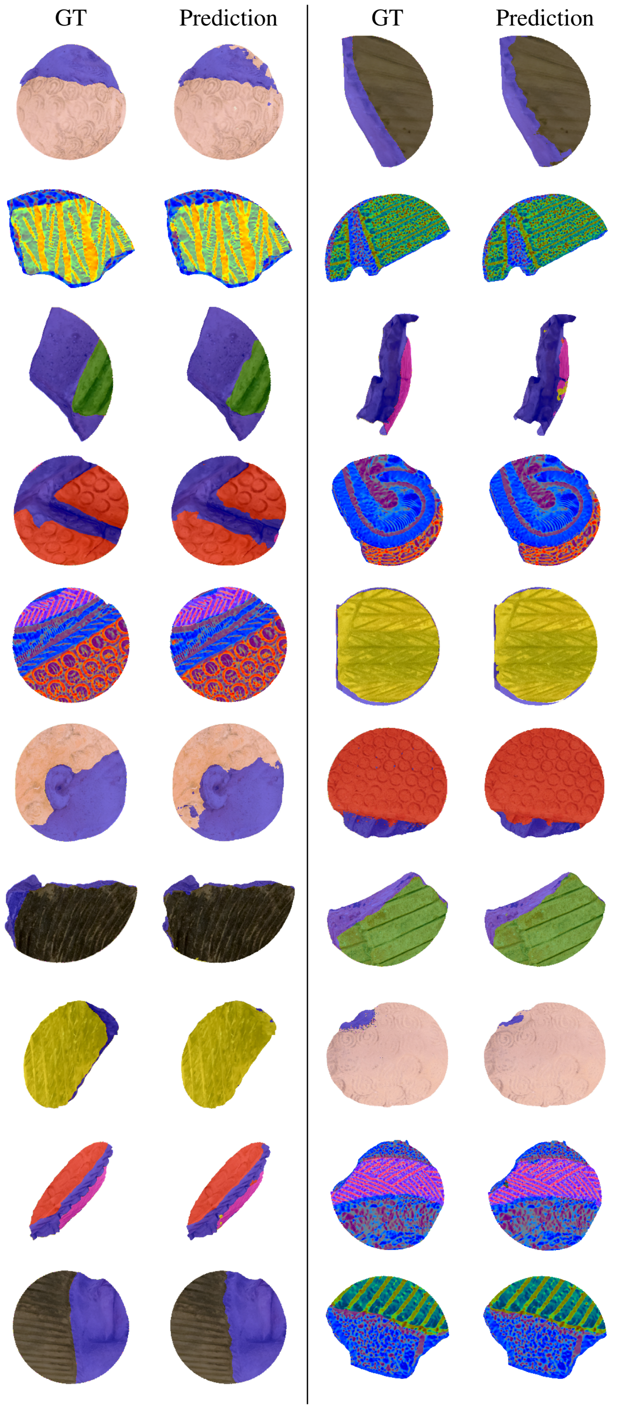

Figure 5: *Semantic segmentation of the ground-truth (first and third columns, labeled with GT) and our predictions (second and fourth columns, respectively). Different colors are used to label different patterns (or lack of). Again, yellow is* oblique fringe, *red is* circlets, *magenta is* eyebrows, *black is* long incisions, *green is* fringe *and pink is* spirals. *Blue is used to cover those part of the patches that are not patterns (*NoPatt, *see Section 3.3).*

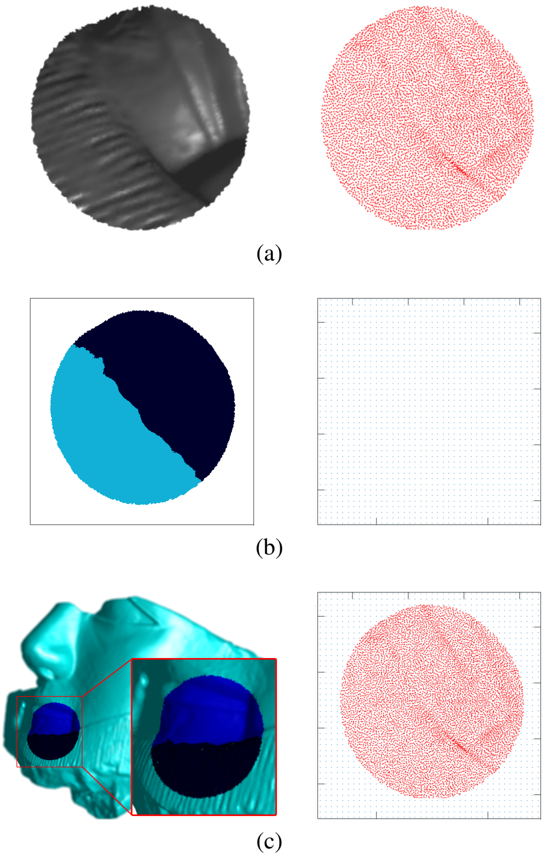

Figure 6: *Overview of how is it possible to derive a segmentation from the classified and segmented images. (a): one of the patches extracted from a model (sx) and its conversion in a 2D set of points (dx) (v_i' in the text), simplified for clarity. (b): the predicted mask obtained from that patch (sx) and its conversion in a 2D grid of points the same size as that of the picture (dx). (c): the final result on the model (sx) and the overlap of the v_i' points on the grid op points p_j'.*

up perfectly with the expected boundaries. The image segmentation back projection algorithm is still under development and needs some improvements such as better managing the regions where the disks overlap and how to treat areas of the model not covered by any patch. In fact, there are areas without labels that show that a few more samples would have been useful to completely cover the surface.

Limitations In the following, we list the main limitations of our method.

- *Patch sampling:* as shown in some examples in Figure 7, the disk-like sampling strategy is computationally very efficient but might leave some regions uncovered. To overcome this limitation we plan to use a sampling thickening strategy based on a uniform geodesic sampling. Moreover, we implicitly assume that a surface can be projected onto a regression plane without a large

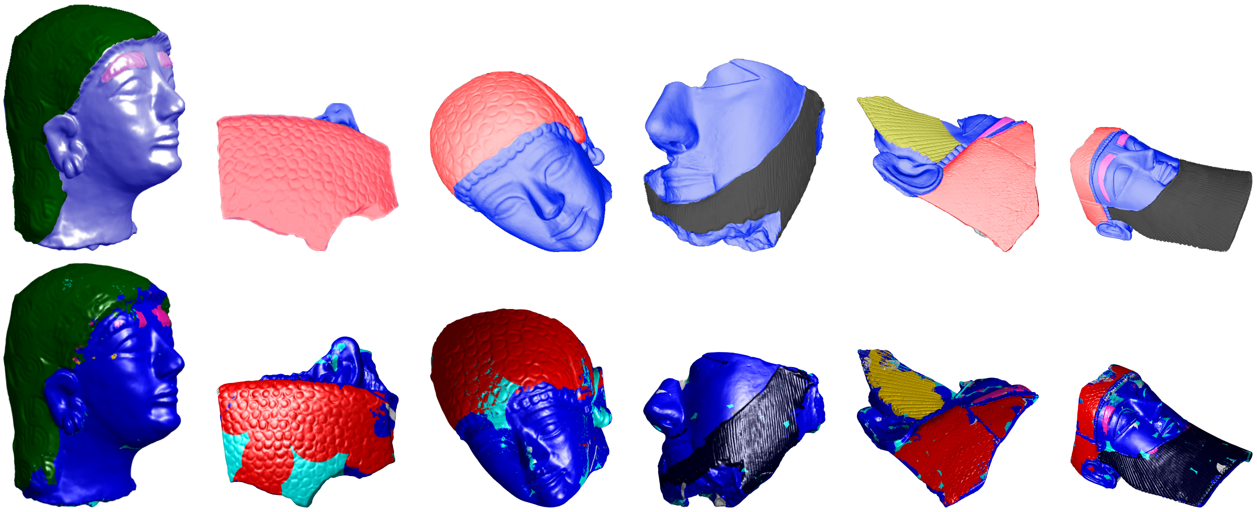

Figure 7: *Top: the ground-truth of a sub-sample of models, showing examples of all the reliefs in the SHREC'18 dataset. Bottom: The prediction of our method with the model* ***RGB+SI*** *R=2 basic-data-aug. Vertices in cyan represent unlabeled areas. The presence of these areas are caused to the randomness of the sampling, not a failure of the classification/segmentation process. See Section 4.4 for more details.*

distortion: this is generally true in the case of artifacts with artistic reliefs. However, in the case of highly curved surfaces, a possible solution is to modify the disk-like projection using curved templates, such as quadric surfaces.

- *NN training:* the major limitation of the current method concerns the high resolution of the images used for training the network (and the consequent choice of the depth of the backbone) required to achieve these high levels of segmentation accuracy. The next iteration of the method will involve a multi-radius random sampling for the patches and this should confer further robustness to the trained models, which can then hopefully be downscaled to lower resolution images and thus shallower backbones which are also faster to train.

5. Concluding remarks

Although the advent of high-resolution 3D scanners have made 3D models of archaeological finds available at previously unthinkable resolutions, such models are still a fairly scarce resource. Furthermore, precise ground-truth labeling for semantic segmentation is still a labor-intensive task that often requires massive interaction with domain experts and end users [PCDS19, PGDF*20]. Dealing with non-trivial datasets like the one shown in this paper, with a relatively small number of 3D models and therefore a limited amount of resulting images and "total amount of information", makes the task of semantic segmentation via CNNs, definitely challenging.

In this work, we have proposed a strategy to automatically perform a segmentation as compliant as possible with the characteristics and the reliefs on the surface of an archaeological fragment. To achieve this, we have investigated and experimented how to reduce the 3D surface pattern recognition problem into an image segmentation problem and for this we have fed a CNN which played the role of data segmenter and classifier. Going back from the images to the 3D surface is relatively simple as our images were obtained by the local projection of the points of a patch onto the regression plane and we have retained the relative height. Moreover, the problem of the non-uniqueness of the inverse of the projection is mitigated by the fact that for each sample we have considered only the connected component that contains it and therefore, also in the reverse step, we only consider points adjacent to it.

Overall, we observe that using a CNN-based approach is a challenging but promising solution for the relief-driven segmentation, even in the Cultural Heritage domain, where the use of learning-based methods is often prevented by the low amount of examples or the uniqueness of the models. Indeed, we have seen that the use of a set of images per model, even generated with a simple sampling strategy, is able to convey enough information to feed a neural network and obtain satisfactory results. Although most of the time the network correctly identifies the label of the relief pattern, the segmentation boundary is sometimes noisier than that of the ground-truth. Aggressive data augmentation policies certainly help postpone network overfit and help it generalize better, but what can really increase model accuracy is more data and more diversity in that data. A further possibility to make the most of this dataset is to feed the network with multiple images by sampling the same models at different scales. In this way, in addition to enriching the training set, the network would have the opportunity to see decorations and reliefs at different levels of detail. On a slightly different path, the recent unsupervised contrastive learning techniques, which are gaining traction in the literature [CKNH20, HFW*20, MM20, CMM*20], promise to make it possible to exploit the enormous amount of images of cultural heritage artifacts already available for free on the internet, thus avoiding the manual acquisition and labeling of hundreds of 3D models. Part of our efforts is already spent in this direction: indeed, unsupervised pre-training and subsequent fine-tuning for semantic segmentation are a topic that are still very new in the current literature.

Acknowledgements We thank Michela Spagnuolo for bringing interest and creating engagement on this topic. We would also like to thank Francesco Lescai, Luca Pavarino, Gabriele Merli and all the team managing the UniPV EOS cluster. Without their work, this paper would not have been possible.

This work has been partially developed within the CNR activity DIT.AD004.100 “Modellazione, analisi e confronto di forme 3D”.

References

[ADD*21] Anichini F., Dershowitz N., Dubbini N., Gattiglia G., Itkin B., Wolf L.: The automatic recognition of ceramics from only one photo: The archaide app. *Journal of Archaeological Science: Reports 36* (2021), 102788. doi:https://doi.org/10.1016/j.jasrep.2020.102788. 2

[BFC08] Brostow G. J., Fauqueur J., Cipolla R.: Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters xx*, x (2008), xx–xx. 5

[BMS19] Biasotti S., Moscoso Thompson E., Spagnuolo M.: Context-adaptive navigation of 3d model collections. *Computers & Graphics 79* (2019), 1–13. doi:https://doi.org/10.1016/j.cag.2018.12.004. 3

[BMTA*17] Biasotti S., Moscoso Thompson E., Aono M., Hamza B., Bustos B., Dong S., Du B., Fehri A., Li H., Limberger F., Masoumi M., Rezaei M., Sipiran I., Sun L., Tatsuma A., Velasco Forero S., Wilson R., Wu Y., Zhang J., Zhao T., Fornasa F., Giachetti A.: SHREC’17: Retrieval of Surfaces with Similar Relief Patterns. In *Eurographics Workshop on 3D Object Retrieval* (2017), Pratikakis I., Dupont F., Ovsjanikov M., (Eds.), The Eurographics Association. 1

[BMTB*18] Biasotti S., Moscoso Thompson E., Barthe L., Berretti S., Giachetti A., Lejemble T., Mellado N., Moustakas K., Manolas I., Dimou D., Tortorici C., Velasco-Forero S., Werghi N., Polig M., Sorrentino G., Hermon S.: Recognition of Geometric Patterns Over 3D Models. In *Eurographics Workshop on 3D Object Retrieval* (2018), Telea A., Theoharis T., Veltkamp R., (Eds.), The Eurographics Association, pp. 71 – 77. 1, 2, 3, 4

[BSFC08] Brostow G. J., Shotton J., Fauqueur J., Cipolla R.: Segmentation and recognition using structure from motion point clouds. In *ECCV (1)* (2008), pp. 44–57. 5

[CCS12] Corsini M., Cignoni P., Scopigno R.: Efficient and flexible sampling with blue noise properties of triangular meshes. *IEEE Transactions on Visualization and Computer Graphics 18*, 6 (2012), 914–924. 2

[CKNH20] Chen T., Kornblith S., Norouzi M., Hinton G.: A simple framework for contrastive learning of visual representations. In *International conference on machine learning* (2020), PMLR, pp. 1597–1607. 8

[CMK*14] Cimpoi M., Maji S., Kokkinos I., Mohamed S., Vedaldi A.: Describing textures in the wild. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition* (Washington, DC, USA, 2014), CVPR ’14, IEEE Computer Society, pp. 3606–3613. 2

[CMKV16] Cimpoi M., Maji S., Kokkinos I., Vedaldi A.: Deep filter banks for texture recognition, description, and segmentation. *International Journal of Computer Vision 118*, 1 (May 2016), 65–94. 2

[CMM*20] Caron M., Misra I., Mairal J., Goyal P., Bojanowski P., Joulin A.: Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882* (2020). 8

[CZP*18] Chen L.-C., Zhu Y., Papandreou G., Schroff F., Adam H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)* (2018), pp. 801–818. 1, 2, 3

[DCV14] Digne J., Chaine R., Valette S.: Self-similarity for accurate compression of point sampled surfaces. *Computer Graphics Forum 33*, 2 (2014), 155–164. doi:https://doi.org/10.1111/cgf.12305. 1

[DTC*17] Debroutelle T., Treuillet S., Chetouani A., Exbrayat M., Martin L., Jesset S.: Automatic classification of ceramic sherds with relief motifs. *J. of Electronic Imaging 26*, 2 (2017), 1–14. 1, 2

[FBF77] Friedman J. H., Bentley J. L., Finkel R. A.: An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw. 3*, 3 (Sept. 1977), 209–226. 2

[Gia18] Giachetti A.: Effective characterization of relief patterns. *Computer Graphics Forum 37*, 5 (2018), 83–92. 1, 2

[GK15] Golla T., Klein R.: Real-time point cloud compression. *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2015), 5087–5092. 1

[HFW*20] He K., Fan H., Wu Y., Xie S., Girshick R.: Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020), pp. 9729–9738. 8

[HG20] Howard J., Gugger S.: Fastai: A layered API for deep learning. *Information 11*, 2 (Feb 2020), 108. doi:10.3390/info11020108. 5

[HSL*17] Hackel T., Savinov N., Ladicky L., Wegner J. D., Schindler K., Pollefeys M.: Semantic3D.net: A new large-scale point cloud classification benchmark. *CoRR abs/1704.03847* (2017). arXiv:1704.03847. 1

[HZRS15] He K., Zhang X., Ren S., Sun J.: Deep residual learning for image recognition. *CoRR abs/1512.03385* (2015). arXiv:1512.03385. 1

[Kar93] Karageorghis V.: The Cypro-Archaic period, large and medium size sculpture. *The chloroplast art of ancient Cyprus III* (1993), 59. 3

[KSH12] Krizhevsky A., Sutskever I., Hinton G. E.: Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems 25* (2012), 1097–1105. 2

[Kv92] Koenderink J. J., van Doorn A. J.: Surface shape and curvature scales. *Image and Vision Computing 10*, 8 (1992), 557–564. 1, 3

[MB18] Moscoso Thompson E., Biasotti S.: Description and retrieval of geometric patterns on surface meshes using an edge-based LBP approach. *CoRR abs/1805.00719* (2018). arXiv:1805.00719. 1

[MBG*20] Moscoso Thompson E., Biasotti S., Giachetti A., Tortorici C., Werghi N., Obeid A. S., Berretti S., Nguyen-Dinh H.-P., Le M.-Q., Nguyen H.-D., Tran M.-T., Gigli L., Velasco-Forero S., Marcotegui B., Sipiran I., Bustos B., Romanelis I., Fotis V., Arvanitis G., Moustakas K., Otu E., Zwiggelaar R., Hunter D., Liu Y., Arteaga Y., Luxman R.: Shrec 2020: Retrieval of digital surfaces with similar geometric reliefs. *Computers & Graphics 91* (2020), 199–218. doi:https://doi.org/10.1016/j.cag.2020.07.011. 1

[MM20] Misra I., Maaten L. v. d.: Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020), pp. 6707–6717. 8

[MTB19] Moscoso Thompson E., Biasotti S.: A Preliminary Analysis of Methods for Curvature Estimation on Surfaces With Local Reliefs. In *Eurographics 2019 - Short Papers* (2019), Cignoni P., Miguel E., (Eds.), The Eurographics Association. doi:10.2312/egs.20191006. 3

[MTBS*18] Moscoso Thompson E., Biasotti S., Sorrentino G., Polig M., Hermon S.: Towards an Automatic 3D Patterns Classification: the GRAVITATE Use Case. In *Eurographics Workshop on Graphics and Cultural Heritage* (2018), Sablatnig R., Wimmer M., (Eds.), The Eurographics Association. doi:10.2312/gch.20181372. 1, 3

[MTW91] Munro J. A. R., Tubbs H. A., Wroth W.: Excavations in Cyprus, 1890. third season's work. Salamis. *Journal of Hellenic Studies 12* (1891), 59. 3

[OB19] Opitz J., Burst S.: Macro F1 and macro F1. *CoRR abs/1911.03347* (2019). arXiv:1911.03347. 5

[OVSP13] Othmani A., Voon L. F. L. Y., Stolz C., Piboule A.: Single tree species classification from terrestrial laser scanning data for forest inventory. *Pattern Recognition Letters 34*, 16 (2013), 2144–2150. 2

[PCDS19] Ponchio F., Callieri M., Dellepiane M., Scopigno R.: Effective annotations over 3d models. *Computer Graphics Forum* (2019). 8

[Pey] Peyre G.: Toolbox graph - A toolbox to process graph and triangulated meshes. http://www.ceremade.dauphine.fr/ peyre/ matlab/graph/content.html. 3, 4

[PGDF*20] Pavoni G., Giuliani F., De Falco A., Corsini M., Ponchio F., Callieri M., Cignoni P.: Another brick in the wall: Improving the assisted semantic segmentation of masonry walls. In *Eurographics Workshop on Graphics and Cultural Heritage* (2020), The Eurographics Association. 8

[PVG*11] 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.: Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12* (2011), 2825–2830. 5

[RBFB13] Ruhnke M., Bo L., Fox D., Burgard W.: Compact rgbd surface models based on sparse coding. In *AAAI* (2013). 1

[Sør48] Sørenson T.: *A Method of Establishing Groups of Equal Amplitude in Plant Sociology Based on Similarity of Species Content and Its Application to Analyses of the Vegetation on Danish Commons*. Biologiske skrifter. I kommission hos E. Munksgaard, 1948. 5

[TP97] Tittle J. S., Perotti V. J.: The perception of shape and curvedness from binocular stereopsis and structure from motion. *Perception & psychophysics 59*, 8 (1997), 1167–1179. 3

[UUTCIC*] (UK) I. I. C., (UK) B. M., The Cyprus Institute (Cyprus) C. N. d. R. I. o. A. M., (Italy) I. T., of Amsterdam (Netherland) U., of Technology (Israel) T. I. I., of Haifa (Istrael) U.: GRAVITATE: Discovering relationships between artefacts using 3D and semantic data. EU H2020 REFLECTIVE project. 3

[WBB15] Werghi N., Berretti S., Bimbo A. D.: The mesh-LBP: A framework for extracting local binary patterns from discrete manifolds. *IEEE Trans. Image Processing 24*, 1 (2015), 220–235. 1

[WJHM15] Weinmann M., Jutzi B., Hinz S., Mallet C.: Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers. *ISPRS Journal of Photogrammetry and Remote Sensing 105* (2015), 286–304. doi:https://doi.org/10.1016/j.isprsjprs.2015.01.016. 1

[WLKT09] Wei L.-Y., Lefebvre S., Kwatra V., Turk G.: State of the art in example-based texture synthesis. In *Eurographics 2009, State of the Art Report, EG-STAR* (2009), Eurographics Association. 1

[WTBdB15] Werghi N., Tortorici C., Berretti S., del Bimbo A.: Local binary patterns on triangular meshes: Concept and applications. *Computer Vision and Image Understanding 139* (2015), 161 – 177. 1

[ZPS*16] Zeppelzauer M., Poier G., Seidl M., Reinbacher C., Schulter S., Breiteneder C., Bischof H.: Interactive 3D segmentation of rock-art by enhanced depth maps and gradient preserving regularization. *J. Comput. Cult. Herit. 9*, 4 (Sept. 2016), 19:1–19:30. 1, 2