Amphora Detection Based on a Gradient Weighted Error in a Convolution Neuronal Network

Jérôme Pasquet, Stella Demesticha, Dimitrios Skarlatos, Djamal Merad, Pierre Drap

IMEKO International Conference on
Metrology for Archaeology and Cultural Heritage
Lecce, Italy, October 23-25, 2017

Amphora Detection Based on a Gradient Weighted Error in a Convolution Neuronal Network

Jérôme Pasquet[1], Stella Demesticha[2], Dimitrios Skarlatos[3], Djamal Merad[1], Pierre Drap[1]

[1] *Aix-Marseille Université, CNRS, ENSAM, Université De Toulon, LSIS UMR 7296, Domaine Universitaire de Saint-Jérôme, Polytech, Av. Escadrille Normandie-Niemen, Marseille, France*
[2] *University of Cyprus, Cyprus*
[3] *Cyprus University of Technology, Civil Engineering and Geomatics Dept.*

***Abstract* – In this paper, we propose a method based on pixel prediction to detect objects into a large image. We propose to integrate the Weighted Error Layer (WEL) in a Convolution Neuronal Network (CNN) architecture in order to weight the error during the back-propagation and to reduce the impact of the borders. We estimate the orientation of the objects when the detection step is achieved. Our proposed layer is evaluated on real data in order to detect amphorae on the Mazatos underwater archaeological site.**

I. INTRODUCTION

Mazotos shipwreck lies at a depth of -44 m, some 14 nautical miles south-west of Larnaca, 1.5 nm from the shore. The main visible feature of the site is a virtually undisturbed concentration of amphorae on a sandy, almost flat seabed. Its maximum dimensions are 17.5 x 8 m. The oblong concentration, almost in the form of a ship, has a north-south orientation and consists of 500-800 Chian amphorae partly or totally visible, dating to the middle of fourth century BC [3]. In 2007, the initial photographic survey was aiming in the creation of a photomosaic and sketch of the site, to be used for further planning, rather than documentation. These photos were captured in order to provide complete monoscopic coverage, rather than a multi-view survey. In 2012, after two excavation periods, with the wreck site being disturbed, the original photo data set from 2007 was re processed [4] with a free network adjustment, to establish both the control point network as well the 3D documentation of the site as discovered. Because of the initial scope of the 2007 photography, and the strong relief of the object, there are 3D gaps on the 3D model.

The set of photographs made in 2007 were then used in this paper in order to automatically extract known amphorae present on the site.

Using both an orthophoto and a dense cloud of 3D points we propose a fully automated pipeline able to detect amphorae, the corresponding typology and estimate a pose for each artifact. The goal is to obtain a full 3D representation of the cargo using partial observations of each amphorae and the corresponding theoretical model.

The rest of the paper is organized as follow: first, in section ii. we present the data used. Then, in section iii. we introduce the CNN and our proposed layer. Further, section v. resumes the operation of the orientation estimation. Finally, we conclude and give some results in the last section.

II. MAZOTOS DATA

In our context, we detect amphorae in a 3D scene. Amphora detection is a challenging problem, as amphorae vary in orientation and appearance. Indeed, they can be broken or hidden under the sand or by another amphora. Moreover the second difficulty has to do with the data; as a matter of fact, the amount of examples of amphora is very low which is an issue to train a machine learning model.

III. CONVOLUTIONAL NEURAL NETWORK

To deal with the high number of 3D points we detect the amphorae directly on the orthophoto obtained by the 2D projection. The size of this orthophoto is around 38,000×15,000 pixels.

To avoid the lack of data, we use a pixel segmentation method based on a CNN [11, 12]. However the CNN can not process the entire image at once. In such a case, it is thus appropriate to process the image piece by piece. So we propose to use a sliding window to get patches of size 400×400 as input of the CNN.

The CNN is composed of 7 convolution layers and 3 pooling layers to reduce the dimension of the input image to a set of feature maps of size 25×25. The CNN is then composed by 3 unpooling layers and 3 convolution layers to obtain the final probability map. We perform a batch normalization [8] and then we apply the rectified linear unit (ReLU) function after each convolution.

The Xlendi database [5] is used to pre-train the CNN using an auto-encoder process [7]. Then, we do finetuning on the CNN and we consider around 25% of the Mazotos

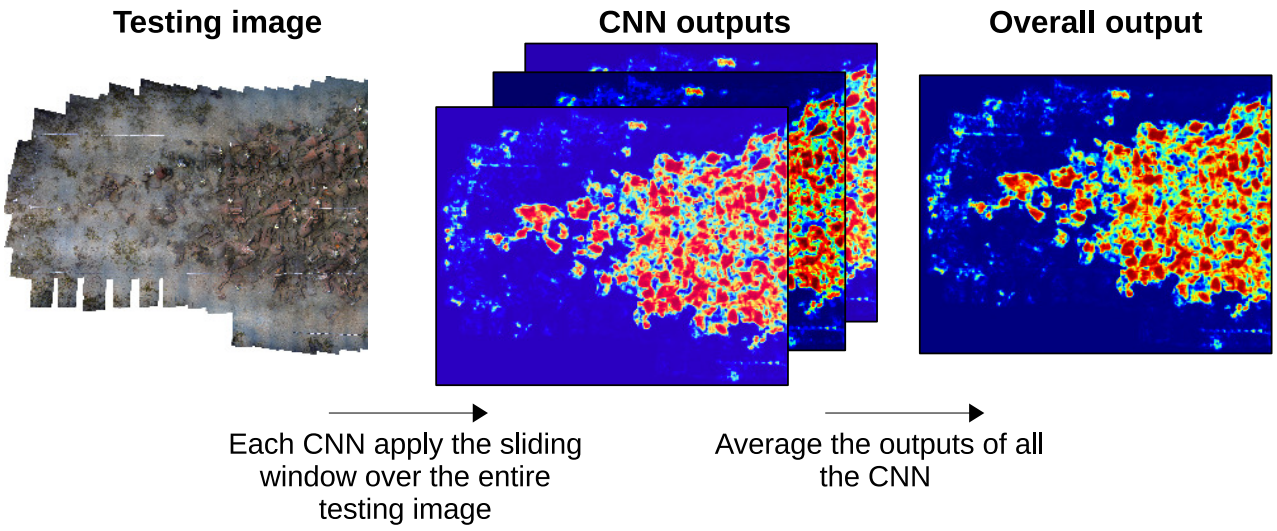

Fig. 1. The testing images is processed by all the models in the CNN ensemble. The final result is the average of all the different outputs.

image as the ground truth. During the learning step, random patches of the ground truth of size 400×400 feed the CNN. We define 3 classes in the database : the head which is the rim, the neck and the handles of the amphora; the body of the amphora; and all others objects such as rock, sand, piece of amphora...

During the prediction step it is impossible to predict if a pixel on the border of the patch, noted p, belongs to an amphora class or to a broken piece of amphora. To solve this problem the sliding window has a stride of 1 pixel and so the pixel p has different predictions corresponding to all positions in the sliding window. We can use the average of all the predictions as a final prediction score.

However during the training step this problem is more complex. During the back-propagation, the CNN is activated to give a prediction for each pixel in the patch. This prediction map is compared to the ground truth to obtain an error on each pixel. This error is then propagated in the network to update the weights. As shown in Figure 2, the classes of pixels close to the borders can not be predicted using only the content of the sliding window. The error introduced by the predictions of these pixels are back-propagated in the network and distorts the updating of weights.

To solve this problem, we propose to add a specific layer on the last convolution layer. This layer, named Weighted Error Layer (WEL), is equivalent to the identity function during the forward pass and so does not change the features created by the last layer. WEL is used during the backward process.

The update rule for a weight, noted w_{ij}, between the

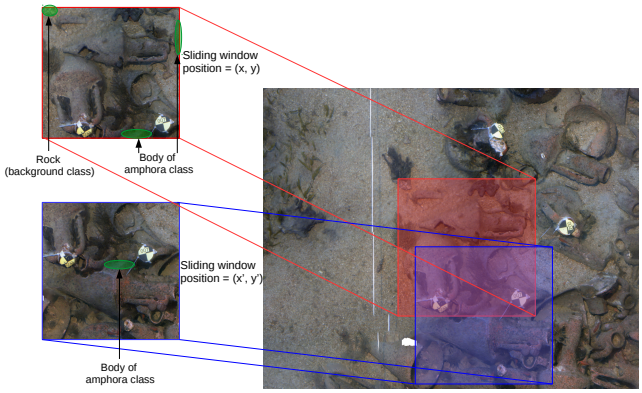

Fig. 2. Representation of the content of the sliding window at two different positions. The red sliding window shows two pieces of amphora on the border and one piece of rock which can not be identified by human. The blue sliding window represents the same amphora as the red one but it can be easily identified using the entire content.

neuron i and j is :

$$\Delta w_{ij} = \lambda \Delta w_{ij} - \epsilon \langle \frac{\delta E}{\delta w_{ij}} \rangle_D \qquad (1)$$
$$w_{ij} = w_{ij} + \Delta w_{ij}$$

where λ is the momentum, ϵ is the learning rate and E is the average error over the batch D. The error can be defined as :

$$\frac{\delta E}{\delta w_{ij}} = o_i . \delta_j \qquad (2)$$

where o_i is the output of the neuron i and δ_j is the error product by the neuron j. In our case, the CNN produces one probability map for each different labels (neck, body, other). The index j, in the equation 2 can be expressed as a position (x, y) and a label l.

WEL is associated to a specific function in order to weight the error gradient back propagated. We propose to use a Gaussian function as :

$$\frac{\delta E}{\delta w_{ij}} = o_i.\delta_j.A.e^{-\frac{\sqrt{(x-x_0)^2+(y-y_0)^2}}{2.\sigma^2}} \quad (3)$$

where A is a normalization term and σ, x_0, y_0 are Gaussian parameters. We notice in the equation 3 that the index l is not used because WEL is processed to all probability maps in the same way.

In this paper, we use the following WEL parameters : $A = 3$, $\sigma = 4.1$, $x_0 = 200$ and $y_0 = 200$. So the error given by the central pixel is weighted by a 3 factor and a pixel located on the left border is weighted by a 7.8×10^{-3} factor.

Moreover during the testing step we use WEL to weight the prediction map. We note $p(x, y)$ the prediction of pixel p with the relative position (x, y) in the sliding window. The final prediction, noted p_f, given by the CNN is :

$$p_f = \frac{\sum_{x=0}^{400}\sum_{y=0}^{400} p(x,y).A.e^{-\frac{\sqrt{(x-x_0)^2+(y-y_0)^2}}{2.\sigma^2}}}{\sum_{x=0}^{400}\sum_{y=0}^{400} A.e^{-\frac{\sqrt{(x-x_0)^2+(y-y_0)^2}}{2.\sigma^2}}} \quad (4)$$

So the final score for each pixel is given by the sum of all the predictions for this pixel divided by the sum of the WEL weights.

IV. SEGMENTATION RESULTS

To get better results we train an ensemble of four CNN with different initializations [1, 2]. The training step for each CNN takes around one day using the Caffe [9] framework on a GTX 1080 graphics card. The adaptive gradient [6] optimization method is used during the training step. We also use the following data augmentation [10] techniques to improve the training database information, which include horizontally and vertically flipping, random crops, rotation and random scaling.

On the Figure 1 we can see the segmentation map given by the CNN. To overcome the asymmetric data we reduce the background probability by 2 and then we build the quantify map using the maximum probabilities. The Figure 3 represents the efficiency of our approach based on different scenarios.

The objects in the Xlendi image doesn't have the same topologies than the objects in Mazatos image. However, the figure 3 shows that the CNN using a pre-training on Xlendi gives better performances. Indeed, for a recall of

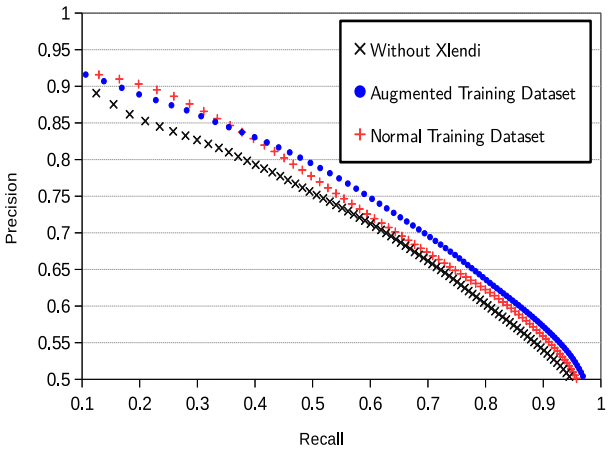

Fig. 3. The ROC curves represent the performance of three CNN using different databases. The CNN A uses only the data from Mazatos and not the Xlendi data. The CNN B uses the data from Xlendi as pre-train and Mazatos data for the training. The CNN C is trained using 10% more amphorae from Mazatos.

80.0%, the precision is 60.0% for the CNN without a pre-train on Xlendi compare to 61.4% for the CNN including a pre-train step on Xlendi data. Moreover the size of the training database impacts the performance. We can assume that using more images during the training step and the pre training increases the overall performance. In our case, we fix the threshold to get a hight recall. Indeed the following orientation estimation step will remove the false positive detections and the duplicate detections.

V. ORIENTATION ESTIMATION

After the prediction step, we obtain the probability map of each pixel. We quantify the map using a threshold in order to get all the positions of pixels which belong to the head or body classes. We estimate the position of all the amphorae using a difference of Gaussian approach (DoG). The standard deviation for Gaussian Kernel is in the range of 20 to 90. The smallest value corresponds to the size of the amphora's rim, and the largest value to the size of the entire amphora. After the detection, in order to check all the blob detections, we reject each blob whose the average probability of all its pixels values is below 0.85.

For each blob detected, we extract from the 3D model a 3D patch of size 300×300×300. In this 3D patch we remove all points corresponding to pixels whose the probability to be an amphora are low. We check that all extracted patches contain enough information to process an orientation estimation. The Figure 4 represents the pipeline from the tested image to the 3D extracted patch.

We assume that amphora is randomly oriented and its

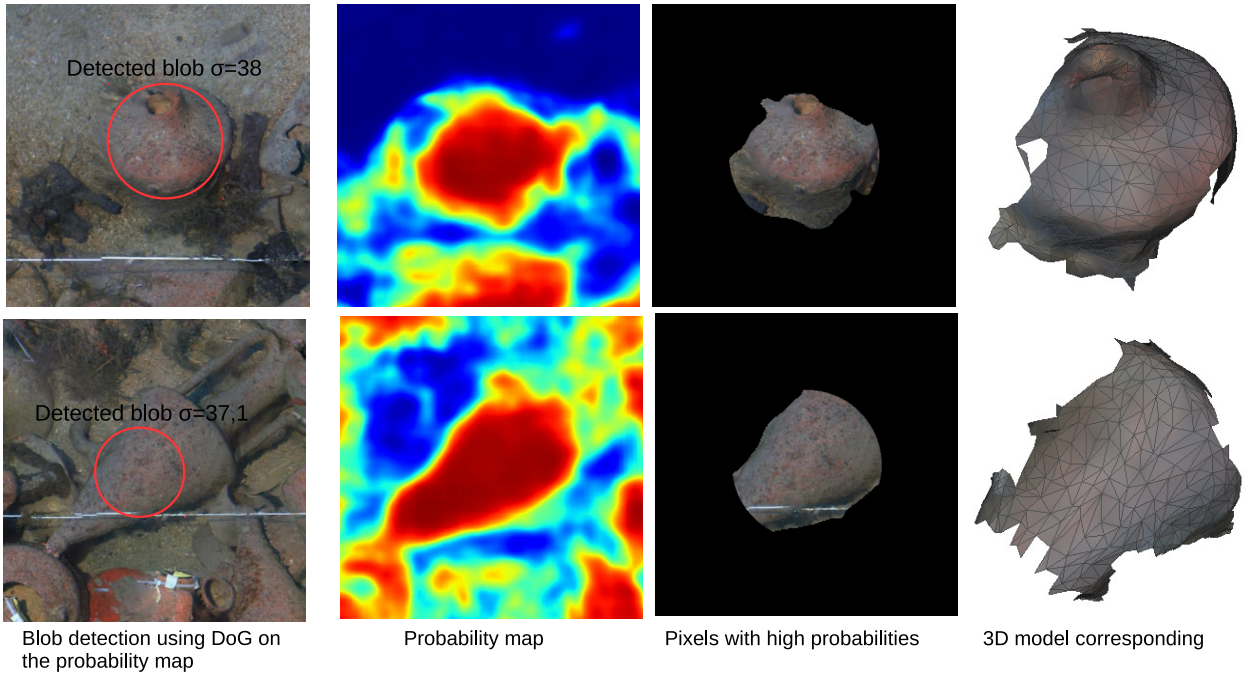

Fig. 4. Representation of the proposed framework using two images as examples. The first step uses the ensemble of CNN to create the probability map. In the second step, all the blobs are detected in the probability map. In the third step, pixels whose the probability to be an amphora is low are masked for each blob. Finally, We apply the mask on the 3D model in order to fit the theoretical 3D model using the Go-ICP algorithm.

representation is only a piece of the theoretical model. To deal with this problem we propose to use the Go-ICP [13] witch is an algorithm of iterative closest point based on the branch-and-bound schema. This algorithm is efficiency because the global solution is found regardless of the initialization and the number of points needed to fit the data with the model can be low.

VI. CONCLUSION

In this paper we introduce the Weighted Error Layer to improve the pixel segmentation. First experiment on the Mazatos database shows good performance. Indeed we detect around 90.3 percent of amphorae. The main issue in the data is that amphorae are too close to each other and so some amphorae are just ignored during the blob detection. A more costly alternative would be to repeat all the steps and to remove true amphorae detected and fitted. However it is not feasible because the estimation of the orientation process needs huge computational cost, around 8 hours/amphora. Therefore, it will be interesting to optimize this fitting step for future work. Moreover as this work is done using only the orthophoto, the adaptation of the network to take into account the 3D information could improve the detection step.

VII. ACKNOWLEDGEMENT

This work is partially supported by European Union Horizon 2020 research and innovation programme in the framework of the iMareCulture project (http://www.imareculture.eu) and by the French Armaments Procurement Agency (DGA) within the LORI project.

Special thanks to Dr. Bruce Hartzler, for the initial photo documentation of the wreck in 2007.

VIII. REFERENCES

[1] Forest Agostinelli, Michael R Anderson, and Honglak Lee, *Adaptive multi-column deep neural networks with application to robust image denoising*, Advances in Neural Information Processing Systems 26 (C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, eds.), Curran Associates, Inc., 2013, pp. 1493–1501.

[2] D. Ciregan, U. Meier, and J. Schmidhuber, *Multi-column deep neural networks for image classification*, 2012 IEEE Conference on Computer Vision and Pattern Recognition, June 2012, pp. 3642–3649.

[3] Stella Demesticha, *The 4th century bc mazotos shipwreck, cyprus: a preliminary report*, International Journal of Nautical Archaeology **40** (2011), no. 1, 39–59.

[4] Stella Demesticha, Dimitrios Skarlatos, and Andonis Neophytou, *The 4th-century b.c. shipwreck at mazotos, cyprus: New techniques and methodologies in the 3d mapping of shipwreck excavations*, Journal of Field Archaeology **39** (2014), no. 2, 134–150.

[5] Pierre Drap, Djamal Merad, Bilal Hijazi, Lamia Gaoua, Mohamad Motasem Nawaf, Mauro Saccone, Bertrand Chemisky, Julien Seinturier, Jean-Christophe Sourisseau, Timmy Gambin, and Filipe Castro, *Underwater photogrammetry and object modeling: A case study of xlendi wreck in malta*, Sensors **15** (2015), no. 12, 30351–30384.

[6] John Duchi, Elad Hazan, and Yoram Singer, *Adaptive subgradient methods for online learning and stochastic optimization*, J. Mach. Learn. Res. **12** (2011), 2121–2159.

[7] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio, *Why does unsupervised pre-training help deep learning?*, J. Mach. Learn. Res. **11** (2010), 625–660.

[8] Sergey Ioffe and Christian Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift.*, ICML (Francis R. Bach and David M. Blei, eds.), JMLR Workshop and Conference Proceedings, vol. 37, JMLR.org, 2015, pp. 448–456.

[9] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell, *Caffe: Convolutional architecture for fast feature embedding*, arXiv preprint arXiv:1408.5093 (2014).

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, *Imagenet classification with deep convolutional neural networks*, Advances in Neural Information Processing Systems 25 (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), Curran Associates, Inc., 2012, pp. 1097–1105.

[11] Pedro O. Pinheiro and Ronan Collobert, *From image-level to pixel-level labeling with convolutional networks*, The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2015.

[12] Evan Shelhamer, Jonathan Long, and Trevor Darrell, *Fully convolutional networks for semantic segmentation*, IEEE Trans. Pattern Anal. Mach. Intell. **39** (2017), no. 4, 640–651.

[13] Jiaolong Yang, Hongdong Li, Dylan Campbell, and Yunde Jia, *Go-icp: A globally optimal solution to 3d icp point-set registration*, IEEE Trans. Pattern Anal. Mach. Intell. **38** (2016), no. 11, 2241–2254.