Sigma Journal of Engineering and Natural Sciences
Web page info: https://sigma.yildiz.edu.tr
DOI: 10.14744/sigma.2025.00153

Research Article

Classification of ancient coins in archaeology using a novel deep learning approach: Bayesian convolutional neural network

Ebru PEKEL ÖZMEN[1,*], Soner ÖZMEN[2], Mehmet Emre SERTKAYA[3], Tuncay ÖZCAN[4], Vedat KELEŞ[5]

[1]*Department of Industrial Engineering, Samsun University, Samsun, 55000, Türkiye*
[2]*Department of Archaeology, Ondokuz Mayıs University, Samsun, 55270, Türkiye*
[3]*Distance Education Application and Research Center, Samsun University, 55000, Samsun, Türkiye*
[4]*Department of Management Engineering Department, Istanbul Technical University, Istanbul, 34467, Türkiye*
[5]*Department of Archaeology, Ondokuz Mayıs University, Samsun, 55270,Türkiye*

ARTICLE INFO

Article history
Received: 04 April 2024
Revised: 03 July 2024
Accepted: 24 September 2024

Keywords:
Archaeology; Bayesian Optimization; Convolutional Neural Network; Dating; Deep Learning

ABSTRACT

This study looks at classifying and dating old coins. Coins are important in archaeology because they tell us about history, culture, and economy. Knowing the right date of coins helps to understand excavation sites and also helps studies in art, politics, and social life. Normally, numismatics experts do this work, but it takes a lot of time and their judgment can be different. In this research, we used some deep learning models like DenseNet-201, GoogLeNet, InceptionV3, MobileNetV2, and Xception. We also tested a new model called Bayesian Convolutional Neural Network (B-CNN). This model uses Bayesian optimization to choose parameters. The B-CNN reached about 97% accuracy, which is better than the other models.
The results show that B-CNN can be a good tool for archaeologists, especially for dating coins. It gives more clear and correct results and reduces the need for special experts. The new part of this study is mixing Bayesian optimization with CNNs. This makes the model stronger than older methods. The work connects archaeology and computer science and shows better sensitivity and performance, but it also needs more training time.

Cite this article as: Pekel Özmen E, Özmen S, Sertkaya ME, Özcan T, Keleş V. Classification of ancient coins in archaeology using a novel deep learning approach: Bayesian convolutional neural network. Sigma J Eng Nat Sci 2025;43(4):1580–1591.

INTRODUCTION

One of the fundamental working concepts of archaeology is the requirement to appropriately date the material uncovered. The ancient coins, which were discovered in vast quantities during the excavations, give vital information for the identification and date of the region where they were discovered, as well as other archaeological material in the area [1]. Many archaeological works are interpreted with a wide range of dates rather than a specific period, because their building technique and aesthetic qualities have been

***Corresponding author.**
*E-mail address: ebru.pekel@samsun.edu.tr

This paper was recommended for publication in revised form by Editor-in-Chief Ahmet Selim Dalkilic

maintained for years. On the other hand, the ancient coins can be dated in time intervals, sometimes up to several years, by interpreting the figures, inscriptions, and monograms on them as a result of the political, military, religious, economic, and social events that the politically dominant people or the ruler experienced with the society in a short period of time [2, 3]. Experts (numismatists) do coin recognition and dating because they give thorough and trustworthy information on the location and/or geography. The identification and dating of coins are important since it directly helps sub-studies such as pottery, art, architecture, politics, religion, geography, social and economic life, and notably archaeology.

Traditionally, experts in numismatics (coin specialists) have been responsible for the recognition and dating of these coins, as they provide thorough and reliable information about the geography and time period of their origin. The identification and dating of coins are crucial because they contribute directly to sub-disciplines such as pottery, art, architecture, politics, religion, geography, social and economic life, and, most importantly, archaeology.

Classifying and dating coins is important, but it usually takes a long time and often depends on what experts think. Doing this work by hand also has problems. It needs a lot of historical knowledge and very close study of small details on the coins.

In the literature, there are only a few studies that use deep learning for coin classification and dating. Earlier studies mostly depended on traditional techniques or very basic machine learning models. To reach better accuracy and reliability, there is now a need to explore stronger deep learning methods, such as Convolutional Neural Networks (CNNs).

In this work, we tested different CNN models, including DenseNet-201, GoogLeNet, InceptionV3, MobileNetV2, and Xception. We also proposed a Bayesian Convolutional Neural Network (B-CNN), which uses Bayesian optimization to fine-tune its parameters. This work represents the first known use of Bayesian optimization in CNN-based ancient coin classification. The results suggest that the method reduces subjectivity, shortens the analysis process, and provides a stronger basis for interpretation. It can be a helpful support tool for archaeologists and numismatists. In this way, the study fills a gap in the literature and shows an interdisciplinary method that brings deep learning into archaeological research.

The remainder of this paper is organized as follows: In the second section, the literature review is presented. In the third section, the proposed approaches for the classification of ancient coins are introduced. The fourth section explained information which performance measures are selected for interpretation. The fifth part shows experimental results to compare the performances of the proposed approaches. Finally, the discussion and conclusions are presented.

LITERATURE REVIEW

In this study, the literature review was carried out in two different stages. In the first stage, a general review of the Convolutional Neural Networks method was conducted. Afterwards, Deep Learning studies in the archaeological area were reviewed. There is a few studies in the literature that applies deep learning algorithms to the classification problem of ancient coins.

Convolutional neural networks are now employed successfully in a variety of applications such as classification, pattern recognition and so on. There is no standard neural network model since the problem types differ. The difficulty of the problem and the structure of the neural network model are affected by factors such as the inputs, predicted outputs, and output form of the problem.

[4] conducted a study on the automatic recognition of ancient coin types using a semi-supervised learning method called Graph Transduction Games (GTG). This method deals with the difficulty of classifying ancient coins. The problem is hard because different coin types can look very similar, and coins from the same type can look very different. Another problem is that there are not many large labeled datasets.

The authors introduced a new dataset, RRC-60, which is an extension of a previous dataset, both in quantity and diversity. In practice, when dealing with old coins, researchers rarely trust a single glance. Most of the time, the coin

Table 1. Summarizes several research from the literature

Paper	Dataset	Method	Accuracy
(4)	Roman Empire	Graph Transduction Games (GTG)	97.4%
(5)	Roman Republican	Feature Fusion and Attention Mechanisms	98.5%
(6)	Roman Empire	MobileNetV3-L	95.2%
(7)	Roman Empire, Ancient Greece	Transfer Learning	98.32%
(8)	Turkish Republic	CNN, GAN	97.71%
(9)	Roman Empire	Graph Transduction Games (GTG)	87.2%
(10)	Roman Republican	Local Feature Matching, Geometric Consistency	83.3%
(11)	Roman Republican	radial-polar tiling scheme	91.68%

is turned over, examined again, and even compared with similar samples. This is a very ordinary step for a human expert, but when transferred into a model, it creates a rather clever mechanism. In one recent approach, the front and back sides of a coin were not treated separately, but instead as two sources of evidence that could "talk" to each other. The idea sounds almost obvious—if one side is blurred or worn, the other side may still carry useful details—but building this logic into a formal system is not trivial. By letting both sides support each other, the model manages to imitate the reasoning process of a person who is trying not to miss any clue. That is why the method has been described as game-theoretic, although in reality it simply reflects the back-and-forth nature of how experts already work.

The experiments showed that this method boosts performance in this difficult task, even when only a small number of labeled images are available.

Roman Republican coins were studied with a model called CoinNet, which effectively handled challenges caused by very old and damaged coins, especially in recognizing reverse side motifs such as objects, faces, animals, and buildings. A dataset of more than 18,000 images and 228 motif types was used, and the architecture—combining bilinear pooling, residual blocks, and attention layers—achieved 98.5% accuracy, significantly outperforming Bag-of-Visual-Words and traditional CNN models, thus demonstrating the advancement of modern deep learning methods [5].

Roman period coins were analyzed with deep learning models using the RRC-60 dataset, where Xception, MobileNetV3-L, EfficientNetB0, and DenseNet201 were tested. The best result was obtained by MobileNetV3-L with 95.2% accuracy, 98.2% Precision, and 96.8% Recall. The study also emphasized that considering both sides of a coin provides richer context and improves classification, much like how historians examine both sides before making judgments [6].

Ancient coins were also examined with local feature extraction and geometric checks combined with CNNs, which improved accuracy, particularly for visually similar coins [7].

A dataset named TurCoins, consisting of 11,080 images of Turkish Republic coins, was introduced, where ResNet50 with transfer learning was used and synthetic image generation for rare classes increased the accuracy to 97.71% [8]. Another approach, Graph Transduction Games (GTG), treated coin classification as a game where coins adopted labels based on peers, achieving 73.6% accuracy with one image per class and 87.2% with two on 180 Roman coins across 60 classes [9].

The ILAC project focused on Roman Republican coins with methods such as image matching, legend recognition, and coin recognition, reaching up to 83.3% accuracy on 4,100 coins, proving effective even for worn or damaged samples [10].

Motif recognition on the reverse side of Roman Republican coins was also conducted with a Bag of Visual Words (BoVWs) model, where radial-polar tiling provided the best performance with 91.68% accuracy [11].

Research Aim

When the literature studies are examined, it is seen that the classification of ancient coins using deep learning algorithms can be an innovative and important research topic. This study allows that the deep learning algorithms will be used for the first time to perform dating and classification tasks that have hitherto only been carried out by humans. As a result, this approach helps overcome limitations such as avoiding or overlooking areas that are difficult to access or hard to study with the naked eye for humanitarian reasons.

Moreover, this study marks the first attempt to analyze coins from two of the most significant periods of antiquity—the Greek and Roman eras—using deep learning methods. Compared with many other archaeological materials, coins stand out as especially valuable artifacts, since they preserve highly detailed and comprehensive information uncovered during excavations. Only experts (numismatists) can evaluate and interpret coins, which can only be identified after mechanical and chemical cleaning procedures, in addition to going through the phases of high-resolution photography and microscope analysis. While the tests conducted, it is possible to run into human factors like ignoring, neglecting, different interpretation, and quick-wrong conclusion. A specialist numismatist is not always present in excavations or projects for scientific archaeological investigations that are completed in a short amount of time, under challenging circumstances, and with few possibilities. At this point, this study presents a decision support system that can be used by non-technical experts for the classification of ancient coins.

In this study, a different perspective was taken by introducing the B-CNN model. The motivation behind developing this model was to see whether combining a standard CNN with Bayesian-based optimization could make the training process smarter and more efficient. Instead of relying only on fixed parameters, the model was allowed to "learn" better settings during the process. This way, the B-CNN does not simply copy what traditional CNNs already do, but rather tries to improve them by adapting to the data in a more flexible manner. The study shows that such an approach can open new directions for deep learning applications, especially in areas where accuracy and reliability are critical. Notably, the uniqueness of this study extends beyond archaeology; it also introduces novel contributions to image processing technologies.

MATERIALS AND METHODS

Pure Convolutional Neural Networks Models

Deep learning is a helpful technology that came about because regular neural networks got better. Deep learning

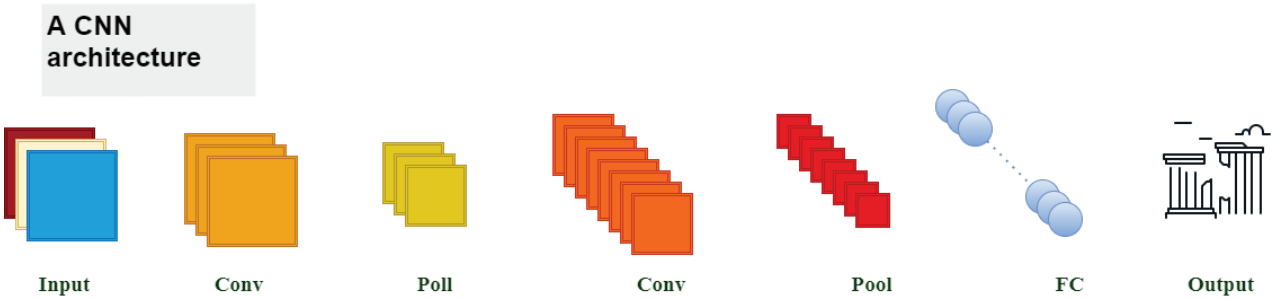

Figure 1. CNN architecture

is widely used in classification, image segmentation, identification, prediction, and detection. Deep learning algorithms can predict complex connections between inputs and outputs and manage large amounts of data. CNN is one of the most effective and extensively used deep learning algorithms. With the convolution process, the attributes of the input data are automatically retrieved and transferred to the following layers with these properties. The convolution method reveals the image's most significant features. Data is transferred through the next layer of the CNN's multilayered structure by performing a different operation on each layer as seen in Figure 1. Each layer has a distinct purpose.

The layers of CNN can be summarized as follows:

Convolution: In the convolution process, a matrix of any size, such as 3*3, 5*5, 7*7 so on passes along the image matrix. The previously described tiny size matrix cycles the entire image matrix and concretizes the image's attributes. After that, a new image matrix is created.

Pooling: After the convolution process, the pooling layer is frequently applied. A matrix of size n*n is generated with n determined by the original picture size. This empty matrix slides along the image matrix to obtain the biggest value in the relevant region of the image matrix and assigns a new image matrix's value. The volume of data handled in the network is also lowered in this way.

Flattening: This layer's goal is to prepare the data for the fully connected layer's entrance. Neural networks are typically fed data from a one-dimensional (1D) array. When it comes to the flattening layer, however, the data is in the form of two-dimensional (2D) matrices.

Fully connected: This layer's data is one-dimensional. In this layer, all neuron connections are complete. Each neuron is connected to the neuron in front of it. As a result, it is referred to as the fully connected layer. In ReLU layer, an activation procedure is carried out. The data is sent to the activation function, which returns a value. Many activation functions may be utilized in this layer, including hyperbolic tangent, sinus, and sigmoid.

Softmax: It calculates the classification process's probability value and allocates the class with the highest probability.

The CNN models used in this study are summarized in the following subsections.

DENSENET-201

Deep learning models started to show good results in image classification around 2012. At that time, shallow networks were replaced by deeper ones with many layers. Figure 1 shows the overall CNN model and its layers. The recently created Deep Learning networks have varied architectures and have found solutions to issues like gradient fading and numerous shortcut connections [12]. Contrary to other techniques, DENSENET-201 has improved the efficiency of all the characteristics created by the earlier layers by allowing them to enter the subsequent layer. By allowing access to all feature maps created by the previous layers, the deep layers in the networks can reuse the features [13]. The Densenet-201 structure is shown in Figure 2. The current layer can draw conclusions from earlier layers' features since all previous levels are related to one another [14].

GoogleNet

GoogleNet is a convolutional neural network (CNN) model with 22 layers. The model showed its success by

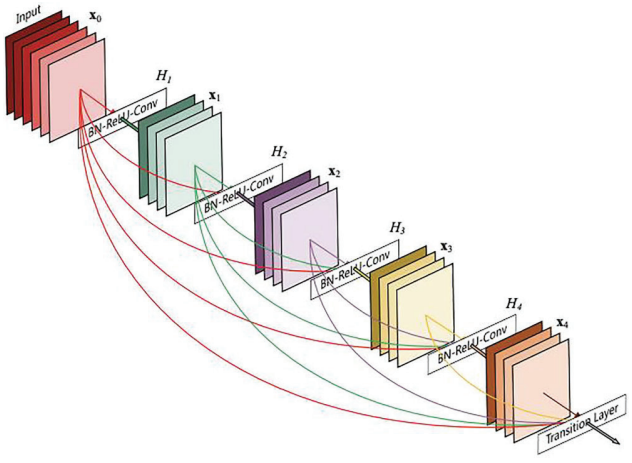

Figure 2. DENSENET-201 architecture [14] [created by the authors]

winning the ILSVRC competition held in 2014. Unlike Densenet-201 model in the GoogleNet architecture, multiple layers work in parallel at the same time. Parallel layers consist of variable sized convolution kernels, 1 × 1, 3 × 3 and 5 × 5. The purpose of using 1 x 1 filters used in the GoogleNet model is to reduce binary dimensionality and summarize the contents in the previous layer [15]. The initial branch of GoogleNet's branching structure is the 1 x 1 convolution kernel. A 1 x 1 convolution kernel and a 3 x 3 convolution kernel are combined in the second branch. A 1 x 1 convolution kernel and a 5 x 5 convolution kernel are combined in the third branch. The fourth branch consists of a convolution kernel and a 3 x 3 maximum pooling kernel. Four outputs are produced by the model's four branches, which process the input data. The final output, the feature map, is produced by combining these outputs with the channel dimensions [16].

Inception V3

The Inception V3 model, which entered the top 5 in ILSVRC-2015, is a 48-layer model. Since the model is initially sparse, it can produce few convolutions [17]. Inception V3 suggests a method to optimize the network and has a bigger mesh than Inception-V1 and V2 models [18]. Increasing computational efficiency in application instances and minimizing the impact of low parameter values on the model are two of the model's primary objectives. These characteristics make it quick to react and simple to use. Convolutional cores of various sizes are used by Inception V3 to enable it to have various receiver areas. Batch normalization (BN) is a layer that is used in Inception V3. Between the fully connected layer and the classifier, this layer serves as a moderator [19].

Mobilenet V2

A popular deep learning model for classifying images is MobileNet V2. It is so widespread because it functions on both mobile devices and computers with less processing power. The model has a controllable structure that allows it to regulate the parameters accuracy and latency [20]. The MobileNet V2 mesh consists of convolution layers each 3 x 3 deep, followed by a 1 x 1 point convolution layer that combines this filter output. Each of the 13 blocks in the previously stated structure contains a point convolution layer. The point convolution layer that MobileNet V1 utilizes allows it to reflect data with many channels to tensors with much fewer channels. Residual connectivity is another benefit of MobileNet V2, since it facilitates gradient flow across the network [21]. The network does not use the usual residual connections used to connect two extended units. Instead, inverted residual links provide the connection between thin blocks. The model deals with the possibility of representing the versatility obtained from a pixel in less dimensional space through channels by expanding the width factor [22].

Xception

Xception is an enhanced version of Inception V3. This model has the advantage of using channel correlation and spatial correlation separately. Therefore, Xception provides a deeply separable network model. There are two distinct components to the deeply separable convolution structure. Each input channel is subjected to independent spatial convolution after which point-to-point convolution is applied using 1 x 1 cores for point-to-point wrapping. The Xception structure is shown in Figure 3 [23]. There are 71 layers, 36 convolution layers, and 14 blocks in the Xception model. All layers, except for the first and last, are connected linearly. To preserve the sequential features within the network, a linear residual link is introduced as a shortcut. This approach helps to overcome both bottleneck and vanishing gradient problems. Instead of simply concatenating outputs, the method uses an aggregation process to pass information from one layer to the next, which improves stability and learning [24].

Bayesian Convolutional Neural Networks Approach

Bayesian optimization

Bayesian Optimization (BO) has been tried in many different areas, like physics, chemistry, and engineering, and it has shown good results. One reason it works well is that it can keep a balance: it explores new options while still using

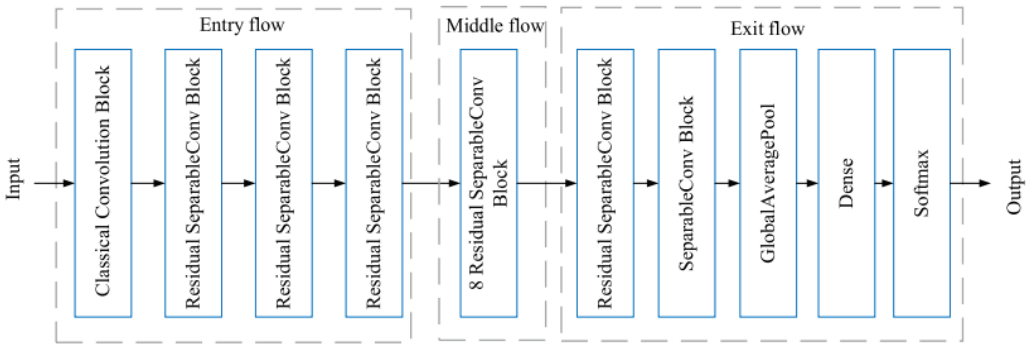

Figure 3. Xception architecture [23] [created by the authors]

the best choices it already knows. This balance is very helpful when decisions need to be made under uncertainty.

Another factor behind its popularity is the growth of open-source tools. These platforms made BO easier to use, and they also opened the door for applying it to multi-objective optimization and experimental design.

Overall, BO is both efficient with data and dependable in practice. Today it is seen as a useful method across many fields, and it has become an especially important tool in machine learning and artificial intelligence.

Bayesian Optimization typically relies on a probabilistic model, often modelled using a Gaussian Process (GP). The GP is a distribution over functions, defined by its mean function $\mu(x)$ and covariance function ($k(x,x')$). For a set of observed data $D = \{(x_i, y_i)\}$ where x_i represents input parameters and y_i represents the corresponding function values, the predictive distribution of the GP at a new point x_* is given by Eq.(1).

$$P(f_*| x_*, D)=N(\mu_*,\sigma_*^2) \quad (1)$$

Here, μ_* is the predictive mean, and σ_*^2 is the predictive variance.

The acquisition function guides the optimization process by balancing exploration and exploitation. One common acquisition function is the Expected Improvement (EI), defined as in Eq.(2).

$$EI(x) = E[\max(0, f(x) - f(x^+))] \quad (2)$$

where $f(x)$ is the mean of the GP, $f(x^+)$ is the best observed value so far, and the expectation is taken over the predictive distribution.

The steps of the BO are given in the following:

1. **Set up the surrogate model:** Start with some initial data D and build a Gaussian Process model that will act as an approximation of the real function.
2. **Choose the next point:** Use the acquisition function to decide where to sample next (x_next).
3. **Check the real outcome:** Evaluate the actual objective function at that point to get the result (y_next).
4. **Update and repeat:** Add this new information into the surrogate model. Keep repeating steps 2–4 until the method either converges or a chosen stopping rule says it's done.

The Gaussian Process is defined by Eq.(3-5).

$$(x)\sim gp\big(\mu(x), k(x,x')\big) \quad (3)$$

$$\mu(x) = K(X,x)^T[K(X,x) + \sigma_n^2 I]^{-1} y \quad (4)$$

$$K(x,x') = k_0 \exp(-\frac{1}{2}\sum_{d=1}^{D}\frac{(x_d + x'_d)^2}{l_d^2}) \quad (5)$$

where $K(X, x)$ is the vector of covariances between x and all points in the training set X, $K(X, X)$ is the covariance matrix for all pairs of points in X, σ_n^2 is the noise variance, y is the vector of function values in the training set, k_0 is a constant, and l_d are length scale parameters.

Bayesian Optimization leverages the mathematical foundation of Gaussian Processes and acquisition functions to efficiently explore and exploit the parameter space, making it a valuable tool for optimizing complex and computationally expensive objective functions.

A novel deep learning approach by bayesian convolutional neural network

This study initially establishes a simple Convolutional Neural Network (CNN) structure designed for image classification. In this work, we developed a CNN model with eight layers for image classification. The network takes RGB images at 227×227 resolution as input. This size was selected intentionally to shorten the training time and to make the model more practical on different computer systems.

The first convolutional stage applies small filters and is followed by normalization and ReLU activation, after which a pooling step reduces the feature dimensions. This same idea is repeated with more filters in the following layers so that the network can capture increasingly detailed visual patterns.

Once the convolutional part is complete, the extracted features are passed to a fully connected layer with two outputs. Finally, a softmax layer converts the outputs into probabilities, and the classification layer assigns each image to its class. In addition, we applied parameter optimization during training to improve both efficiency and accuracy.

The Bayesian CNN model incorporates Bayesian optimization to enhance the learning processes of traditional Convolutional Neural Network (CNN) architectures. This model is designed to identify the most effective values for important hyperparameters, including MaxEpochs (the maximum number of epochs), InitialLearnRate (the initial learning rate), and MiniBatchSize (the size of the mini-batches).

Bayesian optimization is commonly applied when tuning hyperparameters. Rather than testing values at random, the method evaluates how the model performs and keeps track of the outcomes. Using this information, it proposes the next set of values to examine. Through this gradual adjustment, the model is guided toward better performance in a more practical and less wasteful manner.

When settings like MaxEpochs, InitialLearnRate, and MiniBatchSize are adjusted, the Bayesian CNN can train faster. It also adapts well when evaluated on unseen data. In practice, this not only increases accuracy but also reduces the overall computational cost. As a result, the training process becomes both easier and more efficient.

We can describe the core math behind the B-CNN model as follows:

Algorithm 1. The pseudocode for B-CNN

```
datastore ← (datasetPath, 'IncludeSubfolders', true, 'LabelSource', 'foldernames')
[trainingSet, testSet] ← splitdata(datastore, 0.7, 'randomized')
inputSize ← [227 227 3]
trainingSet_Resized ← arrange_image_size(inputSize, trainingSet)
testSet_Resized ← arrange_image_size (inputSize, testSet)
objectiveFunction = @(params) cnnObjectiveFunction(params, trainingSet_Resized, testSet_Resized, testSet)
Define the parameter ranges for optimization:
param1 = optimizableVariable('MaxEpochs', [5, 50], 'Type', 'integer')
param2 = optimizableVariable('InitialLearnRate', [0.001, 0.1], 'Transform', 'log')
param3 = optimizableVariable('MiniBatchSize', [16, 64], 'Type', 'integer')
Create a Bayesian optimization object:
bayesOpt = bayesopt(objectiveFunction, [param1, param2, param3], 'maximum_iterations', 100)
Get the optimized parameters:
optimizedParams ← bestPoint(bayesOpt)
Train the model using the best parameters:
Layers ← Assign the layer to be optimized
Options ← trainingOptions('sgdm', ...
'MaxEpochs', optimizedParams(MaxEpochs), ...
'InitialLearnRate', optimizedParams(InitialLearnRate), ...
'MiniBatchSize', optimizedParams(MiniBatchSize))
Make predictions using the optimized model:
YPredOptimized = classify testSet_Aug by optimizedNet
YTest ← get the Labels from testSet
Create confusion_matrix (YTest, YPredOptimized);
Get the accuracy value
```

$f(x)$, represents an objective function measuring the performance of the CNN model, such as accuracy or loss function value (accuracy for this study). x, *represents hyperparameter values. For example*, x = (*MaxEpochs, InitialLearnRate, MiniBatchSize*).

Gaussian Process (GP) Model is constructed as seen in Eq. (6).

$$f(x) \sim GP(m(x), k(x, x')) \tag{6}$$

$m(x)$ is the mean function and $k(x,x')$ is the kernel function. On the other hand, the acquisition function is established as given in Eq. (7)

$$Acquisition\ Function: x_{next} = \arg\max EI(x) \tag{7}$$

x_{next} is the next hyperparameter suggestion in Eq. (7).

$$EI(x) = E[0, f(x_{best}) - f(x)] \tag{8}$$

$f(X_{best})$is the best performance value observed so far. A new hyperparameter set is chosen with x_{next}. This hyperparameter set is given to the CNN model, and the model is trained. The performance of the model is measured, and the objective function value is obtained. This value is added to the Gaussian Process model, updating the model. This process is repeated until a certain criterion is met (e.g., a specific number of iterations).

This process mathematically expresses the goal of Bayesian optimization to find the best hyperparameter set for improving the performance of a CNN model.

PERFORMANCE MEASURES

In a classification problem, classifiers can be compared using performance measures such as accuracy, precision, precision, and F-measure. The model's success is proportional to the number of samples assigned to the correct class and the number of samples assigned to the incorrect class in the test set. The confusion matrix is used to calculate the performance measures of the classifiers. In the confusion matrix shown in Table 2, the rows represent the actual values of the samples and the columns represent the predicted values [25].

Table 2. Confusion matrix

	Predicted		
Actual		**C0**	**Not-C0**
	C0	(TP)	(FN)
	Not-C0	(FP)	(TN)

The accuracy rate is defined as the ratio of true categorized samples (*TP* + *TN*) to total samples (*TP* + *TN* + *FP* + *FN*). Eq.(9) can be used to calculate the accuracy rate.

$$AR = (TP + TN)/(TP + TN + FN + FP) \quad (9)$$

Precision is defined as the ratio of True Positives (TP) samples predicted as class 1 to the total number of samples projected as class 1 (TP + FP). Eq.(10) can be used to calculate the precision.

$$P = TP/(TP + FP) \quad (10)$$

Sensitivity is defined as the ratio of properly categorized positive samples (TP) to total positive samples (TP + FN). Eq.(11) can be used to calculate the sensitivity.

$$S = TP/(TP + FN) \quad (11)$$

Another metric used to assess the success of classification algorithms is the F-measure. The F-measure is the harmonic mean of precision (P) and sensitivity (S), and it can be determined using Eq.(12).

$$F\_measure = (2 * P * S)/(P + S) \quad (12)$$

In practice, these metrics tell slightly different stories about the same model. Accuracy gives a quick overall picture, but precision and sensitivity (recall) point to opposite kinds of mistakes: precision punishes false positives, while sensitivity punishes false negatives. When we inspected a few borderline cases, the coins with heavy wear tended to hurt sensitivity more than precision, which fits the intuition that faint motifs are easier to miss than to hallucinate. For that reason, we report F-measure alongside accuracy to keep both effects in view.

APPLICATION

Parion is one of the most prominent port cities in the Troas region, which was colonized and built around 709 BC. It is situated in northwest Turkey, in the province of Çanakkale, in the Biga district, close to the village of Kemer. A large number of coins have been uncovered in the ancient city of Parion, where archaeological excavations and surveys have continued for more than two decades. These coins are mainly linked to the Roman and Greek periods, which were therefore chosen as the class labels in this study. Out of the 112 samples, 56 belong to the Roman class, while the remaining 56 are associated with the Greek class. For each coin, the width and height were recorded, and Figures 4–5 illustrate representative examples from both categories.

While training the models, 70% of the dataset was used for training and the rest for testing. When training models in deep learning, a considerable quantity of data is typically necessary. In this way, the model is aimed to increase performance by detecting more features. However, in our study, since the ancient coins belonging to 2 different eras are in a uniform structure, methods such as image reproduction led to overfitting.

Matlab R2021B 64 bit software was employed for analysis. The computer's characteristics include an AMD Ryzen 7 3800X 8-core CPU 3.90 GHz as the processor. RAM has a storage capacity of 32 GB. Matlab trained the models

Figure 4. Greek coins

Figure 5. Roman coins

using the graphics card processor. The graphics card on the machine is an NVIDIA GeForce RTX 3060 12 GB.

Figure 4 presents the performance of the pure CNN models for the training data. In this figure, the x-axis shows the number of iterations, and the y-axis shows the accuracy rate. As can be seen in Figure 5, Densenet-201, Googlenet and InceptionV3 models gave the best results, respectively. The accuracy of the models increased with the number of iterations.

The performance analysis of the pure CNN models for the test data is presented in Figure 6. As can be seen in Figure 7, when the models were tested, the Densenet-201 model performed the best. The models' validation baselines grew by 40–60% on average and varied linearly across 110 iterations. The accuracy rate of the Densenet-201 model for the test data is 95.54%. When compared to other models, Googlenet produced the second-best result, with a accuracy rate of 93.75%. The main factor in the Densenet-201 model's performance is that it eliminates fading gradients, which reduces feature loss. To do this, each layer creates a feature map that incorporates all of the preceding levels' characteristics. However, it is believed that the parallel layer design is the major component in producing good results, such as the Googlenet model, which ranks second. The reason that these two models outperform other models is that they strive to minimize feature loss. When the graphs of both models are reviewed, the gradient loss is kept to a minimal, allowing them to draw a linear trend.

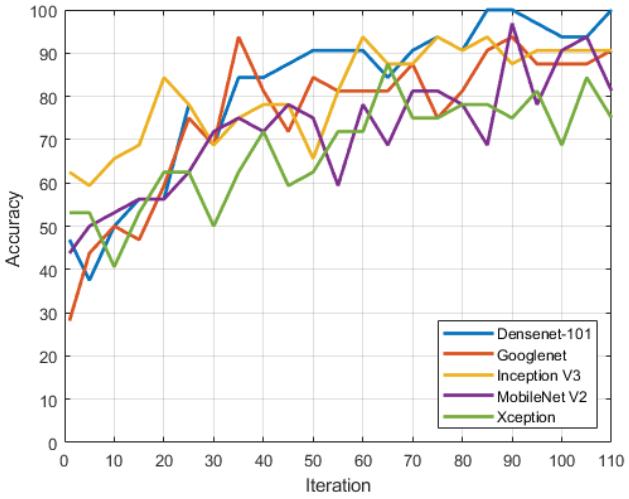

Figure 6. Performance analysis of the pure CNN models for the training data

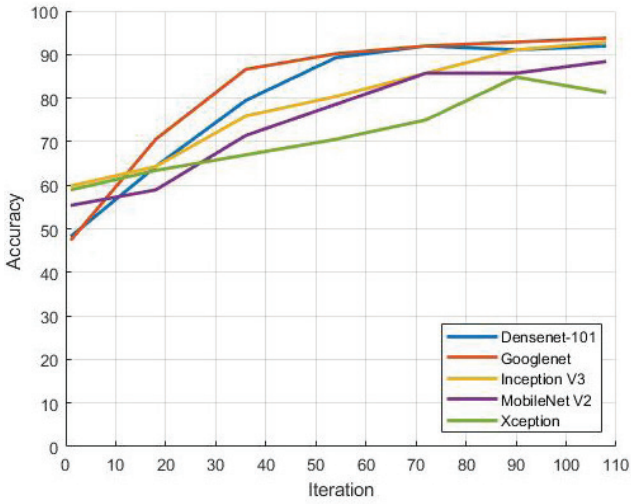

Figure 7. Performance analysis of the pure CNN models for the test data

The performance measures and run times of the pure models are presented in Table 3.

The proposed model within the scope of the study is the B-CNN model, and performance metric values associated with the best objective function obtained after a 100-iteration optimization process have been calculated.

The hyperparameter ranges utilized for the B-CNN approach are provided in the table below. Additionally, the best parameter values obtained at the end of the optimization process are also presented in this table.

Table 3. Hyperparameter to be optimized in B-CNN

Hyperparameter	Solution space	Optimum value
MaxEpoch	(5, 50)	8
InitialLearnRate	(0.001, 0.1)	0.0011289
MiniBatchSize	(16, 64)	35

Table 4. Performance analysis of the pure CNN models

MODEL	Se. (%)	Sp. (%)	Pre. (%)	F-Scr.	Acc (%)	Collapse Time
DENSENET-201	**98.11**	93.22	92.86	0.954	95.54	6 min 17 sec
GoogleNet	92.86	94.64	94.55	0.936	93.75	2 min 38 sec
InceptionV3	92.98	91.07	91.38	0.921	92.86	3 min 56 sec
MobilenetV2	85.71	91.07	90.57	0.880	88.39	5 min 8 sec
Xception	84.91	83.93	83.33	0.841	82.14	3 min 4 sec
B-CNN	94.44	**100**	**100**	**0.981**	**97.06**	15 min 3 sec

The optimal hyperparameter values obtained in Table 3 are provided. These values were obtained on the training set and applied to the test dataset as well.

Table 4 compares the results of different deep learning models for classifying ancient coins. The table shows Sensitivity (Se.), Specificity (Sp.), Precision (Pre.), F-Score, Accuracy (Acc.), and Collapse Time.

DenseNet-201 gave good results, with 98.11% sensitivity and 95.54% accuracy. It took around six minutes to train, which isn't the fastest, but it works. GoogleNet performed well, with 93.75% accuracy and 94.64% specificity, and it also trained very quickly—only 2 minutes and 38 seconds. InceptionV3 gave balanced results, showing 92.86% accuracy and 91.07% specificity.

MobileNetV2 and Xception performed less strongly. MobileNetV2 reached 88.39% accuracy, while Xception achieved only 82.14%. Although these models are faster, their accuracy is lower.

The Bayesian Convolutional Neural Network (B-CNN) produced the best outcomes, reaching 100% specificity and precision with about 97% accuracy. However, training took much longer, nearly 15 minutes. This indicates that B-CNN is highly dependable but requires more computational power. For situations where training speed is important, DenseNet-201 and GoogleNet are more practical options.

Overall, B-CNN delivered the highest accuracy and reliability, making it valuable for sensitive fields such as archaeology. Still, the longer training time needs to be considered when selecting the model.

If we take a closer look at the methods listed in Table 4, it quickly becomes apparent that no single approach can be considered flawless. Each has certain points where it performs well and others where it lags behind. For instance, DENSENET-201 often shows a strong outcome in terms of sensitivity, which makes it appealing when that metric is of central importance. However, this strength should not be interpreted as a guarantee of equally strong results in every other category. In fact, in some cases it may fall short, and this highlights the need to evaluate the model with a balanced view. On the other hand, GoogleNet is usually mentioned for its shorter training time. This can be an undeniable advantage when rapid experimentation or limited computing resources are part of the research setting. Its speed does not always guarantee the highest level of accuracy, and this kind of trade-off is something that scholars and practitioners alike need to weigh with care. After all, in most real research settings there is rarely a single "best" option that works under every condition. Much depends on what the study is trying to achieve and the constraints under which it is carried out. For some projects, efficiency and shorter training times may be the decisive factor, while in others the focus may shift toward maximizing precision, even if that requires more time and resources. Archaeological applications provide a good example of this dilemma: a misclassified artifact is not just a numerical error but could potentially alter the historical interpretation of an entire site. For this reason, accuracy often receives special emphasis in such contexts, though in time-sensitive studies a more balanced or pragmatic solution may still be considered acceptable. For some projects, efficiency and shorter training times may be the decisive factor, while in others the focus may shift toward maximizing precision, even if that requires more time and resources. Archaeological applications provide a good example of this dilemma: a misclassified artifact is not just a numerical error but could potentially alter the historical interpretation of an entire site. For this reason, accuracy often receives special emphasis in such contexts, though in time-sensitive studies a more balanced or pragmatic solution may still be considered acceptable. In other studies with stricter time constraints, however, the faster option could be more practical.

CONCLUSION

This study shows a good example of cooperation between archaeology and computer science. It proves that an interdisciplinary approach can help solve complex problems. Dating and classifying old coins, especially from Greek and Roman times, has always been difficult in archaeology and needs special experts. This research uses deep learning to make the process faster and less dependent on humans.

Different Convolutional Neural Networks were tested. Among them, DenseNet-201 gave the best result with 95.54% accuracy, so it was the strongest of the classical models. The main focus, however, was on the Bayesian Convolutional Neural Network (B-CNN). This model uses Bayesian optimization to improve CNN performance. After 100 iterations of optimization, the model reached 97.06% accuracy, with better sensitivity, specificity, and precision than the other models.

The B-CNN improved performance a lot, but training took longer. The collapse time was about 15 minutes, which can be a problem when time is limited. For this reason, the choice of model should depend on the needs of the archaeological work.

In conclusion, this study shows how deep learning can be applied in archaeology. The use of Bayesian optimization makes the method different from earlier studies and gives high accuracy and reliability. This interdisciplinary work creates a base for future research and opens new ways to study historical artifacts with artificial intelligence.

AUTHORSHIP CONTRIBUTIONS

Authors equally contributed to this work.

DATA AVAILABILITY STATEMENT

The authors confirm that the data that supports the findings of this study are available within the article. Raw

data that support the finding of this study are available from the corresponding author, upon reasonable request.

CONFLICT OF INTEREST

The author declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

ETHICS

There are no ethical issues with the publication of this manuscript.

STATEMENT ON THE USE OF ARTIFICIAL INTELLIGENCE

Artificial intelligence was not used in the preparation of the article.

REFERENCES

[1] Thonemann P. The Hellenistic World: using coins as sources. Cambridge: Cambridge University Press; 2015. [CrossRef]

[2] Atlan S. Grek sikkeleri. Istanbul: Arkeoloji ve Sanat Yayınları; 1993.

[3] Klawans ZH. Handbook of ancient Greek and Roman coins: Whitman Coin Pub; 1995.

[4] Aslan S, Vascon S, Pelillo M. Two sides of the same coin: Improved ancient coin classification using graph transduction games. Pattern Recogn Letters 2020;131:158-165. [CrossRef]

[5] Anwar H, Anwar S, Zambanini S, Porikli F. Deep ancient Roman Republican coin classification via feature fusion and attention. Pattern Recogn 2021;114:107871. [CrossRef]

[6] Kayaalp K, Özkaner F. Bi-directional classification of roman period coins by deep learning methods. Int J Eng Innov Res 2023;5:161-169. [CrossRef]

[7] Zambanini S, Kavelar A, Kampel M, editors. Classifying ancient coins by local feature matching and pairwise geometric consistency evaluation. 2014 22nd International Conference on Pattern Recognition; 2014: IEEE. [CrossRef]

[8] Temiz H, Gökberk B, Akarun L, editors. TurCoins: Turkish republic coin dataset. 2021 29th Signal Processing and Communications Applications Conference (SIU); 2021: IEEE. [CrossRef]

[9] Aslan S, Vascon S, Pelillo M, editors. Ancient coin classification using graph transduction games. 2018 Metrology for Archaeology and Cultural Heritage (MetroArchaeo); 2018: IEEE. [CrossRef]

[10] Kavelar A, Zambanini S, Kampel M, Vondrovec K, Siegl K. The ILAC-project: Supporting ancient coin classification by means of image analysis. Int Arch Photogram Remote Sens Spat Inform Sci 2013;40:373-378. [CrossRef]

[11] Anwar H, Zambanini S, Kampel M. Coarse-grained ancient coin classification using image-based reverse side motif recognition. Machine Vision Appl 2015;26:295-304. [CrossRef]

[12] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, et al, (editors). Going deeper with convolutions. Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2015. [CrossRef]

[13] Yu X, Zeng N, Liu S, Zhang Y-DJMV, Applications. Utilization of DenseNet201 for diagnosis of breast abnormality. Machine Vision Appl 2019;30:1135-1144. [CrossRef]

[14] Adam K, Mohamed II, Ibrahim YJIA. A selective mitigation technique of soft errors for dnn models used in healthcare applications: Densenet201 case study. IEEE Access 2021;9:65803-65823. [CrossRef]

[15] Balagourouchetty L, Pragatheeswaran JK, Pottakkat B, Ramkumar GJIjob, informatics h. GoogLeNet-based ensemble FCNet classifier for focal liver lesion diagnosis. IEEE J Biomed Health Inform 2019;24:1686-1694. [CrossRef]

[16] Yang N, Zhang Z, Yang J, Hong Z, Shi JJNRR. A convolutional neural network of GoogLeNet applied in mineral prospectivity prediction based on multi-source geoinformation. Nat Resour Res 2021;30:3905-3923. [CrossRef]

[17] Al Husaini MAS, Habaebi MH, Gunawan TS, Islam MR, Elsheikh EA, Suliman FJNC, et al. Thermal-based early breast cancer detection using inception V3, inception V4 and modified inception MV4. Neural Comput Appl 2022;34:333-348. [CrossRef]

[18] Mujahid M, Rustam F, Álvarez R, Luis Vidal Mazón J, Díez IdlT, Ashraf IJD. Pneumonia classification from X-ray ımages with ınception-V3 and convolutional neural network. Diagnostics (Basel) 2022;12:1280. [CrossRef]

[19] Cao J, Yan M, Jia Y, Tian X, Zhang Z. Application of a modified Inception-v3 model in the dynasty-based classification of ancient murals. EURASIP J Adv Signal Process 2021;2021:1-25. [CrossRef]

[20] Srinivasu PN, SivaSai JG, Ijaz MF, Bhoi AK, Kim W, Kang JJ. Classification of Skin Disease Using Deep Learning Neural Networks with MobileNet V2 and LSTM. Sensors (Basel) 2021;21:2852. [CrossRef]

[21] Michele A, Colin V, Santika DD. Mobilenet convolutional neural networks and support vector machines for palmprint recognition. Proced Comput Sci 2019;157:110-117. [CrossRef]

[22] Shen Y, Sun H, Xu X, Zhou J, (editors). Detection and positioning of surface defects on galvanized sheet based on improved MobileNet v2. 2019 Chinese Control Conference (CCC); 2019: IEEE. [CrossRef]

[23] Chen B, Liu X, Zheng Y, Zhao G, Shi Y, Technology SfV. A robust GAN-generated face detection method based on dual-color spaces and an improved Xception. IEEE 2021;32:3527-3538. [CrossRef]

[24] Kassani SH, Kassani PH, Khazaeinezhad R, Wesolowski MJ, Schneider KA, Deters R, (editors). Diabetic retinopathy classification using a modified xception architecture. 2019 IEEE international symposium on signal processing and information technology (ISSPIT); 2019: IEEE. [CrossRef]

[25] Pekel Özmen E, Özcan T. Diagnosis of diabetes mellitus using artificial neural network and classification and regression tree optimized with genetic algorithm. J Forecast 2020;39:661-670. [CrossRef]