Manuscript. DOI: Not available at this time.

Insect identification by combining different neural networks

Loris Nanni[1,*], Nicola Maritan[1], Daniel Fusaro[1], Sheryl Brahnam[2], Francesco Boscolo Meneguolo[1], Maria Sgaravatto[1]

[1] Department of Information Engineering, University of Padova, 35122 Padova, Italy; loris.nanni@unipd.it; nicola.maritan@studenti.unipd.it; fusarodani@dei.unipd.it; francesco.boscolomeneguolo@studenti.unipd.it; maria.sgaravatto@studenti.unipd.it

[2] Department of Information Technology and Cybersecurity, Missouri State University, 901 S. National Street, Springfield, MO 65804, USA sbrahnam@missouristate.edu

*Correspondence: loris.nanni@unipd.it

Abstract:

Background:

Traditional insect species classification relies on taxonomic experts examining unique physical characteristics of specimens, a time-consuming and error-prone process. Machine learning (ML) offers a promising alternative by identifying subtle morphological and genetic differences computationally. However, most existing approaches classify undescribed species as outliers, which limits their utility for biodiversity monitoring.

Objective:

This study aims to develop an ML method capable of simultaneously classifying described species and grouping undescribed species by genus, thereby advancing the field of automated insect classification.

Method:

We propose a novel ensemble approach combining neural networks (convolutional and attention-based) and Support Vector Machines (SVM), with both DNA barcoding and insect images as input data. To optimize the neural networks for diverse data types, we transform one-dimensional feature vectors into matrices using wavelet transforms. Additionally, a transformer-based architecture integrates DNA barcoding and image features for enhanced classification accuracy.

Experimental Results:

Our method was evaluated on a comprehensive dataset containing paired insect images and DNA barcodes for 1,040 species across four insect orders. The results demonstrate superior performance compared to existing methods in classifying described species and grouping undescribed ones by genus.

Conclusion:

The proposed approach represents a significant advancement in automated insect classification, addressing both described and undescribed species. This method has the potential to revolutionize global biodiversity monitoring. The MATLAB/PyTorch source code and dataset used are available at https://github.com/LorisNanni/Insect-identification.

Keywords: ensemble; convolutional neural networks; support vector machine; attention network; insect classification; DNA barcode.

1. Introduction

Analyzing biodiversity among insects is a challenging task. Entomologists must first conduct extensive fieldwork across diverse and often remote habitats to collect insect specimens. Once gathered, these specimens undergo identification through detailed morphological studies, genetic analysis, and taxonomic comparisons. This labor-intensive process expands our understanding of insect diversity, ecological roles, evolutionary relationships, and potential impacts on human activities.

Despite the estimated 5.5 million insect species, only about 20% have been cataloged (Stork, 2018). Complicating matters further, many species are vanishing before being formally identified (Costello, May, & Stork, 2013), making biodiversity assessments increasingly difficult. Taxonomists use morphological keys (Buck et al., 2009) to classify insects based on physical traits, but these keys become less effective when dealing with undescribed species that lack clear distinguishing features. DNA barcoding (Hebert, Cywinska, & Ball, 2003),

offers a supplementary approach by identifying species through genetic variation, especially when traditional phenotypic traits fall short (Burns, Janzen, Hajibabaei, Hallwachs, & Hebert, 2008).

Nonetheless, the identification process remains a bottleneck. Although the DNA Barcode Database (BOLD) (Ratnasingham & Hebert, 2007) stores vast data, only a fraction corresponds to identified species. This discrepancy underscores the slow pace of identification, exacerbated by a decline in taxonomists and traditional taxonomy (Orr, Ascher, Bai, Chesters, & Zhu, 2020). There is a pressing need for innovative and scalable methods to accelerate species discovery and identification if these obstacles are to be overcome.

Automated approaches for data analysis and integration, such as methods capable of extracting features directly from data without relying on geometric morphometric landmarks, remain relatively underexplored in the context of integrative taxonomy (Solís-Lemus, Knowles, & Ané, 2015). Frequently, researchers analyze different types of evidence independently and then synthesize them qualitatively. Over the past two decades, numerous studies have applied machine learning (ML) techniques to species delimitation or identification using images or genetic data (Ärje et al., 2020). However, these methods have not yet fully exploited multisource data in an automated manner. Advances in artificial intelligence offer promising opportunities to integrate multidimensional data objectively. For instance, artificial neural networks can automatically extract quantitative features to develop species profiles that capture unique character combinations along with their intraspecific variation (Valan, Makonyi, Maki, Vondráček, & Ronquist, 2019)

Building on recent advancements in deep learning, B. Yang et al. (2021) introduce a convolutional neural network (CNN) approach, the Morphology-Molecule Network (MMNet), designed to integrate morphological and molecular data for species identification across the tree of life.

ML methods offer promising solutions since they utilize intricate data patterns to classify and detect outliers. Traditional ML has demonstrated potential in identifying subtle morphological features in images (Haarika, Babu, & Nair, 2023), even for undescribed species. Although not yet as precise as DNA-based techniques, recent studies suggest ML is approaching expert-level accuracy in entomology (Milošević et al., 2020; Raitoharju & Meissner, 2019; Valan et al., 2019). Yet these traditional models face limitations, particularly due to incomplete training datasets, especially for rare or undescribed species and morphological variations across insect life stages (Badirli et al., 2023).

Deep learning, a subset of ML, has been applied across various entomological domains, from pest detection to the study of plant-insect interactions (Doan, 2023; Hedrick et al., 2020; B. Yang et al., 2022); however, these methods often focus on specific insect groups, limiting their scope, (MacLeod, Canty, & Polaszek, 2022). Crucially, ML insect identification methods must accommodate both described and undescribed species. Most existing approaches assume a complete representation of species in training data. Furthermore, these methods struggle with large class numbers and outlier differentiation within the diverse Insecta class (Badirli et al., 2023). Overcoming these limitations is vital for advancing entomological research and biodiversity assessment.

In this study, we tackle these challenges to ML in insect classification, especially the handling of incomplete representations, by proposing an ensemble model designed to identify both known and unknown species. Our approach integrates traditional Support Vector Machines (SVMs) with deep learning by transforming traditional feature patterns into 2D representations through vector-to-matrix reshaping into a three-channel input and concatenating DNA barcoding and image features to train a transformer-based architecture.

Finally, the rationale for combining SVMs and neural networks in an ensemble is that it benefits from the unique strengths of both algorithms, making the system useful for tasks where diversity in decision boundaries and robustness to overfitting are key. Neural Networks and SVMs learn in fundamentally different ways, so their errors often occur on different samples. While neural networks try to minimize error through backpropagation across multiple layers, SVMs maximize the margin between classes. This difference can lead to models that make complementary predictions. By combining them, the ensemble can achieve better performance as it benefits from the diverse decision boundaries learned by each model.

Our evaluations demonstrate that our approach surpasses the performance of finely-tuned SVMs, which remain prevalent in studies where patterns are described as 1D feature vectors, as evident in the current state-of-the-art performances reported in the insect dataset used in this work.

The main contributions of this paper include the following:

- Developing an ensemble classifier that surpasses traditional SVM performance and previous SOTA;
- Tested the proposed ensemble on more than one dataset;

- Proposing a novel method for representing feature vectors as images, achieved through continuous 1D wavelet transforms with varying mother wavelets for each image channel. An ensemble of networks is created by employing different sets of randomly selected mother wavelets;
- Providing all resources and source code to researchers in open source.

The paper is organized as follows: Section 2 describes the dataset used in this paper, and Section 3 reviews related work in vector-to-matrix reshaping methods and outlines the proposed approach. Section 4 details the experiments, followed by a discussion of issues in Section 5. The paper concludes with a summary and some future directions of research.

2. Material

In this section, we outline the DNA and image data employed in our study. We then describe the utilization of deep learning models for extracting feature vectors rich in information from insect images and DNA barcodes. Finally, we explain how the feature data is divided for training, validation, and testing purposes, as well as for simulating undescribed species.

Our study utilizes paired insect images and DNA sequence data acquired from the Barcode of Life Data (BOLD) system (Ratnasingham & Hebert, 2007) spanning four major Insecta orders: Diptera, Coleoptera, Lepidoptera, and Hymenoptera. Table 1 offers a breakdown of the dataset by order. Each insect pattern contains a 658 bp DNA barcode sequence (cytochrome oxidase subunit I-COI), an image, and supplementary details such as country of origin, life stage, order, family, subfamily, genus, and species names.

Order	**#Genera**	**#Species**	**#Samples**
Diptera (true flies)	63	108	2270
Coleoptera (beetles)	164	329	4764
Hymenoptera (sawflies, wasps, bees and ants)	59	189	3173
Lepidoptera (butterflies and moths)	82	414	22,641
Totals	368	1040	32,848

Table 1. Breakdown of the BOLD dataset by order.

The raw images are in full RGB color and typically have a width of 640 pixels and a height of 300–1000 pixels. Only images with corresponding DNA barcodes were included in this study, and each image underwent manual scrutiny to eliminate low-quality, duplicate, and incomplete insect bodies, immature pigmentation, and missing images (e.g., with only a label provided). Only species with a minimum of ten images within a single barcode index number (BIN) were considered.

BOLD sets itself apart from other genetic databases by accepting data for unidentified or unknown organisms, see Agarwala et al. (2017). BOLD's DNA-based grouping algorithms initially assign a BIN to unidentified samples, which are closely aligned (though not perfectly so) with species groupings. Subsequently, the BOLD database translates the sample's DNA sequence into its protein sequence and searches its repository for a species or genus match. If the sequence contains less than 1% divergence from a reference sequence, the sample is assigned to a species; if the divergence ranges between 1% and 3%, it is assigned to a genus; otherwise, the sample remains unidentified. As of May 25, 2024, the BOLD Insecta database contained 7,924,425 records with DNA sequences, of which only 3,019,210 had species names, indicating that most records remain unidentified. While crucial for new species discovery, the BOLD database has a significant limitation: it does not facilitate the identification of these new species beyond the aforementioned measures.

For a fair comparison with SOTA, we use the same features to represent the patterns proposed in Badirli et al. (2023), where deep learning models extract significant features from raw insect images and DNA barcode data. Specifically, we employed a pretrained ResNet-101 model to derive 2048-dimensional image feature vectors from the insect images, following established guidelines. The ResNet suite of models is advantageous as they have been pretrained on a vast dataset of over one million ImageNet images representing 1000 different classes, most of which are non-insect categories. This extensive pretraining enables ResNet models to recognize general image characteristics akin to human perception, such as edges, corners, blobs, and colors.

The raw insect images were preprocessed using standard transformations to ensure compatibility with ImageNet images before feeding them into the pretraned ResNet-101 model: each image was resized to 256 × 256 × 3, center-cropped to 224 × 224 × 3, and normalized using the ImageNet image pixel means [0.485, 0.456, 0.406] and standard deviations [0.229, 0.224, 0.225]. We chose not to fine-tune the ResNet-101 model on our

dataset after determining that fine-tuning had minimal impact on model accuracy (see Section 5 for further discussion of this point).

For extracting the DNA barcode sequence feature vectors (500-dimensional), a convolutional neural network (CNN) architecture tailored for this problem was applied by Badirli et al. (2023). The DNA barcodes were transformed into 658 × 5 arrays, indicating the presence (1) or absence (0) of adenine, guanine, cytosine, thymine, and other tokens for each position in the 658-length sequence. The CNN architecture comprised three blocks of convolutional layers, followed by batch normalization and 2D max-pooling. The output of the third convolutional layer was flattened and batch normalized before being fed into a fully connected layer with 500 units, whose output represented the learned features. Finally, a softmax layer concluded the CNN architecture.

To demonstrate the effectiveness of our model in identifying previously unidentified species, it underwent validation and testing using datasets containing samples of species not encountered during training. As our collected BOLD data lacked true samples of unidentified species, simulated data was generated. For genera with three or more species, one-third were randomly designated as undescribed, while the remainder were marked as described. Only the test set included undescribed species to evaluate the model's capability to identify them. For validation purposes, the unidentified species within the training set were segregated similarly to the described species. When insect species were represented by multiple images capturing different perspectives (e.g., ventral and dorsal views), all images were allocated to the training set. Consequently, 27 described species had no representatives in the test dataset. The test dataset contained 4965 samples from 770 described species and 8463 samples from 243 unidentified species (refer to Table 2 for details).

	Train	**Test (described)**	**Test (undescribed)**
# of samples	19,420	4965	8463
# of species	797	770	243

Table 2. Dataset splitting.

We wish to stress that the dataset used in this study is derived from previous literature, where specific sampling and stratification details (such as stratification by species or genus) were not explicitly provided.

Given the constraints of the dataset documentation, we worked within the available framework, applying the dataset as proposed by its original authors to maintain comparability with related studies.

3. Method

3.1 Related works on 2D matrix descriptors

Numerous advancements and adaptations of 1D transformations have been developed to process matrix data directly. These include 2D Principal Component Analysis (2DPCA) (J. Yang, Zhang, Frangi, & Yang, 2004) and 2D Linear Discriminant Analysis (2DLDA) (Li, Janardan, & Li, 2002), which effectively address the issue of singular scatter matrices. However, it has been noted that while LDA preserves covariance information among various local geometric structures, this valuable data is lost in 2DLDA.

Additional notable contributions include Sparse Two-Dimensional Discriminant Locality-Preserving Projection (S2DDLPP) (Zheng, Lai, & Li, 2008), 2D Outliers-Robust PCA (ORPCA) (Zhi & Ruan, 2008), Individual Local Mean-Based 2DPCA (ILM-2DPCA) (Razzak, Abu-Saris, Blumenstein, & Xu, 2020), and Regional Covariance Matrix Based on 2DPCA (RCM-2DPCA) (Hancherngchai, Titijaroonroj, & Rungrattanaubol, 2019), which tackles the problem of ineffective eigenvector generation by 2DPCA (Titijaroonroj, Hancherngchai, & Rungrattanaubol, 2020).

Recently, a novel approach for histopathological image classification using 2D vector quantification encoding with a bag of features was introduced by Pal and Saraswat (2020), along with a new 2D quaternion PCA known as BiG2DQPCA, explicitly designed for color image processing (Zhao, Jia, Gong, & Zhang, 2023).

Beyond these adaptations, native 2D matrix descriptors have also been developed, such as Gabor filters (Eustice, Pizarro, Singh, & Howland, 2002) and Local Binary Patterns (LBPs) and their derivatives (Brahnam, Jain, Lumini, & Nanni, 2014), which are considered powerful when applied to images. The performance advantage of these methods led researchers to explore transforming 1D data into 2D matrices so that such features as LBPs and Gabor filters could be extracted from them (Kim & Choi, 2007; Liu & Chen, 2006; Nanni, Brahnam, & Lumini, 2011; Z. Wang & Chen, 2008). The exploration of 1D reshaping methods in (Zhe Wang, Chen, Liu, & Zhang, 2008; Z. Wang & Chen, 2008) is particularly pertinent to this study since these methods were used to diversify classifiers using techniques such as AdaBoost.

Traditional classifiers like min-sum matrix products (MSPs) (Felzenszwalb & McAuley, 2011), nonnegative matrix factorization (NMF) (Seung & Lee, 2001), and the matrix-pattern-oriented modified Ho-Kashyap classifier (MatMHKS) (Chen, Wang, & Tian, 2007) have been specifically designed for 2D matrix data. Moreover, adaptations of classic learning models, such as the 2D nearest neighbor (Bentley, 1975), have been developed to handle 2D matrix data. Methods for feeding matrix data into CNNs by representing patterns as matrices have been documented in Shen et al. (2021) and Zhu et al. (2021). The method proposed by Zhu et al. (2021) (called IGTD in the experimental section of this paper) demonstrated superior results compared to models trained on conventional tabular data.

Recently, DeepInsight (Sharma, Vans, Shigemizu, Boroevich, & Tsunoda, 2019) (called DeepINS in the experimental section of this paper) was developed to convert non-image samples into well-organized image shapes, allowing any data to be classified by CNNs, including sets of vector features. Modifications to DeepInsight leveraging transformer models have been explored by Gokhale, Mohanty, and Ojha (2023). Other approaches based on DeepInsight have been proposed for various applications (Sharma, Lysenko, Boroevich, & Tsunoda, 2023), including predicting patient-specific anticancer drug responses and cancer classification. Finally, Tran, Tayara, and Chong (2024) present an interesting application of vector-to-matrix transformation for mutagenicity assessment, which is vital for determining the safety of chemicals and pharmaceuticals. Computational models play a crucial role in predicting toxicity efficiently. Tran et al. introduced AMPred-CNN, a mutagenicity prediction model that uses Convolutional Neural Networks to analyze molecular structures as images, taking advantage of CNNs' powerful feature extraction capabilities.

A couple of detailed surveys along this line of research are available, where experimental investigations into the performance of various CNN models and transformers were conducted (Medeiros Neto, Rogerio da Silva Neto, & Endo, 2023; Nanni, Brahnam, Loreggia, & Barcellona, 2023).

3.2 Proposed approach

In this work, we combine SVM and neural networks. A set of neural networks is created by varying the input to the networks using different approaches for transforming the feature vector describing each

pattern into a 3-channel input (see Figure 1). In this way, we can use network topologies created for computer vision approaches and feed them with images and DNA barcoding features.

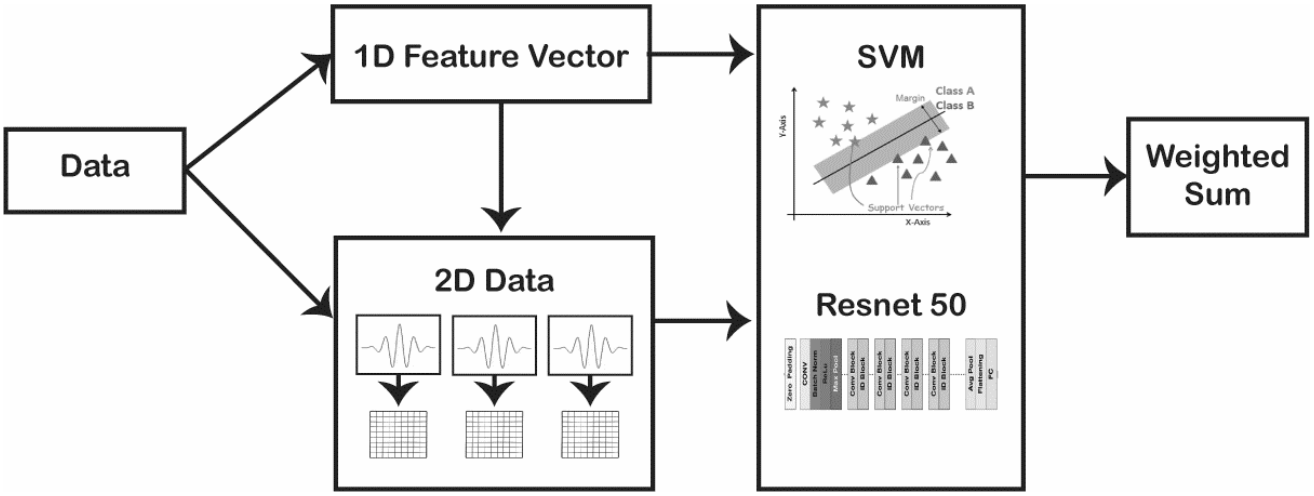

Figure 1. A schematic of the proposed method is shown. Ensembles are created by training SVMs on feature vectors and by converting 1D vectors into three channels of 2D data using a set of three wavelet transforms, which are then processed by a neural network (ResNet50 in the figure); the outputs are subsequently fused.

We employ ResNet50 for classification (He, Zhang, Ren, & Sun, 2016) as it performs similarly to ResNet101 but needs less computational time. It should be noted that for feature extraction, we used ResNet101 for a fair comparison with the current SOTA since it applies ResNet101 for feature extraction.

We also train from scratch a small neural network (called Attention in the experimental section) that incorporates convolutional layers and self-attention. Self-attention (Vaswani et al., 2017) is a method that enables the network to dynamically assess the significance of different features. This mechanism, commonly used in tasks that require understanding contextual relationships within data, is typically represented as:

$$Attention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = softmax\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_x}}\right)\mathbf{V},$$

where **Q**, **K**, **V** are queries, keys, and values, respectively, derived from the input data, and d_x is the dimension of the keys and queries, defined in the following way:

$$\mathbf{Q} = \mathbf{x} \cdot \mathbf{W_q}$$

$$\mathbf{K} = \mathbf{x} \cdot \mathbf{W_k}$$

$$\mathbf{V} = \mathbf{x} \cdot \mathbf{W_v},$$

where $\mathbf{x}$ is the input of the layer, and the matrices $\mathbf{W_q}$, $\mathbf{W_k}$, and $\mathbf{W_v}$ are learned during training.

The above formula aids the model in directing its Attention toward pertinent patches of the input data, thereby bolstering its predictive and analytical process. Moreover, the multi-head attention technique is employed, entailing the application of Attention multiple times. The operation of multi-head self-attention is expressed as:

$$\text{MultiHead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{Concat}(head_1, head_2, \ldots, head_h)\boldsymbol{W}^O,$$

where h is the number of heads, $\boldsymbol{W}^O$ is a learnable projection matrix, and $head_i = attention(\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i)$, the MultiHead learnable matrices. $Concat()$ is the concatenation of the output of the h heads.

Here is a detailed breakdown of the model architecture:

- Input: The network accepts input images with size 250×250×3.
- Convolutional Layer: Filters: 32; Kernel Size: 3×3; Padding: applied to ensure the output size matches the input size.
- Batch Normalization: This layer normalizes the activations from the previous convolutional layer to stabilize and accelerate training.
- A rectified linear unit (ReLU) Activation: this function is applied to introduce non-linearity to the model.
- Max Pooling: Pool Size: 2×2; Stride: 2; this layer reduces the spatial dimensions of the feature maps.
- Convolutional Layer: Filters: 64; Kernel Size: 3×3; Padding: applied to maintain the output size equivalent to the input size.
- Batch Normalization: Normalizes the activations from the second convolutional layer.
- ReLU Activation: Applies the ReLU activation function again.
- Max Pooling: Pool Size: 2×2; Stride: 2; this layer further reduces the spatial dimensions of the feature maps.

- Flatten: Converts the 2D feature maps into a 1D vector.
- Self-Attention Layer: Number of Heads: 8; Key and Query Channels: 64; This layer enables the network to focus on different parts of the input for a more comprehensive understanding.
- Fully Connected Layer: This layer has units equal to the number of output classes, facilitating the final classification.
- Softmax Activation: Applies the softmax function to the output of the fully connected layer, producing a probability distribution over the classes for classification.

The architecture of the Attention network is designed to utilize the strengths of CNNs for feature extraction and self-attention to capture long-range dependencies within the data, providing a robust model for classification tasks.

The network topologies outlined earlier are fortified through the approach detailed in Nanni, Lumini, Ghidoni, and Maguolo (2020), which entails forming an ensemble of networks. This ensemble is created by introducing randomness into the architecture of the networks, achieved by substituting various activation functions for each ReLU activation layer in the original network. Specifically, a replacement activation function is randomly chosen from a predetermined pool. See Nanni et al. (2020) for the list of activation functions and equations. This alteration yields a distinct network (due to the inherent randomness in the selection process) with each iteration. In addition, input is shuffled for each network before transforming from vector to the three-channel network input. The resultant ensemble of networks is then amalgamated using the mean rule, resulting in a resilient combined network.

Moreover, we propose a more computationally demanding ensemble that further boosts performance by adding to the above ensemble another transformer-based architecture (called TranConc in the experimental section), trained by concatenating DNA barcoding and image features. This topology is detailed below in section 3.2.2.

3.2.1 Vector-to-matrix using continuous wavelet

In mathematics, the continuous wavelet transform (CWT) is a formal method devoid of numerical computations that provides a comprehensive representation of a signal by continuously varying the translation and scale parameters of the wavelets.

In this context, the term "mother wavelet" refers to a continuous function existing in both the frequency and time domains. The primary function of the mother wavelet is to serve as a foundational function from which daughter wavelets are derived. These daughter wavelets are essentially translations and scaled versions of the mother wavelet. The scaling factor alters the signal by either compressing or dilating it. A lower scale factor compresses the signal, resulting in a graph with finer details, while a higher scale factor stretches the signal, producing a graph with fewer details. The outcome of the wavelet transform yields a wavelet coefficient associated with the scale a and position τ. Figure 2 illustrates an instance of the Morlet Wavelet varied in both scale and position.

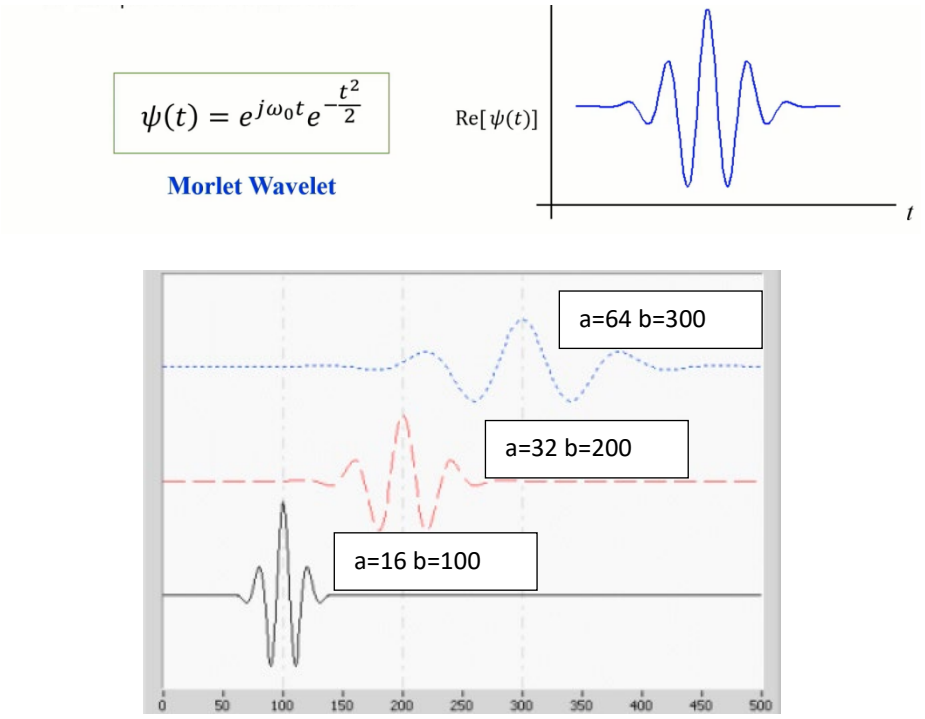

Figure 2. Example of a wavelet, from (Nanni and Brahnam (2020)).

We view the feature vector as a signal, subjecting it to a series of wavelet transforms. The resulting output forms a matrix where each column represents a specific scale and each row a particular time point. Within this matrix, the values correspond to the wavelet coefficients, which are then resized to match the

input size of the designated network. The Wavelet Transform can be likened to a convolution between the signal and scaled/shifted iterations of the mother wavelet (German-Sallo, 2011).

Applying the wavelet-based approach, as in Nanni and Brahnam (2020), we obtain a square matrix equal to the number of features, an unmanageable size for training a network. A possible solution is to resize such a matrix into a matrix with a size equal to the neural network input. This solution, however, significantly reduces the amount of information. An alternative approach is to divide the vector into a subsequence of 25 features, where, for each block, we apply a wavelet, obtaining a square matrix of size 25. These matrices are then concatenated ten per row to create a matrix of size 250×250, with missing elements replaced with 0. This matrix becomes the network input. This solution is performed three times to generate a final matrix with three channels. Our objective is to construct ensembles where, for each network, we randomly select three mother wavelets to generate three channels per signal (i.e., three feature vectors). These matrices are then concatenated to form 3-channel inputs for the network. It is important to note that we consistently employ the same three mother wavelets in a predefined order to generate the three channels for a given network. Before the wavelet transform, the patterns undergo linear normalization to fall within the [0, 255] range. Only the training data is utilized to determine the normalization parameters. In the experimental section, we call this approach BlockStocWave.

The mother wavelets composing the sets are the following:

- Meyer, support width: infinite; effective support: [-8 8].
- Haar, support width: 1, filter length: 2; scaling function phi = 1 on [0 1] and 0 otherwise; wavelet function psi = 1 on [0 0.5), = -1 on [0.5 1] and 0 otherwise.
- Daubechies order N=6, support width: 2N-1; filter length: 2N; regularity: about 0.2 N for large N; number of vanishing moments for psi: N.
- Symlet order N=6, support width: 2N-1; filter length: 2N; number of vanishing moments for psi: N.
- Coiflets order N=2, filter length: 6N; Number of vanishing moments for psi: 2N.
- Biorthogonal wavelet, order Nr=2, Nd=2; support width: 2Nr+1 for reconstruction, 2Nd+1 for decomposition; filter length: max(2×Nr, 2×Nd) + 2; regularity for psi reconstruction: Nr-1 and Nr-2 at the knots; number of vanishing moments for psi decomposition: Nr.

- Reverse biorthogonal wavelet, order Nr=2, Nd=2; support width: 2Nd+1 for reconstruction, 2Nr+1 for decomposition; filter length: max(2×Nd, 2×Nr) + 2; regularity for psi reconstruction: Nd-1 and Nd-2 at the knots; number of vanishing moments for psi decomposition: Nd;
- Discrete Meyer;
- Gaussian wavelet, order 4; support width: infinite; effective support: [-5 5];
- Mexican hat, support width: infinite; effective support: [-5 5];
- Morlet, support width: infinite; effective support: [-4 4];
- Fejer-Korovkin orthogonal, order=4;
- Beylkin orthogonal;
- Vaidyanathan orthogonal.

It is important to emphasize that our selection of wavelets is not driven by a desire to optimize performance and risk overfitting. Instead, we utilized those available in MATLAB, employing either the default hyperparameters or those specified in the literature. We use these default parameters because we did not create a standalone method that requires the tuning of hyperparameters for boosting performance (but such could be performed on any of the datasets). We use the same hyperparameters to avoid overfitting because we are producing a high-performing system that can be applied to any dataset in this domain. In this way, we can assume that our system will work well on a wide choice of datasets, a result that should benefit the community.

We have tried to replicate other published methods based on Deep learning for DNA barcoding but were never able to reproduce the results of the original papers, probably because parameter selections were made related to the particular dataset and there were no hyperdefault parameters. In our method, everything is the same for all datasets. Moreover, the code is downloadable, so it is easy to replicate our experiments. The uniform hyperparameters of the tested methods are reported in Table 3.

3.2.2 Transformer-based architecture

ResNet50/Attention - Hyperparameters	Values
Number epochs	10
Batch size	30
Stopping criteria	Early stopping based on validation loss monotony, validation loss not decrease for >= 2 epochs
Optimization approach	SGD
Learning rate	0.001
L2 Regularization	1.0000e-04
Gradient Threshold Method	L2 norm

TranConc - Hyperparameters	Values
Number epochs	3000
Batch size	512
Stopping criteria	Early stopping = validation loss, no decrease for >= 20 epochs
Optimization approach	SGD
Learning rate	0.01:0.055:0.1 (optimized using the grid search approach and optimizing for the validation loss).

Table 3. The hyperparameters of the tested networks.

The previous ensemble is, in turn, combined with a more computationally demanding deep learning-based architecture (called TranConc in the experimental section) that performs Cross-attention mixing DNA barcoding and image features. Cross-attention (Nanni et al., 2020) is based on self-attention but differs in that the source of keys and values comes from a different sequence than the queries. In self-attention, the key, queries, and values are from the same source. Cross-attention enables the model to combine two sources of information by computing attention weights that reflect the relevance of each element in one vector to the elements in another.

In TranConc (schematized in Figure 3), the insect and DNA barcoding feature vectors are first processed by two linear layers that map vectors to the same embedding dimension of 512. Then, the Cross-attention technique is applied twice in both modes: the first mode is to extract queries from the image vector and keys (values from the DNA vector), while the second mode is to extract queries from the DNA vector and keys (values from the image vector). In both modes, the Multi-head Attention mechanism is done in a 2-head setup followed by a normalization layer and a residual connection sum with the unnormalized context vector. A GELU activation function is performed only the first time the Cross-attention technique processes the vectors. The two images and

DNA feature vectors are then concatenated, forming a vector of size 1024. The species and genera logits are obtained from this vector using two linear layers.

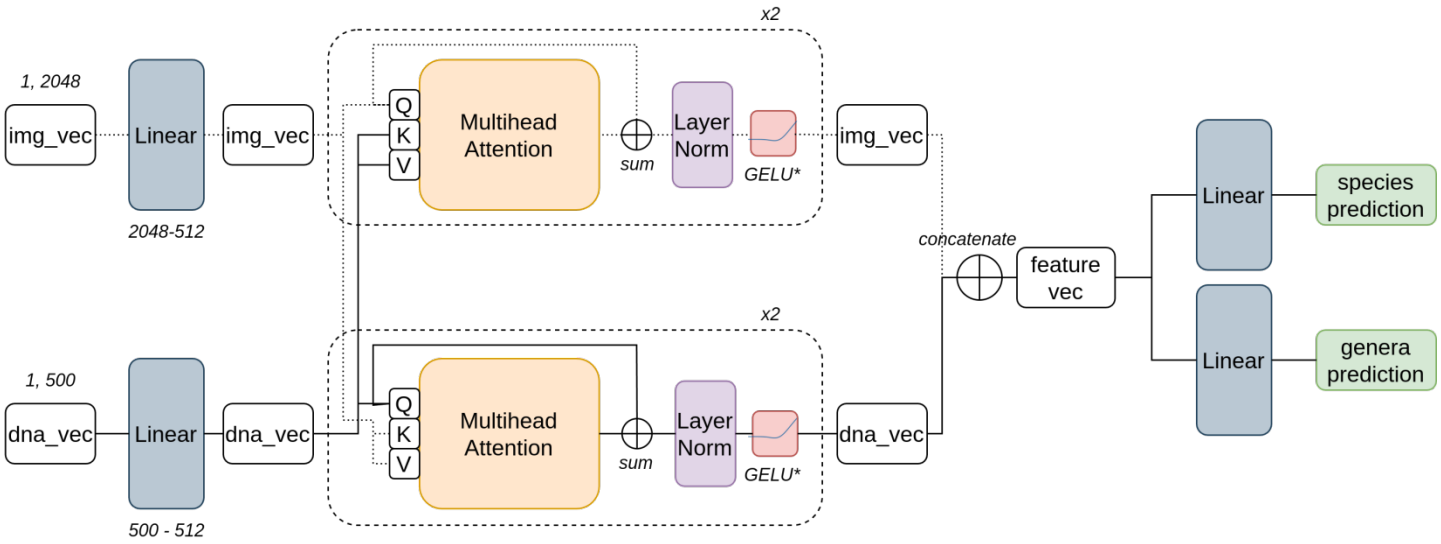

Figure 3. TranConc architecture. The species and genera logits are obtained using the Cross-attention technique, which processes the image and DNA barcoding feature vectors jointly. *Cross-attention is applied twice: the GELU activation function is applied only the first time.

4. Experimental section

The classification performance was assessed by the average described species accuracy and the average undescribed species genus accuracy:

$$\frac{1}{770}\sum_{i=1}^{770}\frac{y_i}{n_j}\text{, Species accuracy}$$

$$\frac{1}{134}\sum_{i=1}^{134}\frac{y_i}{n_j}\text{, Genus accuracy,}$$

where for class j, y_j is the number of correctly classified patterns of class j, and n_j is the number of total patterns for that class. 770 is the number of seen species, and the genera number of the undescribed species is 134.

In the first test, see Table 4, we compare the different approaches for transforming a vector into a matrix by considering the species classification (i.e., by considering only insects with known species). The average performance of 25 ResNet50 nets is reported. The method named StocWave applies the wavelet approach proposed by Nanni and Brahnam (2020). As can be observed, StocWave performs worse than BlockStocWave (the one proposed here); the other approaches perform similarly.

ResNet50	**Species**
DeepINS (Sharma et al., 2019)	98.56
StocWave	97.05
BlockStocWave	**98.65**
IGTD (Zhu et al. (2021))	98.62

Table 4. Average species accuracy of the 25 nets.

In the next test, see Table 5, we report the performance of the mean rule for combining 25 networks. In this test, we suppose that an oracle divides the insects between those with known species and those only with known genera, a division protocol used by Badirli et al. (2023).

Interestingly, Attention works better than ResNet50. BlockStocWave obtains the best performance in three out of four tests (i.e., two networks and two classification tasks), but it only slightly outperforms the other approaches. Using wavelets performs comparably with the current SOTA, and it deserves to be investigated further for boosting performance.

ResNet50	**Species**	**Genus**
DeepINS (Sharma et al., 2019)	99.06	**73.26**
BlockStocWave	**99.21**	72.53
IGTD (Zhu et al. (2021))	99.20	66.93
Attention	**Species**	**Genus**
DeepINS (Sharma et al., 2019)	99.30	78.71
BlockStocWave	**99.32**	**79.88**
IGTD (Zhu et al. (2021))	99.30	78.44

Table 5. The mean rule between the 25 networks.

In Table 6, we compare the following methods:

- The previous SOTA (Badirli et al., 2023);

- SVM, where the kernel and hyperparameters have been chosen with two-fold cross-validation using only the training data. The LibSVM toolbox for SVM. Hyperparameters are meticulously optimized using a grid search strategy. Both the kernel and its hyperparameters are selected through this process, with the polynomial kernel emerging as the best choice;
- SVM+BlockStocWave(Attention), mean rule between SVM and BlockStocWave coupled with the Attention network (mean rule among the 25 networks), i.e., first the 25 networks of BlockStocWave are combined by mean rule, then this score is combined by mean rule with the output of SVM;
- SVM+[BlockStocWave(Attention)+BlockStocWave(ResNet50)] as in the previous approach, but SVM is combined with BlockStocWave coupled with both networks (thus, the mean rule is applied across 50 networks before the fusion with SVM);
- SVM+[BlockStocWave(Attention)+BlockStocWave(ResNet50)]+TranConc, as the previous ensemble, adding the TranConc network.

As evident in Table 6, our proposed fusions obtain SOTA performance.

Method	**Species**	**Genus**
Previous SOTA	98.21	81.95
Previous SOTA (DNA barcoding only)	98.65	71.85
SVM	99.22	81.26
[BlockStocWave(Attention)+BlockStocWave(ResNet50)]	99.37	80.16
TranConc	99.39	80.09
SVM+[BlockStocWave(Attention)+BlockStocWave(ResNet50)]	99.37	82.51
SVM+[BlockStocWave(Attention)+BlockStocWave(ResNet50)]+TranConc	**99.47**	**83.22**

Table 6. Comparison with previous SOTA.

Since the source code in Badirli et al. (2023) is available, we tested it using our protocol for hyperparameters selection, i.e., two-fold using only the training set. Our stringent protocol decreases the performance of that approach (e.g., in genus classification, the performance is ~73%).

In the final test, we adopt a realist protocol where all the insects are classified at the species level (the species are the classes). Let us suppose:

- $\theta_1(\mathbf{x})$ is the highest score among the different species (i.e., classes) given a pattern $\mathbf{x}$;
- $\theta_2(\mathbf{x})$ is the second highest score of that pattern;
- $\theta(\mathbf{x}) = \theta_1(\mathbf{x}) - \theta_2(\mathbf{x})$.

Our rejection criterion is as follows:

- If $\theta(\mathbf{x})>\tau$, the insect is assigned to a species class; otherwise, it is assigned to a genus class (i.e., it is classified by the network trained using the genus as classes);
- If a pattern belongs to a known species but is classified at the genus level, it is considered a classification error (see the previous equation for calculating the accuracy); clearly, a pattern with unknown species is regarded as an error if classified at the species level.

The accuracy is calculated using the average accuracy among the classes: the species accuracy is calculated assuming only the patterns with $\theta(\boldsymbol{x}) > \tau$. In contrast, the genus accuracy is calculated considering only the patterns with $\theta(\boldsymbol{x}) \leq \tau$.

In Figure 4, we report the plot of the species accuracy (x-axis) vs genus accuracy (y-axis) obtained by varying the rejection threshold τ. The green line is obtained by SVM and the black line by our best ensemble (SVM+[BlockStocWave(Attention)+BlockStocWave(ResNet50)]+TranConc). To calculate the rejection threshold for our ensemble, we consider the TranConc logits and obtained a similar performance using SVM or CNN-based ensemble logits. This test clearly shows the usefulness of the proposed ensemble vs SVM.

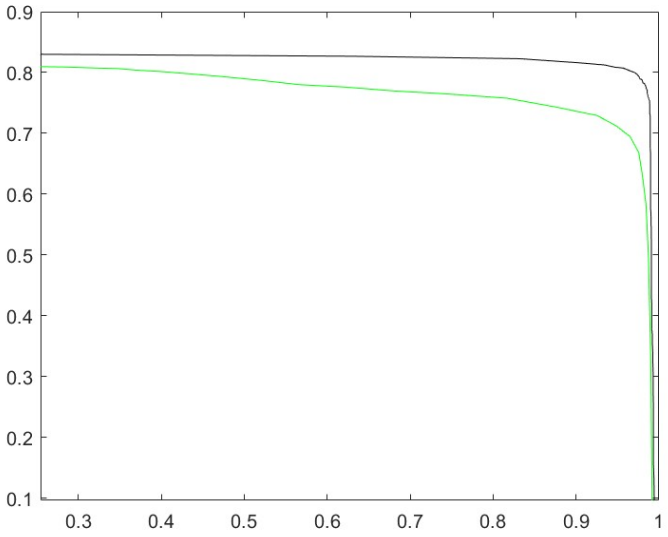

Figure 4. Curve species accuracy (x-axis) vs genus accuracy (y-axis) obtained by varying the rejection threshold (green is the performance of SVM; black is the proposed ensemble).

Because we use a transparent and fair testing protocol, we hope our work will become a baseline for future research in this area. An improvement over SVM is not remarkable using the protocol applied in the results reported in Tables 4-6. However, the advantage is clear, considering the realistic protocol for plotting Figure 4. We want to point out that we chose the kernel and relative hyperparameters for SVM, which is essentially the most commonly used method when feature vectors represent patterns. Thus, any improvement should prove helpful to the community.

It is important to note that the random choice of activation functions reduces overfitting, as detailed in (Nanni et al. (2020). Because of this, we have not made any selections for the training set. To validate our method further, we ran tests on images extracted from the BOLD Insect Classification dataset (De Gobbi, Lavezzi, & De Almeida Matos Junior), which contains 32,424 image samples of insect species from four Insecta orders: Diptera, Coleoptera, Lepidoptera, and Hymenoptera. The number of samples in the New dataset is described in Table 7. As shown in Table 8, the same superior results are obtained on the New dataset using the same method reported above.

	Training	Testing (described)	Testing (undescribed)
# of samples	19,994	4990	7440
# of species	835	797	215

Table 7. Description of the New dataset.

The major drawback of our approach is that it is much more computationally burdensome than previous SOTA or SVM classification methods. However, our approach on the New dataset performs better than prior work.

Method	Species	Genus
SOTA	96.35	80.15
SVM	98.21	82.01
Proposed Ensemble	**99.05**	**84.02**

Table 8. Performance in the New dataset.

Finally, to assess the impact of size on performance using our approach, we ran tests on only one pattern for each class (see Table 9). As would be expected, the performance of our approach declines significantly, but our ensemble still outperforms SVM on this test.

Method	Species	Genus
SVM	53.69	16.36
Proposed Ensemble	**59.75**	**22.28**

Table 9. Performance of using only one pattern for each class.

5. Discussion

One limitation of ensemble methods is their dependence on available datasets and the substantial computational resources they require. When high-end performance is needed at the expense of computational

efficiency, distillation techniques become imperative. However, the execution times of our approach are manageable for many applications, as shown in Table 10, where we report computation times for a batch size of 10,000 patterns, including time and RAM usage for a standalone network. The computation times are reported for a Titan V with 24 GB of RAM. For ensembles, multiply by the number of networks and account for the overhead. A key challenge may arise when loading the trained models into the GPU due to memory constraints. Since all models can run in parallel, using more GPUs will reduce computation time. The row labeled "BlockStocWave" refers to the method used to create matrices for the input into the neural networks. This process runs on a CPU, specifically an i9-10920X with 12 cores at 3.5 GHz, but it utilizes only a single core, allowing for 12 parallel extractions.

	Computation Time	***Model size***
ResNet50	11.24 sec	96 MB
Attention	18.90 sec	3.8 MB
BlockStocWave	0.56 sec/core	---

Table 10. Comparison of computation times.

The inclusion of SVM in our ensembles did not impact computation time; the difference was negligible with respect to the ResNet50/Attention computation time. SVM inference is performed using a CPU, while neural network inference is performed on a GPU. It should be noted, however, that both methods could run in parallel.

From the results reported in Table 10, we can draw the following conclusions:

- Even a large ensemble of 25 ResNet50 and 25 Attention models can classify 10,000 patterns in 12.5 minutes. On May 25, 2024, the BOLD Insecta database contained 7,924,425 records with DNA sequences, while on June 6, 2022, the BOLD Insecta database had 7,192,313 records with DNA sequences. Thus, approximately 750,000 new elements were added in this two-year period. Given the calculations in Table 9, our approach can process the 750,000 new patterns in approximately 15.625 hours.

- Using a 12-core CPU, we were able to extract BlockStocWave representations for approximately twenty patterns every second. Thus, it is possible to extract all the representations for 750,000 additional patterns in 10.45 hours. Note that we do not have to extract these representations for all patterns; after extracting them in the first batch, we can run inference with neural networks.

Clearly, our proposed system can run well if a GPU is available. However, it is not suited for edge computing. Today, however, satellite connections for uploading data are relatively easy to obtain. Many expeditions are armed with a Starlink connection to upload data to servers for analysis. The expense of GPUs is negligible compared to the costs of an expedition. Though our system is not suitable for edge computing, it works well on any machine with a decent GPU. Thus, computation concerns would be an issue only if there is no internet connection.

However, because the computational demand is only during training time, this ensemble would not work well for tasks where lots of classes are being added. In this case, our system would need to be coupled with a continuous learning algorithm.

Figure 5. Low-quality image examples in the New dataset.

Another concern is that the quality of input data (image clarity and the correctness of the DNA sequencing) affects performance (Karim & Abid, 2021). Recall that we eliminated low-quality images in the BOLD dataset used in the experimental section. Figure 5 provides examples of poor-quality images in the New dataset. As already noted in the discussion of Table 6, our system performs better than SVM on the New dataset despite some low-quality images.

Because of the challenges associated with sequencing technologies and the unavailability of high-quality genomic DNA, it is not always possible to obtain the full-length barcode sequence of an organism. To assess how our approach handles low-quality DNA sequencing, we ran the tests on our complete system proposed by Karim and Abid (2021), which evaluates accuracy using 300, 400, 500, and 600 bases. As shown in Table 11, our system is quite robust on a subset of the full DNA barcoding sequence and produces reliable results using only 300 bases.

Method	**Species**	**Genus**
Full-length	99.47	83.22
600 bases	99.18	83.48
500 bases	98.75	83.15
400 bases	98.56	82.92
300 bases	96.54	79.25

Table 11. Results of the test proposed in (Karim & Abid, 2021) to assess poor DNA sequencing using 300, 400, 500, and 600 bases.

Additionally, we ran a test comparing ResNet-101 with and without tuning as a feature extractor. As can be seen in Table 12, the performance is similar in both cases. Notice that ResNet-101 is used as a feature extractor first. These features are then fed into other networks. In our opinion, the fact that ResNet-101 is a feature extractor explains why the performance is so similar.

Method	**Species**	**Genus**
The proposed method using ResNet-101 without tuning it	99.15	90.52
The proposed method using ResNet-101 tuning it	99.21	90.37

Table 12. Comparison of the ResNet-101 feature extractor with and without tuning.

In Figure 6, we show the output of our wavelet-based approach for two images of two different classes. In order to make the figure clearer, we use only four submatrices instead of 100 to generate the images. As can be observed, the images are more similar between patterns of the same class than they are between patterns of different classes.

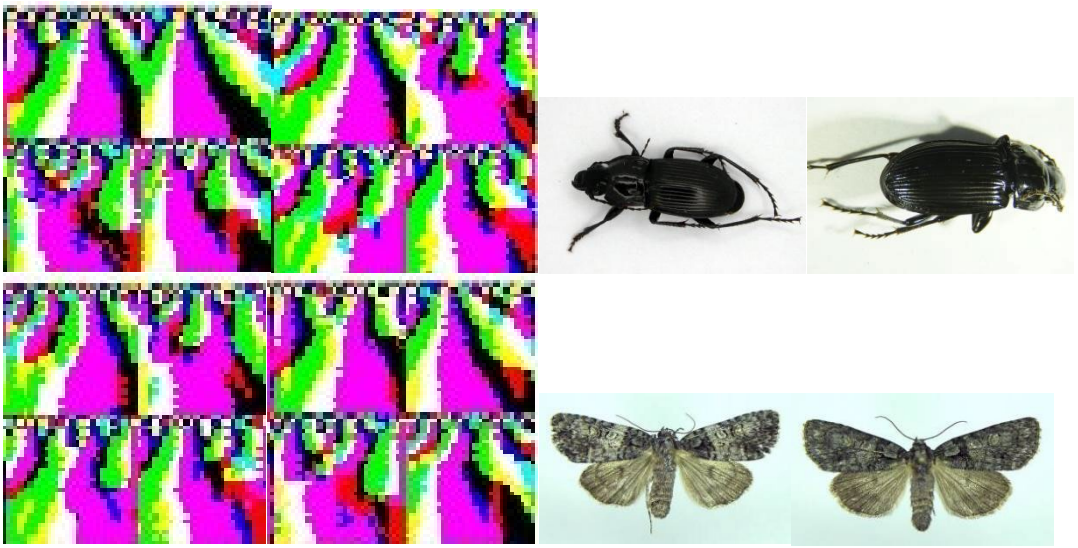

Figure 6. Two images obtained by our wavelet-based method (left) and the corresponding RGB images (right). The first row is related to two patterns of the species Abax parallelepipedus; the second row is related to two patterns of the species Acronicta increta.

Finally, a word about DNA species classification in general and the presentation of one additional test using our ensemble to demonstrate its robustness. Various classifiers have been applied to species classification using DNA barcodes, including SVM, naive Bayes (NB), k-nearest neighbor (KNN), multilayer perceptron (MLP), decision trees (DT), random forests (RF), and hierarchical supervised models (Meher, Sahu, Gahoi, Tomar, & Rao, 2019). RF-based models have effectively predicted fungal species by mapping DNA sequences to numeric features, achieving over 85% accuracy, a performance that improves with additional reference data (Meher et al., 2019). For plant classification, supervised learning algorithms applied to DNA barcodes (e.g., rbcL gene) achieve more than 97% accuracy, closely aligning with NCBI classifications (Rizaa et al., 2023). BayesANT, a Bayesian nonparametric classifier, accurately identifies taxa, including unknown species, using Finnish arthropod data (Zito, Rigon, & Dunson, 2023). Molecular inventories, such as those conducted on Lepidoptera in the Cottian Alps, highlight DNA barcoding's role in discovering cryptic species and addressing taxonomic discrepancies (Huemer & Wieser, 2023). In addition, mini-barcodes (100-300 bp), though shorter than standard barcodes (650 bp), have proven effective for species identification using supervised learning (Karim & Abid, 2021). A notable

deep learning approach, the ESK model, combines Elastic Net-Stacked Autoencoder (EN-SAE) with Kernel Density Estimation (KDE), demonstrating high accuracy in classifying fish families (Jin, Yu, Yuan, & Du, 2021). Comparative studies of database-based and ML methods suggest combining approaches enhances classification accuracy (Tian, Zhang, Zhai, Wang, & Zou, 2024).

Please note that comparing various DNA barcoding classification methods based on deep learning is not the core of this paper; we used the same extraction method as the baseline paper to have a fair comparison with that method. In this way, we can better evaluate the goodness of the classification step. The core of the paper is to motivate the classification part by showing that it does better than the baseline using the same features.

The results of our experiments justify our argument for combining SVM with a deep learner. We chose SVM because it is the most widely used learner in the literature and has a proven track record. Note how, in Table 4, the performance of the single network is lower than that of SVM (Table 6). The next logical step was to test an ensemble of networks, as presented in Table 5, where the performance of the ensemble of networks compared to the standalone network is shown to be higher. At this point, we have different architectures and different methods representing a pattern in a manner suitable for training a neural network. Therefore, we tried combining networks with various architectures and other techniques to present the pattern, as shown in Table 6. This step allowed for a further increase in performance. The last step was to combine the deeplearning ensemble proposed in this paper with SVM. This fusion is our suggested method; it is an improvement over SVM in all tested datasets and achieves results superior to the baselines presented in previous papers.

To demonstrate the robustness of our method, we tested the proposed ensembles, using the same hyperparameters as before but including another dataset in our tests. For this demonstration, we ran our approach on the beetle dataset proposed in B. Yang et al. (2022). The beetle dataset dataset contains 615 mitochondrial COI fragments from 123 beetle species belonging to three families (Coccinellidae, Cantharidae, Anthribidae) and is available at https://datadryad.org/stash/dataset/doi:10.5061/dryad.zgmsbccc3 (accessed 12/06/2024). We obtained the same conclusions as those on the previous datasets. As can be seen in Table 13, our ensemble improves SVM and SOTA. We wish to stress that our ensemble does not have hyperparameters to be set on each dataset, so these results are obtained using the same ensemble across all datasets.

Method	Accuracy
(B. Yang et al., 2022)	98.1
SVM	97.9
Proposed Ensemble	**98.4**

Table 13. Accuracy in the Beetle dataset.

Conclusion

In this study, we explored the training of neural networks using matrices derived from reshaping the original feature vectors. Additionally, we presented findings on the performance achieved by fusing different neural network topologies with SVM. Our research contributes to this field by delving into various network topologies and assessing the effectiveness of combining diverse classifiers to construct heterogeneous ensembles.

The key innovations we introduced in this work are the following: 1) we introduced a novel approach for constructing CNN ensembles by employing different mother wavelets for vector-to-matrix transformation; 2) we demonstrated the superior performance of the proposed approach by comparing ensembles with Support Vector Machines (SVM); and 3) we proposed an ensemble based on three components (SVM, CNN and Transformers) that outperforms both previous SOTA and SVM.

The disadvantage of this approach is that ensembles require more computational power than standalone methods. However, this power demand, as we have shown, is easy to satisfy given the computational power of current GPUs.

In forthcoming research, we intend to augment our analysis by incorporating new datasets to enhance the generalization of our method. We also plan to evaluate alternative techniques proposed in the literature for generating suitable matrices for CNN training. Future directions also include new methods for describing the DNA barcode and including the image of each insect, as well as new approaches for rejecting patterns that do not have a species label. Finally, we will work on developing distillation and continuous learning approaches for our system to facilitate edge computing and the addition of many new classes. Continual learning is necessary to avoid the catastrophic forgetting problem.

ACKNOWLEDGMENTS

We would like to acknowledge the support that NVIDIA provided us through the GPU Grant Program. We used donated GPUs to train the neural networks used in this work.

References

Agarwala, R., Barrett, T., Beck, J., Benson, D. A., Bollin, C., Bolton, E., . . . Canese, K. (2017). Database resources of the national center for biotechnology information. *Nucleic Acids Research, 46*(D1), D8-D13.

Ärje, J., Melvad, C., Jeppesen, M. R., Madsen, S. A., Raitoharju, J., Rasmussen, M. S., . . . Meissner, K. (2020). Automatic image‐based identification and biomass estimation of invertebrates. *Methods in Ecology and Evolution, 11*(8), 922-931.

Badirli, S., Picard, C. J., Mohler, G., Richert, F., Akata, Z., & Dundar, M. (2023). Classifying the unknown: Insect identification with deep hierarchical Bayesian learning. *Methods in Ecology and Evolution, 14*(6), 1515-1530. doi:https://doi.org/10.1111/2041-210X.14104

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM, 18*(9), 509-517.

Brahnam, S., Jain, L. C., Lumini, A., & Nanni, L. (2014). Introduction to Local Binary Patterns−New variants and new applications. In S. Brahnam, L. C. Jain, A. Lumini, & L. Nanni (Eds.), *Advanced computational Intelligence Paradigms in HealthcareL Local Binary Patterns−New Variants and New Applications*. Berlin: Springer-Verlag.

Buck, M., Woodley, N. E., Borkent, A., Wood, D. M., Pape, T., Vockeroth, J., . . . Marshall, S. (2009). Key to Diptera families-adults. *Manual of Central American Diptera, 1*, 95-156.

Burns, J. M., Janzen, D. H., Hajibabaei, M., Hallwachs, W., & Hebert, P. D. (2008). DNA barcodes and cryptic species of skipper butterflies in the genus Perichares in Area de Conservacion Guanacaste, Costa Rica. *Proceedings of the National Academy of Sciences, 105*(17), 6350-6355.

Chen, S., Wang, Z., & Tian, Y. (2007). Matrix-pattern-oriented ho-kashyap classifier with regularization learning. *Pattern Recognition, 40*(5), :1533–1543.

Costello, M. J., May, R. M., & Stork, N. E. (2013). Can we name Earth's species before they go extinct? *Science, 339*(6118), 413-416.

De Gobbi, M., Lavezzi, L., & De Almeida Matos Junior, R. *Insect Classification*. Retrieved from: https://zenodo.org/records/13858306 (accessed October 01, 2024)

Doan, T.-N. (2023). Large-scale insect pest image classification. *Journal of Advances in Information Technology, 14*(2), 328-341.

Eustice, R., Pizarro, O., Singh, H., & Howland, J. (2002). *UWIT: Underwater image toolbox for optical image processing and mosaicking in MATLAB*. Paper presented at the International Symposium on Underwater Technology, Tokyo, Japan.

Felzenszwalb, P., & McAuley, J. (2011). Fast inference with min-sum matrix product. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 33*(12), 2549-2554.

German-Sallo, Z. (2011). Nonlinear wavelet denoising of data signals. *UbiCC J, 6*, 895-900.

Gokhale, M., Mohanty, S. K., & Ojha, A. (2023). Genevit: gene vision transformer with improved deepinsight for cancer classification. *Computers in Biology and Medicine, 155*, 106643.

Haarika, R., Babu, T., & Nair, R. R. (2023). Insect classification framework based on a novel fusion of high-level and shallow features. *Procedia Computer Science, 218*, 338-347.

Hancherngchai, K., Titijaroonroj, T., & Rungrattanaubol, J. (2019). *An individual local mean-based 2DPCA for face recognition under illumination effects.* Paper presented at the 2019 16th International Joint Conference on Computer Science and Software Engineering (JCSSE).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). *Deep residual learning for image recognition*. Paper presented at the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV.

Hebert, P. D., Cywinska, A., & Ball, S. L. (2003). *deWaard JR (2003) Biological identifications through DNA barcodes.* Paper presented at the Proc Biol Sci.

Hedrick, B. P., Heberling, J. M., Meineke, E. K., Turner, K. G., Grassa, C. J., Park, D. S., . . . Davis, C. C. (2020). Digitization and the Future of Natural History Collections. *BioScience, 70*(3), 243-251. doi:10.1093/biosci/biz163

Huemer, P., & Wieser, C. (2023). DNA barcode library of megadiverse lepidoptera in an Alpine Nature Park (Italy) reveals unexpected species diversity. *Diversity, 15*(2), 214.

Jin, L., Yu, J., Yuan, X., & Du, X. (2021). Fish classification using DNA barcode sequences through deep learning method. *Symmetry, 13*(9), 1599.

Karim, M., & Abid, R. (2021, 13-15 Oct. 2021). *Efficacy and accuracy responses of DNA mini-barcodes in species identification under a supervised machine learning approach.* Paper presented at the 2021 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB).

Kim, C., & Choi, C.-H. (2007). A discriminant analysis using composite features for classification problems. *Pattern Recognition, 40*(11), 2958-2966.

Li, J., Janardan, R., & Li, Q. (2002). Two-dimensional linear discriminant analysis. *Advances in Neural Information Processing Systems 17*, 1569-1576.

Liu, J., & Chen, S. C. (2006). Non-iterative generalized low rank approximation of matrices. *Pattern Recognition Letters, 27*(9), 1002-1008.

MacLeod, N., Canty, R. J., & Polaszek, A. (2022). Morphology-based identification of Bemisia tabaci cryptic species puparia via embedded group-contrast convolution neural network analysis. *Systematic Biology, 71*(5), 1095-1109.

Medeiros Neto, L., Rogerio da Silva Neto, S., & Endo, P. T. (2023). A comparative analysis of converters of tabular data into image for the classification of Arboviruses using Convolutional Neural Networks. *PLoS ONE, 18*(12), e0295598.

Meher, P. K., Sahu, T. K., Gahoi, S., Tomar, R., & Rao, A. R. (2019). funbarRF: DNA barcode-based fungal species prediction using multiclass Random Forest supervised learning model. *BMC genetics, 20*(1), 1-13.

Milošević, D., Milosavljević, A., Predić, B., Medeiros, A. S., Savić-Zdravković, D., Piperac, M. S., . . . Leese, F. (2020). Application of deep learning in aquatic bioassessment: Towards automated identification of non-biting midges. *Science of the Total Environment, 711*, 135160.

Nanni, L., & Brahnam, S. (2020). Vector to Matrix Representation for Cnn Networks for Classifying Astronomical Data. *Available at SSRN 4827334*.

Nanni, L., Brahnam, S., Loreggia, A., & Barcellona, L. (2023). Heterogeneous Ensemble for Medical Data Classification. *Analytics, 2*(3), 676-693. Retrieved from https://www.mdpi.com/2813-2203/2/3/37

Nanni, L., Brahnam, S., & Lumini, A. (2011). Local ternary patterns from three orthogonal planes for human action classification. *Expert Systems with Applications, 38*(5), 5125-5128.

Nanni, L., Lumini, A., Ghidoni, S., & Maguolo, G. (2020). Stochastic Selection of Activation Layers for Convolutional Neural Networks. *Sensors (Basel, Switzerland), 20*(6). doi:10.3390/s20061626

Orr, M. C. C., Ascher, J. S., Bai, M., Chesters, D., & Zhu, C.-D. (2020). Three questions: How can taxonomists survive and thrive worldwide? *Megataxa, 1*(1), 19–27-19–27.

Pal, R., & Saraswat, M. (2020). A new weighted two-dimensional vector quantisation encoding method in bag-of-features for histopathological image classification. *International Journal of Intelligent Information and Database Systems, 13*(2-4), 150-171.

Raitoharju, J., & Meissner, K. (2019). *On confidences and their use in (semi-) automatic multi-image taxa identification.* Paper presented at the 2019 IEEE symposium series on computational intelligence (SSCI).

Ratnasingham, S., & Hebert, P. D. (2007). BOLD: The Barcode of Life Data System (http://www. barcodinglife. org). *Molecular ecology notes, 7*(3), 355-364.

Razzak, I., Abu-Saris, R. M., Blumenstein, M., & Xu, G. (2020). Integrating joint feature selection into subspace learning: A formulation of 2DPCA for outliers robust feature selection. *Neural networks : the official journal of the International Neural Network Society, 121*, 441-451.

Rizaa, L. S., Rahman, M. A. F., Prasetyo, Y., Zain, M. I., Siregar, H., Hidayat, T., . . . Rosyda, M. (2023). Comparison of Machine Learning Algorithms for Species Family Classification using DNA Barcode.

Seung, D., & Lee, L. (2001). Algorithms for non-negative matrix factorization. *Advances in neural information processing systems, 13*, 556-562.

Sharma, A., Lysenko, A., Boroevich, K. A., & Tsunoda, T. (2023). DeepInsight-3D architecture for anti-cancer drug response prediction with deep-learning on multi-omics. *Scientific Reports, 13*(1), 2483.

Sharma, A., Vans, E., Shigemizu, D., Boroevich, K. A., & Tsunoda, T. (2019). DeepInsight: A methodology to transform a non-image data to an image for convolution neural network architecture. *Scientific Reports, 9*(1), 11399.

Shen, W., Zeng, X., Zhu, F., Wang, Y. l., Qin, C., Tan, Y., . . . Chen, Y. Z. (2021). Out-of-the-box deep learning prediction of pharmaceutical properties by broadly learned knowledge-based molecular representations. *Nature Machine Intelligence*, 1-10.

Solís-Lemus, C., Knowles, L. L., & Ané, C. (2015). Bayesian species delimitation combining multiple genes and traits in a unified framework. *Evolution, 69*(2), 492-507.

Stork, N. E. (2018). How many species of insects and other terrestrial arthropods are there on Earth? *Annual review of entomology, 63*, 31-45.

Tian, Q., Zhang, P., Zhai, Y., Wang, Y., & Zou, Q. (2024). Application and Comparison of Machine Learning and Database-Based Methods in Taxonomic Classification of High-Throughput Sequencing Data. *Genome Biology and Evolution, 16*(5), evae102.

Titijaroonroj, T., Hancherngchai, K., & Rungrattanaubol, J. (2020). *Regional covariance matrix-based two-dimensional pca for face recognition.* Paper presented at the 2020 12th International Conference on Knowledge and Smart Technology (KST).

Tran, T. T. V., Tayara, H., & Chong, K. T. (2024). AMPred-CNN: Ames mutagenicity prediction model based on convolutional neural networks. *Computers in Biology and Medicine, 176*, 108560. doi:https://doi.org/10.1016/j.compbiomed.2024.108560

Valan, M., Makonyi, K., Maki, A., Vondráček, D., & Ronquist, F. (2019). Automated taxonomic identification of insects with expert-level accuracy using effective feature transfer from convolutional networks. *Systematic Biology, 68*(6), 876-895.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., . . . Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems, 30*.

Wang, Z., Chen, S., Liu, J., & Zhang, D. (2008). Pattern representation in feature extraction and classifier design: matrix versus vector. *IEEE Transactions on Neural Networks, 19*(5), 758-769.

Wang, Z., & Chen, S. C. (2008). Matrix-pattern-oriented least squares support vector classifier with AdaBoost. *Pattern Recognition Letters, 29*, 745-753.

Yang, B., Zhang, Z., Yang, C.-Q., Wang, Y., Orr, M. C., Wang, H., & Zhang, A.-B. (2021). Identification of Species by Combining Molecular and Morphological Data Using Convolutional Neural Networks. *Systematic Biology, 71*(3), 690-705. doi:10.1093/sysbio/syab076

Yang, B., Zhang, Z., Yang, C.-Q., Wang, Y., Orr, M. C., Wang, H., & Zhang, A.-B. (2022). Identification of species by combining molecular and morphological data using convolutional neural networks. *Systematic Biology, 71*(3), 690-705.

Yang, J., Zhang, D., Frangi, A. F., & Yang, J. U. (2004). Two-dimension pca: A new approach to appearance-based face representation and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 26*(1), 131-137.

Zhao, M.-X., Jia, Z.-G., Gong, D.-W., & Zhang, Y. (2023). Data-Driven Bilateral Generalized Two-Dimensional Quaternion Principal Component Analysis with Application to Color Face Recognition. *arXiv preprint arXiv:2306.07045*.

Zheng, W.-S., Lai, J., & Li, S. (2008). 1D-LDA vs. 2D-LDA: When is vector-based linear discriminant analysis better than matrix-based? *Pattern Recognit., 41*(7), 2156-2172.

Zhi, R., & Ruan, Q. (2008). Facial expression recognition based on two-dimensional discriminant locality preserving projections. *Neurocomputing, 71*(7-9), 1730-1734.

Zhu, Y., Brettin, T., Xia, F., Partin, A., Shukla, M., Yoo, H., . . . Stevens, R. L. (2021). Converting tabular data into images for deep learning with convolutional neural networks. *Scientific Reports, 11*(1), 11325.

Zito, A., Rigon, T., & Dunson, D. B. (2023). Inferring taxonomic placement from DNA barcoding aiding in discovery of new taxa. *Methods in Ecology and Evolution, 14*(2), 529-542.