Computational Visual Ceramicology: Matching Image Outlines to Catalog Sketches

Barak Itkin, Lior Wolf, Nachum Dershowitz
The School of Computer Science, Tel Aviv University
barak.itkin@gmail.com, wolf@tau.ac.il, nachum@tau.ac.il

Abstract

Field archeologists are called upon to identify potsherds, for which they rely on their professional experience and on reference works. We have developed a recognition method starting from images captured on site, which relies on the shape of the sherd's fracture outline. The method sets up a new target for deep-learning, integrating information from points along inner and outer surfaces to learn about shapes. Training the classifiers required tackling multiple challenges that arose on account of our working with real-world archeological data: paucity of labeled data; extreme imbalance between instances of different categories; and the need to avoid neglecting rare classes and to take note of minute distinguishing features of some classes. The scarcity of training data was overcome by using synthetically-produced virtual potsherds and by employing multiple data-augmentation techniques. A novel form of training loss allowed us to overcome classification problems caused by under-populated classes and inhomogeneous distribution of discriminative features.

Introduction

Pottery is the most common type of excavated artifact. Its identification permits the understanding of the chronology, function, and importance of an archeological site. Identification is based on the archeologist's domain knowledge and is usually achieved by matching unearthed potsherds to exemplars recorded in catalogs of semi-standardized archeological typologies. These catalogs typically contain—for each type—a standardized sketch of the complete vessel and occasionally a few photos of excavated instances.

In the most common case, the pottery is undecorated, either because it was manufactured that way or because any decorations have been lost to the ravages of time. We tackle the task of classifying the shape of a potsherd based on a single image of it, as depicted in Fig 1(a). Our prior attempts to apply standard image-based classification to undecorated pottery produced poor results, thus we turned to identification relying on the geometry of the sherd. Since our work is aimed toward aiding archaeologists in the field, we forwent complex methods of extracting 3D geometry—as these are impractical in field conditions, and relied instead on the 2D outline of the sherd's fracture surface as the sole source of shape information. After marking the outline in a semi-automatic way and determining the scale using a ruler (Fig 1(b)), our AI-powered mobile app supplies the identification in the form of a list of archaeological types, ranked by their computed relevance to the photographed potsherd.

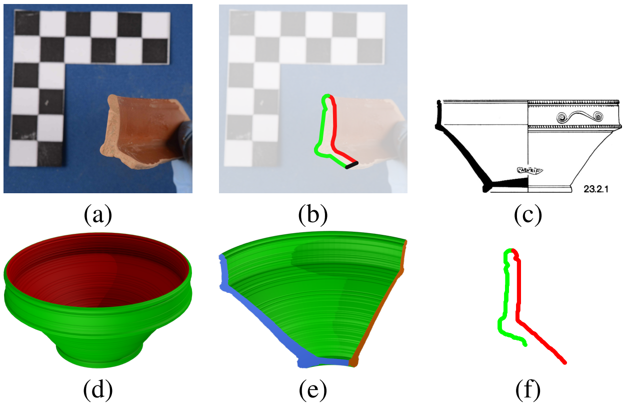
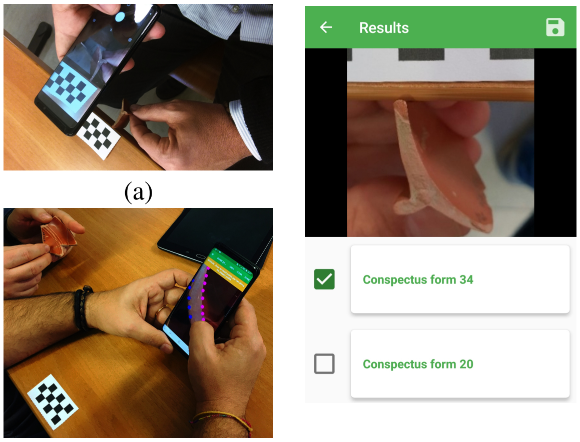

(a)
(b)
(c)
(d)
(e)
(f)

Figure 1: Illustration of the archeological data. (a) An image of a sherd, positioned to show the fracture surface, with a reference scale ruler in the background. (b) A traced fracture outline, overlaid on the source image: green for the outer profile; red for the inner profile; black is for break lines that are ignored by the algorithm. (c) An archeological sketch as it appears in a catalog. One or more sketches define a class of pottery. (d) A 3D computer graphics vessel obtained by rotating the catalog sketch. (e) A synthetic sherd obtained by breaking the 3D vessel. (f) A fracture outline obtained directly from the sketch, skipping the 3D reconstruction and shattering processes.

A major challenge in training the AI tool is that one cannot obtain sufficient training samples that are similar to those used to test or those evaluated in the field, as only a handful of sherds per class have ever been digitized. Furthermore, even if all extant sherds were to be digitized, the variability in the dataset would still cover only a small fraction of the space of possible sherds. Instead, we define each class by one or more 2D sketches of the profile of the complete vessel; see Fig 1(c). Note that the catalog sketch describes the geometry of the profile of the entire vessel, while the



(a)

(b)

(c)

Figure 2: Illustration of the mobile app workflow. (a) The user captures an image of the potsherd. The ruler is placed on a table and the sherd is held in a position such that the scales approximately match. (b) The user employs an intuitive interface to semi-automatically mark the outline of the potsherd. (c) The app applies our method to search for matching pottery from the catalog entries.

excavated sherd is a relatively small piece of the original, containing very limited information regarding the shape as a whole. This poses a serious challenge, as do the accidental shape features introduced when it shattered.

The outline of the fracture is a consequence of both the geometry of the original vessel and of the random breakage incident. On the dataset side, we could have reconstructed the 3D pottery by rotating the profile of the vessel (Fig 1(d)) and then "shattered" it in order to obtain synthetic sherds (Fig 1(e)). (Cf. (Banterle et al. 2017).) However, to avoid the associated computation overhead of 3D reconstruction, we propose a method to obtain the synthetic fracture surface (Fig 1(f)) directly from the 2D catalog sketch, and even on-the-fly during training. To identify outlines, we train a network that supports the unique characteristics of archeological outlines, including the need to distinguish between the inner and outer surfaces of the sherd, the significance of the order of points along the outline, the inherent noise in the tracing process, and the need to compensate for sub-optimal data acquisition processes.

As a real-world application, we were compelled to overcome a large set of compounding challenges. These include: (1) the lack of real-world data to train on; (2) a partial view of the object that is obtained by its random breakage, which presents large variability; (3) a large portion of the sherds, among both the synthetic training samples and the captured test samples, are almost completely non-informative; (4) very similar classes, making their distinction more challenging and also causing ambiguity in the ground-truth classification of the test data; and (5) a noisy acquisition process, prone to errors when extracting the outline and determining the scale from the real-world images.

In addition, to be useful for experts, there is an acute need to optimize to fit considerations beyond accuracy. For example, most neural network losses would be prone to sacrificing challenging classes so as to improve average accuracy across all classes. However, a reference tool such as ours delivers the most value when the identification is less obvious. To tackle the heterogeneous and unbalanced nature of the data, we train using a novel weighting technique that considers both the error of each ground truth class and false positives in each class.

Our results demonstrate a relatively high recognition rate in the face of these many challenges. To ensure the validity of our results, development was carried out in two phases. First, we developed the method on one dataset of potsherds of one specific family; in the second phase, evaluation was performed based on images of sherds captured with a dedicated mobile app. The app is already being used in the field (Anichini et al. 2020) and involves a simple capturing process, as depicted in Fig 2.

The evaluation of the second phase was done on the same ceramic family used for developing the method, as well as on three unseen families, using the same method, with exactly the same training pipeline and (hyper-) parameters. With the Phase I dataset, out of 65 different classes, we are able to identify almost 74% within the top-10 results. With three additional datasets received after the completion of the research phase, and without any tweaking of the pipeline, we reached 81%, 68%, and 60% top-10 accuracy for 65, 98, and 94 classes, respectively. Thus, our network can serve as the basis of a reliable reference tool for the use of archeologists in the field, one that significantly narrows down the list of relevant classes to be considered for each unearthed sherd.

Related Work

Automated pottery classification. In the absence of organic material to allow for carbon dating, pottery classification provides an indispensable tool for dating excavated objects. Much of the work on automated identification of sherds is based on 3D scanning or multi-view reconstruction technologies (Kampel and Sablatnig 2006; Karasik 2010; Calin et al. 2012; Barreau et al. 2014). However, the adoption of such methods is very limited due to the practical difficulties of 3D acquisition in the field. In addition, the challenges of analyzing 3D shapes have only been partially solved. The automatic analysis of profiles of potsherds has been studied using classical computer vision methods, such as the Hough transform (Durham, Lewis, and Shennan 1995), various morphological features (Karasik and Smilansky 2011; Lucena et al. 2014), and curvature descriptions (Gilboa et al. 2004). None of these is robust enough to be applied automatically on a varied set of excavated sherds, and much of it has only been applied to complete profiles. As mentioned, appearance-based methods (Makridis and Daras 2013; Poblome, Bes, and Piccoli 2013) are not relevant for the kind of sherds under consideration here.

Generating potsherds from profile drawings. Reconstruction from line drawing or sketches is a classical problem (e.g. Malik (1987); Tian, Masry, and Lipson (2009); Wang et al. (2009); Xu et al. (2014)). Architects, like archeologists, use semi-structured sketches, which can aid in reconstruction; see (Yin, Wonka, and Razdan 2009). A pipeline for extracting synthetic sherds based on catalog sketches was presented in (Banterle et al. 2017), where it was suggested to reconstruct the 3D model of a class by rotating the inner and outer profiles. To make it computationally feasible to generate a model from an outline with thousands of points, outline-simplification algorithms were used to narrow down the number of points. Such simplifications are detrimental, however, since some pottery present delicate details, and the discriminative parts are sometimes only 1–2 cm long. Since the size of the rotational model is quadratic in the number of profile points (typically several thousand), without simplification, the generation of millions of training samples is infeasible.

Sim2Real. We bridge the large semantic gap between sketches and excavated potsherds by training on synthetic data. Simulation to real world (sim2real) is used to overcome the sample complexity of reinforcement learning (RL) methods, e.g., (Andrychowicz et al. 2018; Tan et al. 2018; Peng et al. 2018). Computer graphics animations and images are used extensively to train and evaluate deep optical flow networks (Fischer et al. 2015), detect text (Gupta, Vedaldi, and Zisserman 2016) or object instances (Hodan et al. 2019) in images, understand indoor scenes (Handa et al. 2016), and estimate the pose of humans (Varol et al. 2017) or objects (Tremblay et al. 2018), among other tasks.

PointNets and similar architectures. The architecture of our classifier relates to an emerging body of work, encoding inputs that are given as sets (Qi et al. 2017a; Zaheer et al. 2017). Similarly to PointNet (Qi et al. 2017a), it employs pooling in order to obtain a representation that is invariant to the order of the elements, following a local computation at each element. It has previously been shown in (Qi et al. 2017a; Zaheer et al. 2017) that, under mild conditions, such pooling is the only way to achieve this invariance. Other contributions in the area of shape classification include PointNet++ (Qi et al. 2017b), which employs local spatial relations, and PointCNN (Hua, Tran, and Yeung 2018), which applies spatial information in order to group the points prior to aligning them spatially to a grid where a convolution can be applied. While previous work mostly focused on the identification of 3D point clouds, we encode a 2D outline and benefit from information that arises from the order of points along the outline. In addition, projected profiles of 3D objects have an inner profile and an outer profile, and the separation between the two carries valuable information.

Data reweighting schemes. Boosting techniques often iteratively weight harder samples, which are misclassified during training, more than other samples (Freund and Schapire 1997). In detection, such hard negatives are of great importance (Viola and Jones 2001), and, as suggested more recently by the focal loss method of (Lin et al. 2020), assigning different weights to the loss of different examples can significantly improve training. Another common reason for introducing weights into the loss function is class imbalance of the available samples, and it is common to assign higher weights to less frequent classes. The reweighting scheme that we propose here addresses both the difficulty of correctly classifying a sample from a given class, as well as the frequency of the current classification of a sample. While the classification difficulty component is somewhat similar to other methods, the other component is—as far as we can ascertain—completely novel.

Our reweighting scheme improves, not just the top-1 result, but also the top-k and is, accordingly, related to recent methods in this field. While it has been proven that the softmax-based cross entropy loss is optimal for every k under i.i.d. sampling and infinite data assumptions (Lapin, Hein, and Schiele 2018), these assumptions do not hold in our case in which there is a significant domain shift between train and test data. Recently, a method was proposed to overcome the infinite data assumption by employing a smoothed variant of a novel top-k SVM formulation (Berrada, Zisserman, and Kumar 2018). This method, however, does not account for domain shift. Unsupervised domain adaption (Mansour, Mohri, and Rostamizadeh 2009) techniques are designed to overcome such shifts. However, these methods, including the recent adversarial-training-based ones following (Ganin et al. 2016), are likewise unsuitable for our case since we do not possess a significant unsupervised set from the target domain.

Method

To generate synthetic training data using the sketches extracted from the catalogs, our process follows the following four steps: (I) Extraction of sketch lines from catalogs; (II) efficient generation of synthetic fracture faces; (III) point sampling; (IV) data augmentation. Performing steps II–IV effectively is non-trivial; see supplementary[1].

Network Architecture

Our OutlineNet is based on PointNet with several improvements. Unlike PointCNN and PointNet++, we do not attempt to cluster points together dynamically, but rather use the natural ordering of points along the outline to enrich the information at each point with more than just its spatial location.

In PointNet, the vector of each point goes through a series of 1D convolutions to generate a per-point feature. The network then applies a max-pooling layer to obtain a fixed-size feature vector, in a manner agnostic of the order of input points. We add two important items of information to each point: (1) annotation of inside/outside; (2) the angle of the outline at that point, which gives a rotation-invariant representation of the context around the point. Instead of representing this information as a quadruple $(x, y, \mathit{side}, \mathit{angle})$, which we found empirically to be ineffective, we suggest in what follows a novel approach, changing both the architecture and data representation throughout the network.

Group-hot encoding for side information. The side information is a categorical value; as such, it would typically be represented using a one-hot encoding. However, using one-hot encoding with inputs taking continuous values can cause problems when it differs significantly from the rest of the values. While the network can theoretically learn the proper weights to compensate for any scale, in practice this does not

[1] The supplementary material for this paper, is available at https://www.cs.tau.ac.il/%7Ewolf/papers/archaidesupp.pdf

always work. Instead, we suggest the following approach for combining categorical and continuous values—an approach we call *group-hot* encoding. To represent d continuous values coupled with one categorical value with c options, create a vector $v \in \mathbb{R}^{cd}$, representing c groups of d values. To represent group i, zero out the values of all but the ith group and store the d values in that group. For our two-value categorical information (inside/outside), the (x, y) location values would be represented as $(x_{in}, y_{in}, x_{out}, y_{out})$, where only one pair is nonzero each time.

Multi-feature and angle information. To encode the spatial context for each point, previous works construct hierarchies between points (Qi et al. 2017b; Hua, Tran, and Yeung 2018). In our case, the points are ordered, and we instead encode the immediate context around each point using angular information by considering, for every point, the cosine and sine of the angle formed at this point along the outline (encoding angle information directly suffers from the discontinuity at 0 and 2π). Angular information is secondary to the spatial information, and employing representations such as (x_{in}, y_{in}, $\sin\theta_{in}$, $\cos\theta_{in}$, x_{out}, y_{out}, $\sin\theta_{out}$, $\cos\theta_{out}$) showed little to no benefit in network performance. Instead, we employ a multi-pathway architecture to enable learning separate features for spatial- and angular-information.

This architecture (Fig 3) begins with two branches of multilayer perceptrons (MLPs), one for angles and one for positions. Both branches have the same shape: four hidden layers, with 64, 128, 128, 256 units, respectively. The outputs of these branches are then concatenated (per point) and fed into two perceptron layers of 512 and 1024 hidden units, respectively, to obtain a feature vector of length 1024 per point. Max pooling is then performed over all points to obtain a global feature vector of the same size. Going through an additional MLP (512, 256 and c hidden units) and a final softmax layer, we obtain the output scores for the c classes. All MLPs, except for the one producing the output score, employ ReLU activations. The MLP after max pooling employs a dropout with a rate of 0.7 after each layer, except for the last one. A batch size of 128 and an Adam optimizer (Kingma and Ba 2014) with an initial learning rate of 1×10^{-6} are used for training.

Loss Reweighting

Most common techniques for combating low class-accuracy introduce weights on the loss expressions of individual samples, with higher weights assigned to inputs low-accuracy classes. While the rationale is clear, there is no actual guarantee that it will cause the classifier to learn anything "meaningful" about these classes. For example, one way to push the accuracy of a given class upwards is to increase the bias of the logit to this class. While this uniformly increases the chance of all inputs to be classified in the class (including correct inputs, thus increasing the accuracy), this new classifier does not contain any new information compared to the previous one. This phenomenon, which can be identified by an accuracy increase accompanied by an increase in the number of false positives that are predicted to be in this class, was encountered numerous times during our research.

To mitigate this issue, we propose a new loss function, one that weights samples not just by their true label but also by predicted label. For each sample, the loss has one weight by the true label (assigning higher weights for classes with low accuracy) and another weight by the predicted label, assigning higher weights to misclassifications into classes with higher false-positive rates. The second weight is aimed at tackling an increase in class accuracy, accompanied by an increase in the number of misclassifications into the same class. As it turns out, the proposed new loss not only increases the uniformity of the accuracy among the classes, but also increases the overall performance on the test set. We attribute this to the fact that, during testing, the same types of confusions that occur in the training data are likely to occur, only more frequently.

Let $X = (x_1, x_2, \dots, x_n)$ be the set of inputs to a specific batch, and let $Y = (y_1, y_2, \dots, y_n)$ be their respective labels, where $y_i \in \{1, 2, \dots, c\}$. Let $f(x_i) = (f^1(x_i), \dots, f^c(x_i))$ be the probability vector predicted by the model f, and let $\widehat{y}_i = \arg\max f(x_i)$ be the class predicted by the network for input sample x_i. As the underlying loss, we employ the conventional cross entropy loss $\ell_i(f) = -\log f^{y_i}(x_i)$ for classifier f and input sample i. The new per-sample loss function, called *CareLoss* as an antonym of "neglecting" some classes, is denoted $\tilde{\ell}_i(f)$. It is created by weighting $\ell_i(f)$ by two weights u and v, which are associated with its ground truth label and the predicted label, respectively:

$$\widehat{u}(f, y_i) = \exp(-\alpha_u \psi(f, y_i)) \quad (1)$$

$$\widehat{v}(f, \widehat{y}_i) = \exp(+\alpha_v \rho(f, \widehat{y}_i)) \quad (2)$$

$$u(f, y_i) = \frac{\widehat{u}(f, y_i)}{\sum_j \widehat{u}(f, j)} \quad (3)$$

$$v(f, y_i, \widehat{y}_i) = \frac{1}{\eta}\left(1 + [y_i \neq \widehat{y}_i] \frac{\widehat{v}(f, \widehat{y}_i)}{\sum_j \widehat{v}(f, j)}\right) \quad (4)$$

$$\tilde{\ell}_i(f) := \tilde{\ell}(f, y_i, \widehat{y}_i) = u(f, y_i) v(f, y_i, \widehat{y}_i) \ell_i(f) \quad (5)$$

Here, α_u and α_v are positive parameters, $\psi(f, j)$ is the accuracy of the classifier f over the inputs originating from class j, $\rho(f, j)$ is the false-positive rate of the classifier f into class j (the ratio between the number of samples that are classified falsely into class j and the total number of misclassified samples), and η is a normalization parameter ensuring that, per batch, the v terms sum to one. In other words, we define unnormalized weights $\widehat{u}$ and $\widehat{v}$ based on the accuracy of the true label and the prevalence of the false positive cases that result in the predicted label and then convert these to pseudo-probabilities u and v. The indicator $[y_i \neq \widehat{y}_i]$ is 1 for a sample that is classified incorrectly, 0 otherwise, and the weight $v \neq \eta^{-1}$ only for misclassified samples.

Note that the signs are such that we up the weight of samples from classes with low accuracy and samples that are falsely predicted to be of classes with high false-positive rate. We especially pay attention to samples of neglected classes that are mapped to one of the classes that are often predicted.

Values $\psi(f, j)$ and $\rho(f, j)$ are computed empirically: $\psi(f, j)$ is the ratio of training samples from class j classified

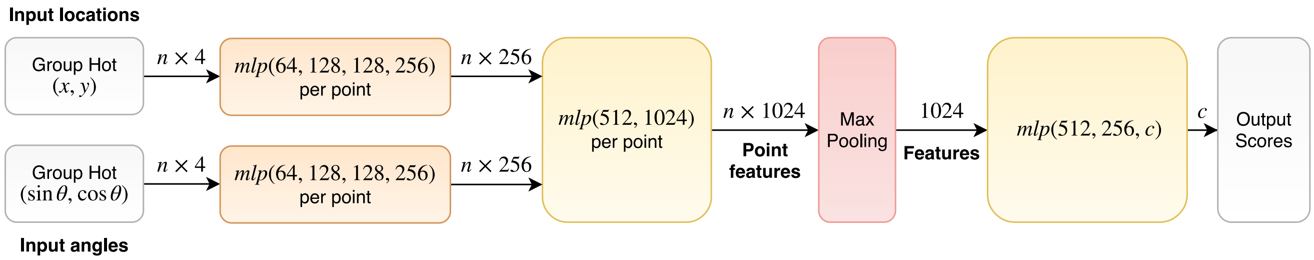


Figure 3: Network architecture, consisting of two pathlines: location and angle.

as class j, and $\rho(f, j)$ is the ratio of misclassified training samples that were incorrectly classified as class j. During training, the weights u, v are updated periodically every b batches, using a moving average with momentum γ to avoid sharp changes in the loss. The underlying class accuracies are recorded over the b batches, and the counters are reset after every weight update, to reflect an updated state of the classification confusion. The parameters used are $b = 50$, $\gamma = 0.8$, $\alpha_u = 6$, and $\alpha_v = 5$.

Experiments and Results

While most reported methods in the literature employ test data that is available during the development stages, this may cause overly optimistic results due to multiple hypothesis testing and other biases. This poses an increased danger in domains in which the datasets are not always large, including archaeology. The development of the reference tool was, therefore, planned as a two phase process, which includes a lengthy validation process. In the first phase, the methods were developed on potsherds of one family, collected from the same geographical region. In the second phase, additional datasets were provided, each with its own set of classes and defined train and test splits.

Phase I experiments. In the first phase, the classification task is to classify potsherds of terra sigillata italica (TSI) into one of 65 standardized top-level classes defined in the *Conspectus* catalog (Ettlinger 2002). These top-level classes are defined by 435 sketches. Each class has 1–8 associated sketches, from which class-balanced synthetic data is generated. The outlines of the real-world sherds, used exclusively for testing, were extracted from images collected across Europe using a dedicated mobile app. As part of the outline extraction, the user annotates outline segments as inner or outer, which is easily inferred by archeologists.

The real-world test dataset contains 240 extracted outlines from 29 different top-level classes. Nevertheless, we train our classifier on all 65 classes. Since the real-world test set is unbalanced, we report mean accuracy across classes. We also report results on a synthetic test set, which is obtained without the augmentation we apply to the training set, making it “easier” in this sense than the training data. Unless otherwise mentioned, all runs use our CareLoss function.

Table 1 reports the experimental results. As may be noted, our OutlineNet’s real-world top-2 classification rate is 1.5 times the top-1 classification rate. This indicates that the classes are easily confused, as can be seen in Fig 4.

We compare our OutlineNet to various baselines. PointNet (Qi et al. 2017a) and PointCNN (Hua, Tran, and Yeung 2018) results are given for the 8D feature vector $(x_{in}, y_{in}, \sin\theta_{in}, \cos\theta_{in}, x_{out}, y_{out}, \sin\theta_{out}, \cos\theta_{out})$ described in Sect. Network Architecture, or to the 2D points, as these methods were originally conceived. When applied to 8D inputs, we enlarge the capacity of these methods, and use the same number of parameters for PointNet as for OutlineNet. Data augmentation was used in all the experiments unless specified otherwise.

In the first set of comparisons, CareLoss is used throughout. As can be seen, PointNet does well on synthetic data. However, it is not competitive with OutlineNet on real-world data when using the 8D features. With the 2D points, PointNet is slightly better in top-1 accuracy than OutlineNet, but not on other top-k accuracies. PointCNN is not competitive in these experiments, showing lower training accuracies than other methods and complete failure in generalization to both test and real-world data. PointCNN implicitly requires normalized data, and we therefore retrained it with normalized data, where the sherd outline radius is scaled to fit the unit circle. As shown in the table, this did not provide any significant improvement. Another possible factor for the failure of PointCNN is the fixed sample counts that it requires, which prevent the application of the adaptation to sampling resolution we gave (Supplementary).

These experiments were repeated without CareLoss. The results for OutlineNet deteriorate along all real-world measurements, excluding the top-2 result. A similar effect is seen for PointNet on the real-world test data for both the 2D and 8D configurations. For PointCNN, which is performing almost at random for the real-world data, the results are similar without CareLoss and are omitted.

To further study the effect of CareLoss, we also compare it to variants where either u or v was set to 1 (so there is only one weight) and to focal loss reweighting (Lin et al. 2020) with the recommended parameter of $\gamma = 2$. The results show that dual weighting is important for real-world top-k results, and especially for the top-5 and top-10 ones. Focal Loss is consistently ineffective for our problem, and it also does not seem beneficial to combine it with CareLoss.

Another group of ablations tests architectural modifications: (i) The row “OutlineNet w/o separation of in/out (inner and outer outlines)” skips the group hot encoding of

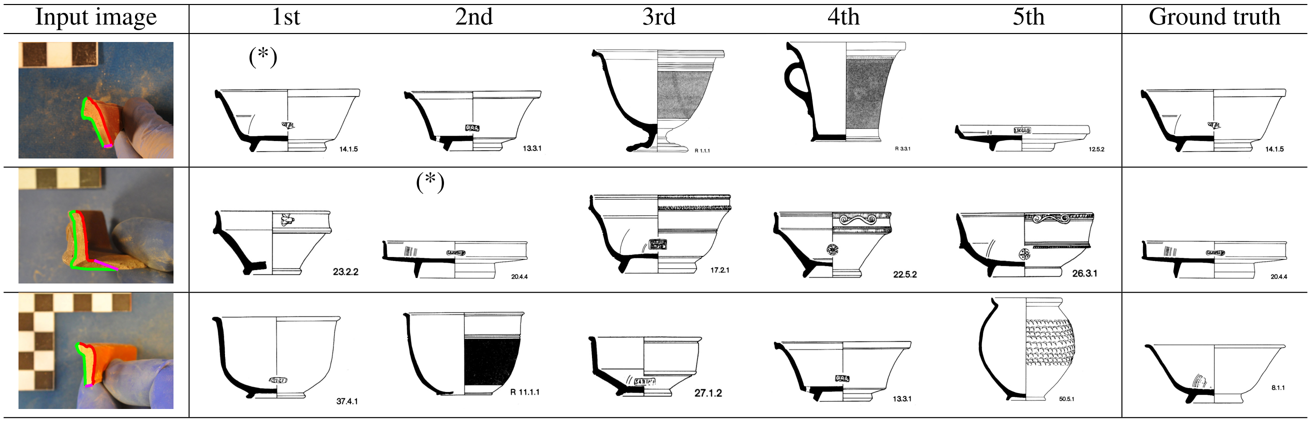


Figure 4: Example sherds and their top-5 results with our model. (*) indicates correct.

	Synthetic data		Real-world test data			
Method	Train	Test	Top 1	Top 2	Top 5	Top 10
OutlineNet	60.9	70.0	22.0	32.7	57.9	73.7
PointNet 8D features	54.4	71.3	16.4	26.9	44.5	65.2
PointNet 2D points	50.5	71.1	23.1	31.3	52.4	72.9
PointCNN 8D features	23.8	2.9	0.0	1.3	2.2	7.1
PointCNN 2D points	45.4	2.8	0.0	3.8	9.7	19.4
PointCNN 2D points, unit radius	23.7	2.8	2.2	2.7	9.6	14.8
OutlineNet w/o CareLoss	63.6	74.1	21.8	33.5	51.5	70.3
PointNet w/o CareLoss, 8D features	57.9	72.9	12.8	22.6	41.9	61.3
PointNet w/o CareLoss, 2D points	49.9	68.7	19.0	28.4	50.8	71.1
OutlineNet w/ CareLoss reweighting with u	57.6	67.3	21.6	31.4	50.0	68.2
OutlineNet w/ CareLoss reweighting with v	60.0	70.8	21.5	31.0	49.4	67.5
OutlineNet w/ Focal Loss (Lin et al. 2020)	57.2	69.6	17.3	28.3	48.2	61.1
OutlineNet w/ Focal Loss + CareLoss	57.1	67.3	20.2	28.1	46.3	66.5
OutlineNet w/o separation of in/out	52.9	69.6	22.8	32.0	50.4	65.7
OutlineNet w/o angle information	46.8	69.4	20.3	30.1	49.5	66.7
OutlineNet w/o group-hot encoding	53.0	66.3	19.7	28.5	47.9	65.0
OutlineNet w/ a single pipeline	49.1	68.8	18.6	27.6	48.9	66.4
OutlineNet w/o data augmentation	86.0	84.6	22.9	32.7	53.1	67.3

Table 1: Mean classification accuracy (in %) across classes on the TSI dataset.

points and angles (effectively discarding the group information); (ii) "OutlineNet w/o angle information" denotes dropping the pipeline of processing angles; (iii) "OutlineNet w/o group-hot encoding" appends the group id as a one-hot vector of size 2; and (iv) "OutlineNet w/ a single pipeline" is similar to PointNet working on 8D vectors but also incorporates adaptive sampling. As can be seen, all those modifications add to the overall performance when considering top-k. This is consistent with our finding for the 2D PointNet: the OutlineNet architecture presents its largest advantage after the top result. This befits the use as a reference tool for domain experts who would be happy to consider a short list of results as part of the mandatory expert verification, but would be discouraged by a tool that often completely omits the correct answer.

A similar phenomenon can be observed when training without augmentation. As expected, performance on the synthetic data is increased. Performance for top-1 and top-2 results on the real data is at least as good. However, going beyond $k = 2$, the advantage of augmentation becomes very clear. A plausible reason is that augmentation helps less with samples that are carefully curated and informative than it helps with the lower-quality ones.

In addition to mean performance, we also observe the standard deviation (SD) between classes of the performance to test if the overall mean success comes at the expense of classes whose performance is left behind, see Table 2. As expected, training without augmentation reduces the SD on the synthetic data but not on real data. OutlineNet without CareLoss, the PointNet 8D and the focal loss methods enjoys a relatively low SD on the top-1 result, but these methods have a relatively low performance there. OutlineNet presents a better SD for top-1 than the other effective methods. In the top-2 ranking, the only methods to show less

	Synthetic data		Real-world test data			
Method	Train	Test	Top 1	Top 2	Top 5	Top 10
OutlineNet	11.8	21.7	25.1	29.4	30.9	28.7
OutlineNet w/o CareLoss	15.5	23.8	23.4	31.8	35.1	31.1
OutlineNet w/o data augmentation	7.4	16.3	25.9	31.6	34.2	32.3
PointNet 8D features	11.4	19.1	22.6	30.7	34.5	33.8
PointNet 8D features w/o CareLoss	16.9	24.6	19.9	29.4	34.0	35.7
PointNet 2D points	10.4	20.2	27.7	31.8	30.7	28.8
PointNet 2D points w/o CareLoss	17.1	25.1	27.7	31.2	30.3	29.6
OutlineNet w/ CareLoss reweighting with u	11.3	20.4	26.8	31.9	32.6	33.3
OutlineNet w/ CareLoss reweighting with v	15.7	24.8	25.1	30.3	34.1	29.9
OutlineNet w/ Focal Loss	17.1	25.7	23.4	27.9	35.1	34.1
OutlineNet w/ Focal Loss + CareLoss	11.5	21.7	23.5	27.6	32.3	31.8

Table 2: Standard deviation (in %) of classification accuracy on the TSI dataset.

	Synthetic data		Real-world test data			
Dataset	Train	Test	Top-1	Top-2	Top-5	Top-10
TSI (Phase I) – 65 classes (29 in eval), 240 samples	60.9	70.0	22.0	32.7	57.9	73.7
TSI (Phase II, only test) – 65 classes (11 in eval), 96 samples	"	"	30.5	43.6	62.8	81.3
TSH – 98 classes (24 in eval), 218 samples	60.3	78.6	27.6	40.6	58.4	68.1
TSSG – 94 classes (34 in eval), 185 samples	57.7	76.6	14.5	25.0	41.9	59.9

Table 3: Mean classification accuracy (in %) across all classes for OutlineNet.

variance between classes than OutlineNet are the ones using Focal Loss with OutlineNet. However, these methods are not competitive in their mean performance. Finally, in the top-5 and top-10 measures, the SD of our method is better than all other methods, with the exception of the PointNet 2D variants at top-5, which are not competitive in this measure. To summarize: no method with a relatively good performance in a top-k measure also displays larger equality among classes than OutlineNet with CareLoss.

We set the parameters of the CareLoss early during development, before the architecture was finalized. However, analysis of the method's stability to its parameters shows similar results for a wide range of values (Supplementary).

Phase II datasets. Following the development of the complete method on the TSI dataset detailed above, we received three additional datasets. The first was an additional TSI test set, collected with the aid of the app. It included outlines of a further 96 actual sherds not included in the previous dataset of real data and belonging to 11 classes previously unseen during testing. Two additional datasets, terra sigillata hispanica (TSH) and South Gaulish terra sigillata (TSSG), were from different geographical origins and manufacturers, and belong to a different set of classes with different typologies.

Table 3 presents our model's results on the new TSI set, without performing any retraining or adaptation, using the same complete OutlineNet model from the previous experiments. The accuracy obtained is even better than the statistics for the prior real-world test set that was employed in Phase I. This further supports the claim of robustness for our methodology and its applicability as a genuine reference tool for archeologists. Table 3 also reports the results on the two additional datasets using our complete method (OutlineNet with CareLoss and data augmentation) with the same methodology and parameters. No tweaking whatsoever was performed for these datasets on any part of the training or classification processes. As can be seen, the pipeline generalizes well to TSH. While it also succeeds in learning for TSSG with similar train/test results on synthetic data, these evaluation results are a bit lower than for the other datasets. It seems that in many of the outlines of the TSSG dataset, the inner and the outer labels are incorrectly marked.

Conclusion

To tackle a real-world cross-modality matching problem that presents a large set of compounding challenges, we have conceived multiple innovations, including the novel data generation techniques, a new shape representation scheme, and a new re-weighting method. Our work also provides—beyond multiple technical novelties and a working application—a case study of deep learning applied to real-world data in a situation where most of the conventional assumptions are grossly violated. The reality gap (sim2real domain shift) is wide, and simulation must be performed with significant care. This is likely to be the case with many other historical and artistic classification problems. As some potsherd classes are visually identical and can only be distinguished using metadata from the excavation, the accuracy obtained in this paper is promising. An expert would then use such additional information to filter the algorithmic results. The method described here is already deployed in the field as the main component of an archeological reference tool. Source code for models and training is available at https://github.com/barak-itkin/archaide-software.

Acknowledgements

The project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 693548 (project ArchAIDE). We thank all the members of the ArchAIDE (archaide.eu) team.

Ethical Impact and Broad Societal Implications

Humankind's cultural heritage is an essential component of our identity as individuals and communities. As technology is precipitating societal changes at an accelerating pace, well preserved and properly identified archeological relics provide an anchor to a shared past. However, very often sherds that are excavated in emergency digs during development work are misidentified or incompletely classified, leading to a loss of cultural knowledge.

More broadly, the application of AI in the humanities is still in its infancy. The tools that exist today are frequently criticized for neglecting the research culture of the application field, for being myopic regarding their hidden assumptions, and for overstating the uniqueness and significance of computational outcomes. There is, accordingly, a dire need for tools that are developed hand-in-hand with the relevant humanities scholars and for their own use. Moreover, most AI in the humanities deals with texts, and some concerns art, whereas, to date, there has been very little research regarding other artifacts as they are reflected in archeology and anthropology.

Indeed, there are precious few examples of real-world AI tools that are in actual use for the humanities. The methodologies used to develop expert systems in medicine or self-driving cars, for example, are only partially applicable to humanities research.

Our work is exemplary in its multiple aspects of leveraging minimal training data for state-of-the-art deep learning, including three with far-reaching impact: (1) developing a network for one test dataset and applying it as is to other datasets without the need to re-tune any parameters; (2) working with a far removed description of the categories in order to make real-world identification of artifacts that contain little characteristic information; and (3) providing a new methodology for avoiding the misclassification of underrepresented classes. This last item epitomizes the difference between AI researchers, who often optimize accuracy, and humanities researchers, who are more concerned with avoiding the neglect of concealed knowledge.

References

Andrychowicz, M.; Baker, B.; Chociej, M.; Jozefowicz, R.; McGrew, B.; Pachocki, J.; Petron, A.; Plappert, M.; Powell, G.; Ray, A.; et al. 2018. Learning Dexterous In-Hand Manipulation. *arXiv preprint* arXiv:1808.00177.

Anichini, F.; Banterle, F.; Buxeda i Garrigós, J.; Callieri, M.; Dershowitz, N.; Lucendo Diaz, D.; Evans, T.; Gattiglia, G.; Letizia Gualandi, M.; Hervás, M. A.; Itkin, B.; Madrid i Fernández, M.; Miguel Gascón, E.; Remmy, M.; Richards, J.; Scopigno, R.; Vila, L.; Wolf, L.; Wright, H.; and Zallocco, M. 2020. Developing the ArchAIDE Application: A Digital Workflow for Identifying, Organising and Sharing Archaeological Pottery Using Automated Image Recognition. *Internet Archaeology* 52.

Banterle, F.; Itkin, B.; Dellepiane, M.; Wolf, L.; Callieri, M.; Dershowitz, N.; and Scopigno, R. 2017. VASESKETCH: Automatic 3D Representation of Pottery from Paper Catalog Drawings. In *Int. Conf. on Document Analysis and Recognition*, 683–690.

Barreau, J.-B.; Nicolas, T.; Bruniaux, G.; Petit, E.; Petit, Q.; Gaugne, R.; and Gouranton, V. 2014. Ceramics Fragments Digitization by Photogrammetry, Reconstructions and Applications. In *International Conference on Cultural Heritage (EuroMed)*, volume abs/1412.1330.

Berrada, L.; Zisserman, A.; and Kumar, M. P. 2018. Smooth Loss Functions for Deep Top-k Classification. In *International Conference on Learning Representations (ICLR)*. OpenReview.net.

Calin, N.; Sorin, P.; Daniela, P.; and Razvan, M. 2012. Using Reverse Engineering in Archaeology: Ceramic Pottery Reconstruction. *Journal of Automation, Mobile Robotics and Intelligent Systems* 6(2): 55–59.

Durham, P.; Lewis, P. H.; and Shennan, S. J. 1995. Artefact Matching and Retrieval Using the Generalised Hough Transform. In Wilcock, J.; and Lockyear, K., eds., *Proceedings of Computer Applications in Archaeology*, 25–30.

Ettlinger, E. 2002. *Conspectus formarum terrae sigillatae Italico modo confectae*. Bonn: Habelt.

Fischer, P.; Dosovitskiy, A.; Ilg, E.; Häusser, P.; Hazırbaş, C.; Golkov, V.; Van der Smagt, P.; Cremers, D.; and Brox, T. 2015. Flownet: Learning Optical Flow with Convolutional Networks. *arXiv preprint* arXiv:1504.06852.

Freund, Y.; and Schapire, R. E. 1997. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences* 55(1): 119–139.

Ganin, Y.; Ustinova, E.; Ajakan, H.; et al. 2016. Domain-Adversarial Training of Neural Networks. *J. Mach. Learn. Res.* 17: 1–35.

Gilboa, A.; Karasik, A.; Sharon, I.; and Smilansky, U. 2004. Towards Computerized Typology and Classification of Ceramics. *Journal of Archaeological Science* 31(10): 681–694.

Gupta, A.; Vedaldi, A.; and Zisserman, A. 2016. Synthetic Data for Text Localisation in Natural Images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2315–2324.

Handa, A.; Patraucean, V.; Badrinarayanan, V.; Stent, S.; and Cipolla, R. 2016. Understanding Real World Indoor Scenes with Synthetic Data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4077–4085.

Hodan, T.; Vineet, V.; Gal, R.; Shalev, E.; Hanzelka, J.; Connell, T.; Urbina, P.; Sinha, S. N.; and Guenter, B. 2019. Photorealistic Image Synthesis for Object Instance Detection. *arXiv preprint* arXiv:1902.03334.

Hua, B.-S.; Tran, M.-K.; and Yeung, S.-K. 2018. Pointwise Convolutional Neural Networks. In *Conf. on Computer Vision and Pattern Recognition (CVPR)*, 984–993. IEEE.

Kampel, M.; and Sablatnig, R. 2006. 3D Data Retrieval of Archaeological Pottery. In Zha, H.; Pan, Z.; Thwaites, H.; Addison, A. C.; and Forte, M., eds., *Interactive Technologies and Sociotechnical Systems: 12th International Conference, VSMM 2006, Xi'an, China, October 18–20, 2006. Proceedings*, 387–395. Berlin: Springer. ISBN 978-3-540-46305-4.

Karasik, A. 2010. A Complete, Automatic Procedure for Pottery Documentation and Analysis. In *Conf. on Computer Vision and Pattern Recognition, Workshops*.

Karasik, A.; and Smilansky, U. 2011. Computerized Morphological Classification of Ceramics. *Journal of Archaeological Science* 38(10): 2644–2657.

Kingma, D. P.; and Ba, J. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint* arXiv:1412.6980.

Lapin, M.; Hein, M.; and Schiele, B. 2018. Analysis and Optimization of Loss Functions for Multiclass, Top-k, and Multilabel Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40(7): 1533–1554.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2020. Focal Loss for Dense Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42(2): 318–327.

Lucena, M.; et al. 2014. Applying Mathematical Morphology for the Classification of Iberian Ceramics from the Upper Valley of Guadalquivir River. In *Mexican Conf. Pattern Recognition*, volume 8495 of *Lecture Notes in Computer Science*. Cham: Springer.

Makridis, M.; and Daras, P. 2013. Automatic Classification of Archaeological Pottery Sherds. *J. Comput. Cult. Herit.* 5(4): 15:1–15:21.

Malik, J. 1987. Interpreting Line Drawings of Curved Objects. *International Journal of Computer Vision* 1(1): 73–103.

Mansour, Y.; Mohri, M.; and Rostamizadeh, A. 2009. Domain Adaptation: Learning Bounds and Algorithms. In *Conference on Learning Theory*. URL http://www.cs.mcgill.ca/~colt2009/papers/003.pdf.

Peng, X. B.; Andrychowicz, M.; Zaremba, W.; and Abbeel, P. 2018. Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 1–8. IEEE.

Poblome, J.; Bes, P.; and Piccoli, C. 2013. Towards the Automatic Classification of Pottery Sherds: Two Complementary Approaches. In *CAA Conference Proceedings*, 463–474.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. Pointnet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Conf. on Computer Vision and Pattern Recognition (CVPR)*, 652–660. IEEE.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Advances in Neural Information Processing Systems (NIPS) 30*, 5099–5108. Curran Associates.

Tan, J.; Zhang, T.; Coumans, E.; Iscen, A.; Bai, Y.; Hafner, D.; Bohez, S.; and Vanhoucke, V. 2018. Sim-to-real: Learning Agile Locomotion for Quadruped Robots. *arXiv preprint* arXiv:1804.10332.

Tian, C.; Masry, M. A.; and Lipson, H. 2009. Physical Sketching: Reconstruction and Analysis of 3D Objects from Freehand Sketches. *Computer-Aided Design* 41(3): 147–158.

Tremblay, J.; To, T.; Sundaralingam, B.; Xiang, Y.; Fox, D.; and Birchfield, S. 2018. Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects. In *Conference on Robot Learning*, 306–316.

Varol, G.; Romero, J.; Martin, X.; Mahmood, N.; Black, M. J.; Laptev, I.; and Schmid, C. 2017. Learning from Synthetic Humans. In *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Viola, P.; and Jones, M. 2001. Rapid Object Detection Using a Boosted Cascade of Simple Features. In *Conf. on Computer Vision and Pattern Recognition*, volume 1. IEEE.

Wang, Y.; Liu, J.; Chen, Y.; and Tang, X. 2009. 3D Reconstruction of Curved Objects from Single 2D Line Drawings. In *Conf. on Computer Vision and Pattern Recognition Workshops*.

Xu, B.; Chang, W.; Sheffer, A.; Bousseau, A.; McCrae, J.; and Singh, K. 2014. True2Form: 3D Curve Networks from 2D Sketches via Selective Regularization. *ACM Trans. Graph.* 33(4).

Yin, X.; Wonka, P.; and Razdan, A. 2009. Generating 3D Building Models from Architectural Drawings: A Survey. *Computer Graphics and Applications* 29(1): 20–30.

Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Poczos, B.; Salakhutdinov, R. R.; and Smola, A. J. 2017. Deep Sets. In *NIPS*, 3394–3404.