Kissing Bugs Identification using Convolutional Neural Network

BASSAM A. ABDELGHANI, SHADI BANITAAN, MINA MALEKI, AND AMNA MAZEN.
Department of Electrical and Computer Engineering and Computer Science, University of Detroit Mercy, Detroit, Michigan
Corresponding author: Bassam A. Abdelghani (e-mail: abdelgba@udmercy.edu).

ABSTRACT
Chagas disease is one of the most important parasitic diseases transmitted to animals and people by insect vectors. According to the World Health Organization, around seven million people were infected with Trypanosoma cruzi (also known as kissing bug) that causes Chagas disease. As kissing bugs belong to different families with different danger levels, accurate classifications of kissing bugs species would help the public authorities create a controlled surveillance system. Clinical methods for detecting kissing bugs are expensive, time-consuming, and need a high level of expertise. To overcome these limitations, computational methods can be used. In this paper, a fully automated deep learning model using a convolutional neural network (CNN) with a fine-tuned transfer learning model is proposed to identify kissing versus non-kissing bugs and classify the type of kissing bug species. The accuracy of 99.45% for the classifications of kissing vs. non-kissing bugs and 96% for the classifications of different kissing bugs species is achieved. Finally, a web application is developed based on the proposed model to help the community collecting and identifying kissing bugs species.

INDEX TERMS
Chagas Disease, Convolutional Neural Network, Kissing Bugs, Transfer Learning

I. INTRODUCTION

In Latin America, Trypanosoma cruzi (T. cruzi.), also known as kissing bug, was identified as the originator of Chagas disease. This disease was initially discovered in 1909 by Carlos Chagas named after him [1]. Worldwide, around seven million people were infected with chronic Chagas disease, according to the World Health Organization WHO [2]. The genetic typing schemes of kissing bugs were successfully classified into six near-clades (TcI to TcVI) known as discrete typing units (DTU), and recently a new DTU was discovered in the south and Central American bats with the name TcBat. This genetic diversity is based on pathogenesis, clinical features, and geographical distribution [3].

Chagas disease has acute and chronic phases. The acute phase is usually mild. Around 20-35% of the infected people in the acute phase develop organ involvement and do not need special treatment [1]. The diagnosis of acute diseases is made by microscopic visualization of trypomastigotes in blood. The Parasites can be checked by a simple, fresh blood examination, microhaematocrit, and the strout method [4]. During the chronic phase, diagnosis is made by the detection of antibodies against T. cruzi. The efficacy of treatment decreases with the increase of infection time. Consequently, early detection and intervention of T. cruzi infection are essential in both endemic and non-endemic areas [5].

From the biological view, not all triatomine bug species are born equal. They belong to different families. The danger of each species is not the same, based on how they interact with humans. If we succeeded in making a classification for each species, it would help the public authorities to create a controlled surveillance system [6].

The classification of triatomine species was derived from morphological characters and cytogenetic and molecular analyses. Using this technique, Kelly Cristine [7] created a cytogenetic identification key procedure for species in Brazilian states. Another experimental method was used by Rachel Curtis-Robles [8, 9] to classify different triatomine bug species in the USA, especially in Texas. There is a significant cost involved with the clinical approaches in terms of the required resources. In addition to requiring DNA extraction kits and make PCR tests for the extracted samples, it is also time-consuming to extract the samples, conduct the requisition tests on them, and wait for the results. Moreover, all of these steps require highly experienced researchers to achieve accurate results [10].

Other approaches that are more rapid and cost-effective, and do not require a high level of expertise include the development of automated classification systems that can differentiate between the triatomine bug species using supervised...
machine learning, data mining, and deep learning techniques. In 2017, a fully automated system to classify 39 Brazilian and 12 Mexican kissing bug species was proposed in [11]. The classification result for Brazilian species was 87.8%, while the Mexican species was 80.3%. This research group was able to improve their previously achieved classification accuracy to 86.7% for Mexican species by using a simple deep learning model. However, the accuracy of their work is still not high enough and their work was restricted to images captured by invention device with certain specifications [12]. To improve the classification results, Zeinab Ghasemi et al. used different data mining techniques to classify Brazilian and Mexican kissing bug species [13]. Although the classification results improved impressively, different methods were required to extract and select the most discriminative features.

This study proposes a fully automated deep learning model using a convolutional neural network (CNN) with a fine-tuned transfer learning model. This model can identify and classify kissing versus non-kissing bugs and also classify different kissing bug species in Mexico and Brazil. In this model, feature extraction, feature selection, and classification are combined in the deep learning method.

The proposed model’s classification accuracy is almost 99.79% for the kissing versus non-kissing bugs model, 97.32% for Mexican kissing bug species, and 96.42% for Brazilian kissing bug species. A web application has also been developed to allow the users to upload a bug image, determine whether it is a kissing bug or not, and identify kissing bug species.

The paper is organized as follows: Section II discusses the related work; Section III describes the proposed model and underlying methodologies used to create it. Dataset, results, and discussion are presented in Section IV. Finally, section V concludes a summary of the proposed work and discusses future work.

II. RELATED WORK

A Texas University team performed the first study to determine the temporal and spatial variation of triatomine activity within the US in 2011 [9]. They collected specimens using standard entomological trapping techniques and encouraging US citizens to report and deliver insects to them within a citizen science program from May 2012 until November 2016. They succeeded in collecting a total of 3215 triatomine specimens. Specimens came from more than 534 locations (3,006 from Texas and remaining from 17 other states). Specimens came from different areas (peridomestic areas, dog kennels, and indoors). The collected triatomines belonged to 7 species: T.gerstaeckeri, T.indictiva, T.lecticularia, T.neotomaie, T.protracta, T.rubida, and T.sanguisuga. The seven species T. gerstaeckeri represents 63.6% of the total collected samples with a higher ratio of females than males. This study focused only on the classification of the different triatomines species. It was concluded that the peak activity period for T. gerstaeckeri is from April to September and from June to October for T. sanguisuga.

In 2018, the same team made another study to check if the faces of 1510 triatomine insects contain T. cruzi DNA by using multiplex PCR assays [8]. Amplification of a 166-bp region technique was used to check the T. cruzi infection status within the triatomine insects. This study revealed that 822 out of 1510 insects tested positive with the T. cruzi parasites and adult insects are ten times more likely to have the disease than the nymphs [8]. The employed experimental technique is relatively expensive in terms of the required resources and expert personnel and takes a long time to get the results.

The second group of studies focuses on employing computational methods to identify and classify Chagas disease vectors instead of time-consuming and costly experimental methods. In 2017, Gurgel-Gonc Alves developed a fully-automated system to classify 39 species of Brazilian kissing bugs and 12 species of Mexican kissing [11]. First, they employed a spatial apparatus shown in Fig.1 to capture images of triatomine insects. Then, captured images were preprocessed using lens distortion correction, background removal, specimen’s body edge identification, clipping the legs, clipping the antennas, and smoothing the clipped edge. Ten geometrical landmarks were extracted from the final output images. Finally, a simple feedforward neural network model was employed for the classification achieving 80.3% and 87.8% accuracy for Mexican and Brazilian species, respectively. This model had low accuracy for Mexican species and was restricted to the good preprocessed quality images generated by a special apparatus. The same research team used a deep learning model that accepts the special apparatus’s images without the need to do any further preprocessing steps to overcome these drawbacks [12]. The output accuracy improved to be 83.0% for Mexican species, but for Brazilian species, it decreases to 86.7%.

In 2020, Ghasemi et al. applied several data mining techniques to overcome the drawbacks of the Gurgel-Gonc Alves. As input data, they used the same dataset from the Kansas team [13]. First, a preprocessing step on input images using the image segmentation K-means clustering method...
was employed to separate the insect’s body’s background. The next step was to use Principal Component Analysis (PCA) to extract the best 150 features from the preprocessed images. For classification, Support Vector Machine (SVM) and Random Forest (RF) were employed. The classification accuracy of 75.3% for Brazilian and 87.7% for Mexican species with SVM and 100% for both Brazilian and Mexican species with RF was reported in this study. The drawback of this study is the need of employing several methods for image preprocessing, feature extraction, feature selection, and classification. Ghasemi and Banitaan applied two convolutional neural networks (CNN) structures to the dataset [14]. The first structure was a pre-trained VGG16 with ImageNet dataset and the second structure was a 7-layer CNN. VGG16 achieved an accuracy of 86.52% and 87.93%, while the 7-layer CNN achieved an accuracy of 96.82% and 96.53% to classify Brazilian and Mexican kissing bug species.

In [15], a website named TriatoKey was created with a PostgreSQL database for the known triatomine bugs species in Brazil. It uses a series of yes/no questions to guide the user in the direction of the correct classification or taxon identification. HTML5, CSS, and Javascript were used to build this website. The drawback of this website is that it is limited to Brazilian species only. Also, it depends on the user to describe the insect using yes/no questions, so any human mistakes or question misunderstanding will lead to the wrong classification.

This paper proposes a fully automated deep learning model based on Convolutional Neural Networks (CNN) not only for the identification of kissing bugs but also for the classification of different kissing bug species. To the best of our knowledge, this is the first work conducted to identify kissing versus non-kissing bugs. Furthermore, in this study, the weights of the feature extraction layers are pre-trained using a transfer learning technique to accelerate the training phase. The proposed model overcomes the shortcomings of models previously reported in the literature, such as the need for using separate methods for extracting and selecting features and the availability of high-quality images. Finally, a web-based application has been developed using streamlit API to provide experts and non-experts with the ability to collect and identify the type of suspected bug images.

III. METHODOLOGY

This section discusses the proposed model and all technologies used to create it.

A. CONVOLUTIONAL NEURAL NETWORK

CNN is one of the most in-demand algorithms for deep learning. Using it, a model learns to perform classification tasks directly from images, video, or text. It is excellent for finding patterns in images to recognize objects. CNN’s main components are an input layer, an output layer, and many hidden layers in between. After learning features in the hidden layers, the architecture of CNN shifts to classification. The final layer uses a classification layer, such as softmax, to provide the classification output. CNN is trained on hundreds, thousands, or even millions of images. Once the CNN was trained, it can be used in real-time applications. As shown in Fig 2, the main building blocks of CNN can be view as a convolutional layer, pooling layer, and fully connected layer [16].

The convolutional layer is the significant layer and significant step on the CNN model, and within it, the model does most of the heavy computational tasks. It creates a feature map by making a convolution operation between the input and a set of filters. The filter is a grid of numbers in a squared shape that stores a single pattern within the input image. It can be considered a feature selector from the input image, starting from the upper left section. The convolution operation performs a dot product between the filter parameter and the matching grid from the input. The result from that operation shows the existence of the feature in that section of the image. Then, the operation will continue by slide the filter from left to right and from top to bottom, and in each step, calculate the dot product again. After testing the full image for that feature, the dot product stores’ output into the feature map in a spatial grid structure. The next layers of CNN heavily depend on that spatial relationship. In each convolutional layer, there are several filters. Each one of them stores a single pattern within the input image, creating a 2D feature map. By stacking all the 2D feature maps together, the final output from the convolutional layer is generated in a 3D feature map. The main hyperparameters of the convolutional layer are filter count, filter size, and stride size. Using more filters gives a robust neural network but also increases the possibility of overfitting. Most of the time, the number of filters within the convolutional layer is between 32 and 512. Filter size indicates the height and width of the filters defining their spatial extent. Usually, small-size filters are used on CNN to reduce the number of learnable parameters ensuring individual patterns are learned from local regions. Stride size specifies as the number of pixels in which the filter window moves, usually sets to be 1. If we need the filter to slide with a more significant interval, the stride size can be increased [17].

The pooling layer on CNN is used to reduce the spatial size of the convolutional layer’s representation. It mainly creates a simple version of the same information. The most common

---

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
pooling layer is max-pooling. The max-pooling layer takes the max value in the sliding window, discarding all other values. For the pooling layer, the filter size and stride size should be specified [17].

The flatten layer on CNN is used to transfer $N$-dimensional tensor from the last max-pooling layer into a 1-dimensional tensor by traversing pixels in row-major order. That operation is mandatory on CNN before starting the classification process in the next fully connected layer [18].

The fully connected layer is the last layer of a CNN used to classify the features detected and extracted by the convolutional layers and pooling layers. The features maps are flattened into a single 1D vector to input them into the fully connected layers [17].

### B. TRANSFER LEARNING

There are two approaches when working with the CNN model (a) building a CNN model from scratch and (b) use a pre-trained CNN model. Training a model from scratch provides maximum control over the network and can produce impressive results if we have many images as input. Using a fine-tuned pre-trained model with transfer learning is typically much faster and easier than training from scratch. Also, it requires the least amount of input data images and less computational resources [16].

Transfer learning aims to adjust the parameters of an already trained model to adapt to the new task at hand. In this way, the dependence on numerous target-domain data can be reduced for constructing target learners [19]. The transfer learning approach can be used when a dataset has a few data to train a full-scale model from the beginning. In the pre-trained model, the first layers tend to learn very general features. Moving to the higher levels of the model, more specific patterns to the task being trained can be learned. So to fine-tune the model to a specific task, the first layers can be kept intact (or freeze them) and retrain the upper layers for the task[20].

Several pre-trained networks are available to enhance the training of deeper networks, including Xception [21], VGG16, VGG19 [22], ResNet50 [23], InceptionV3 [24], MobileNet [25], MobileNetV2 [26], DenseNet [27], and NASNet [28]. ResNet, introduced by the Microsoft team, uses the residual mapping technique to address the issue of weight disappearing/exploding [27]. Inception V3 employs filters of different sizes at the same level, rather than stacking layers deeper as used in previous CNNs [24]. DenseNet refers to a network of densely connected convolutional layers with each layer connected to every other layer in a feed-forward fashion [27]. VGG, a neural network with simple architecture based on the traditional CNNs, was introduced by a team of Oxford University in the ImageNet Challenge in 2014. It has two versions, VGG16 and VGG19, with different numbers of layers [29]. VGG16 is a CNN model that won the ImageNet large scale visual recognition challenge competition in 2014 [30]. It is one of the excellent vision model architecture to date [31]. VGG16 consists of 5 convolutional blocks, which, the last convolutional block is connected to the classifier. The classifier consists of two hidden layers, and an output layer is connected directly to the number of classes. The default number of classes in the VGG16 model is 1000 classes [32].

![Diagram of the proposed framework for the classification of kissing and non-kissing bugs and kissing bugs species.](image-url)
C. THE PROPOSED MODEL

In this paper, a modified CNN model based on VGG16 architecture has been proposed for the binary classification of kissing and non-kissing bugs and also multi-label classification of Brazilian and Mexican kissing bugs species. The detailed information of our proposed model layers, including layer number, type, description, and size is shown in Table 1. Also, the architecture of the proposed model including the original VGG16 layers (section A) and the added layers to create our specific classification task (Section B) is shown in Fig.3. In this figure, n in the final softmax layer refers to the number of classes. For the classification of kissing and non-kissing bugs, n is 2 and for the classification of Brazilian and Mexican kissing bugs species, n is 39 and 12, respectively. Due to the relatively small number of images in training and validation datasets, a pre-trained VGG16 model is used to benefit from its pre-trained weights to extract features from our new images and modify the last layers to fit our application. As explained before, the first layers of the VGG16 model learn the basic features of the images while the later layers learn the most specific features. Several experiments have been made in this study to find the best number of trainable layers which improves the accuracy of the proposed model.

Also, due to an unbalanced distribution in the number of images in each class in our dataset, cross-validation is applied to ensure generalization and provide a more reliable model. Cross-validation is defined as a data resampling technique to assess the generalization of classification models and prevent overfitting [33]. There are many types of cross-validation, such as single hold-out random subsampling, k-fold random subsampling, k-fold cross-validation, Leave-one-out cross-validation, and Jackknife. In our model, the 5-fold cross-validation technique is used.

### TABLE 1. The layers numbers and description of the proposed model.

| Layer No. | Layer Type | Layer Name Desc | Layer Size |
|-----------|------------|-----------------|------------|
| L1        | Convolution| Conv-1-1        | 7 x 7 x 512 |
| L2        | Convolution| Conv-1-2        | 14 x 14 x 512 |
| L3        | Convolution| Max-Pooling - 1 | 14 x 14 x 512 |
| L4        | Convolution| Conv-2-1        | 14 x 14 x 512 |
| L5        | Pooling    | Conv-2-2        | 14 x 14 x 512 |
| L6        | Convolution| Max-Pooling - 2 | 28 x 28 x 512 |
| L7        | Convolution| Conv-3-1        | 28 x 28 x 512 |
| L8        | Convolution| Conv-3-2        | 28 x 28 x 512 |
| L9        | Pooling    | Conv-3-3        | 28 x 28 x 512 |
| L10       | Convolution| Max-Pooling - 3 | 56 x 56 x 256 |
| L11       | Convolution| Conv-4-1        | 56 x 56 x 256 |
| L12       | Convolution| Conv-4-2        | 56 x 56 x 256 |
| L13       | Pooling    | Conv-4-3        | 56 x 56 x 256 |
| L14       | Convolution| Max-Pooling - 4 | 112 x 112 x 128 |
| L15       | Convolution| Conv-5-1        | 112 x 112 x 128 |
| L16       | Pooling    | Conv-5-2        | 112 x 112 x 128 |
| L17       | Convolution| Conv-5-3        | 224 x 224 x 64 |
| L18       | Convolution| Max-Pooling - 5 | 224 x 224 x 64 |
| L19       | Flatten    | Flatten - 1     | 1 x 1 x 25088 |
| L20       | Fully Connected | FC - 1          | 1 x 1 x 1024 |
| L21       | Fully Connected | FC - 12         | 1 x 1 x 1024 |
| L22       | Output     | SoftMax-1       | 1 x 1 x n |

IV. RESULTS AND DISCUSSION

In this section, the performance of the proposed model has been tested and compared against the state-of-the-art models over two different image datasets.

A. EVALUATION METRICS

In this study, the identification of kissing bug is a binary classification problem while the classification of kissing bug species is a multi-label classification problem. For binary classification, the performance of the prediction model can be measured in terms of the model’s precision, recall, and accuracy as defined in the following equations:

\[
\text{precision} = \frac{TP}{TP + FP} \quad (1)
\]

\[
\text{recall} = \frac{TP}{TP + FN} \quad (2)
\]

\[
\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (3)
\]

where, as shown in Table 2, the number of correctly classified kissing (positive) and non-kissing (negative) bugs named True Positive (TP) and True negative (TN), respectively. The number of kissing (non-kissing) bugs incorrectly predicted as non-kissing (kissing) bugs named False negative (FN) (False positive (FP)).

The same evaluation metrics can be calculated with a small adjustment for problems involving multi-label classification. A one-vs-all approach with a micro-averaging concept is used in this study, where one class represents positive samples and the remaining classes are combined to represent

### TABLE 2. Classification Confusion Matrix

| Predicted Positive (1) | Actually Positive (1) | Actually Negative (0) |
|------------------------|-----------------------|----------------------|
| Predicted Positive (1) | True Positive (TP)    | False Positive (FP)  |
| Predicted Negative (0) | False Negative (FN)   | True Negative (TN)   |
negative samples. The micro-average precision measure, for example, is calculated as follows:

\[
\text{precision}_\text{micro} = \frac{TP_1 + \ldots + TP_k}{TP_1 + \ldots + TP_k + FP_1 + \ldots + FP_k}
\]

where \(k\) is the number of classes.

In addition to the previous evaluation measures, the Receiver Operating Characteristic (ROC) curve and the Area Under the Curve (AUC) are utilized in this study [34]. A ROC curve represents the performance of a classification model at all classification thresholds by plotting True Positive Ratio (TPR) against False Positive Ratio (FPR).

As mentioned before, to avoid the overfitting problem, cross validation can be used. The most common form of cross-validation is \(k\)-fold cross-validation where the data is divided into \(k\) equally sized folds. Subsequently, \(k\) iterations of training and validation are performed such that within each iteration, a different fold of the data is held out for validation while the remaining \(k-1\) folds are used for learning. In all of our experiments, 5-fold cross validation is employed.

**B. DATASET**

Two different image datasets are used to test the proposed model and perform an in-depth analysis of its strength. The first dataset, created by the biology team in Kansas’s university, contains 410 high-quality images for 12 Mexican species and 1620 high-quality images for 39 Brazilian triatomine bugs species [11]. To this dataset, 160 standard-quality images for Brazilian triatomine bugs gathered from Texas A&M University’s website [35], Bugwood Network [36], and Google images search are also added.

The second dataset contains images of non-kissing bugs similar to kissing bugs downloaded from Google images search. This dataset contains 2190 kissing bug images and 3396 images of 11 different non-kissing bug species, including Rochnema, Eptoglossus, Clypealis, Leptoglossus, Phyllopus, Microtoma, Milkweed_Bug, Mozena_obtusa, Parapiacaba_tricincta, Squash_bug, Weevil, Wheel_bug, and Zelus_longipes. An example of a high-quality kissing bug image, a standard-quality kissing bug image, and a non-kissing bug image is shown in Fig. 4.

**C. ANALYSIS OF TRANSFER LEARNING ARCHITECTURES**

As mentioned before, transfer learning refers to the process of storing knowledge obtained from solving one problem and applying it to a different but related problem. Several pre-trained transfer learning architectures, including VGG16, VGG19, RESNET 50, DenseNet201, and inception v3, are tested and compared in this study to select the best network for the identification and classification of kissing bugs. The results of the above pre-trained networks for the classifications of Mexican and Brazilian kissing bugs species are shown in Table 3. Based on the accuracy, VGG16 was chosen as the pre-trained network in this study.

**D. ANALYSIS OF KISSING AND NON-KISSING BUGS CLASSIFICATION**

The pre-trained VGG16 model is applied to the kissing/non-kissing bug images dataset to differentiate between them. This model was trained for 20 epochs with a learning rate of 0.0001 and 5-fold cross-validation. As discussed before, the first layers of the VGG16 model extract the images’ basic features, while the later layers extract the most complex ones. To get the best results for that specific features, the number of trainable layers and the number of frozen non-trainable layers of the original VGG16 model should be tuned. The first 15 layers of the VGG16 model (L1 to L15) are set as non-trainable whereas the last three layers (L16 to L18) are set as trainable layers, leading to 99.51% accuracy as shown in Table 4. The number of trainable layers was increased once at a time (from 3 to 7 layers) to detect the best combination of trainable and non-trainable layers as shown in Table 4. The best accuracy of 99.79% is obtained with six trainable layers (L13 to L18) in the VGG16 model. The accuracy, precision, and recall per each class in this dataset are reported in Table 5.

**E. ANALYSIS OF MEXICAN AND BRAZILIAN SPECIES CLASSIFICATION**

The proposed model is used to classify the 39 different Brazilian and 12 Mexican kissing bug species. The same approach to determine the best number of trainable and non-trainable VGG16 layers is used. Based on the results shown in Table 6, the best accuracy for the Brazilian and Mexican are 96.42% and 97.32%, respectively achieved with six trainable layers (L13-L18) of the VGG16 model.

The graphs in Fig. 5 show the training and validation accuracy of our model for Brazilian and Mexican datasets which increase exponentially at almost the same rate. The increase in the accuracy proves the effectiveness of our...
TABLE 5. Classification results of kissing vs non-kissing bugs considering six trainable layers

| Species name     | accuracy | precision | recall |
|------------------|----------|-----------|--------|
| Kissing bugs     | 99.73%   | 97%       | 99%    |
| Non-kissing bugs | 99.85%   | 100%      | 98%    |
| Average          | 99.79%   | 98.5%     | 98.5%  |

FIGURE 5. Classification results of 39 Brazilian kissing bug species (left), 12 Mexican kissing bug species (right)

The proposed model and shows also that there is no overfitting on it.

The ROC curves for Mexican and Brazilian kissing bug species are plotted in Fig.6. The area under the ROC curve for Brazilian and Mexican datasets are 99.9% and 99% respectively.

The accuracy, precision, and recall per each class in the Brazilian and Mexican species datasets are shown in Table 7 and Table 8, respectively. Rows with accuracy less than 100% are highlighted. As shown in Table 7, most of the mis-classification of the Brazilian species are in Triatoma arthurneivai class (50% accuracy). In this class, we have fewer images for training. Moreover, checking the predicted results, we found that some of the Triatoma arthurneivai are mis-classified into Triatoma pseudomaculata and Triatoma guazu classes. As these three species are very similar to each other, more images are required to train the model more accurately. Similarly, as shown in Table 8, most of the mis-classification of the Mexican dataset are in Triatoma mazzotti class (50% accuracy) which mis-classified to Triatoma mazzotti and Triatoma longipenni. These three species are also very similar to each other. So, more training images for training the model are required to improve this result.

F. COMPARISON WITH STATE-OF-THE-ART APPROACHES

A rough comparison between the results of our proposed model and those from other related studies including feed forward neural network (FFNN) [11], deep neural networks

TABLE 7. Classification results of the 39 Brazilian species

| Species name           | No. of Images | Accuracy | precision | recall |
|------------------------|--------------|----------|-----------|--------|
| Cavernicola lenti      | 39           | 100%     | 100%      | 100%   |
| Panstrongylus diasi    | 35           | 100%     | 100%      | 100%   |
| Panstrongylus geniculatus | 58       | 100%     | 87%       | 100%   |
| Panstrongylus lignatus | 39           | 85%      | 80%       | 77%    |
| Panstrongylus latzi    | 35           | 95%      | 95%       | 87%    |
| Panstrongylus megvisus | 99           | 100%     | 89%       | 100%   |
| Psammolestes tertius   | 39           | 100%     | 100%      | 100%   |
| Rhodinus brethesi      | 31           | 100%     | 100%      | 100%   |
| Rhodinus domesticus    | 32           | 100%     | 100%      | 100%   |
| Rhodinus mili           | 46           | 100%     | 100%      | 89%    |
| Rhodinus montenegrinssi | 45          | 97%      | 82%       | 100%   |
| Rhodinus nasutus       | 84           | 75%      | 100%      | 100%   |
| Rhodinus neglectus     | 71           | 100%     | 100%      | 93%    |
| Rhodinus pictipes      | 62           | 100%     | 100%      | 100%   |
| Triatoma arthurneivai  | 35           | 50%      | 100%      | 83%    |
| Triatoma barberai      | 32           | 100%     | 86%       | 100%   |
| Triatoma brasiliensis  | 74           | 100%     | 59%       | 67%    |
| Triatoma carcavaliata  | 41           | 100%     | 100%      | 100%   |
| Triatoma circunmaculata | 22        | 100%     | 100%      | 100%   |
| Triatoma costalimai    | 68           | 95%      | 98%       | 100%   |
| Triatoma delponenti    | 31           | 93%      | 100%      | 80%    |
| Triatoma dimidiata H21 | 44           | 100%     | 100%      | 89%    |
| Triatoma guazu         | 32           | 86%      | 86%       | 100%   |
| Triatoma infestans     | 66           | 100%     | 100%      | 100%   |
| Triatoma jauaizerensis | 26           | 75%      | 67%       | 80%    |
| Triatoma lenti         | 47           | 100%     | 86%       | 67%    |
| Triatoma maculata      | 44           | 100%     | 92%       | 96%    |
| Triatoma matogrossensis | 35          | 100%     | 83%       | 83%    |
| Triatoma melancia      | 36           | 100%     | 86%       | 86%    |
| Triatoma panstrongylus | 29           | 100%     | 100%      | 100%   |
| Triatoma platensis     | 30           | 100%     | 86%       | 100%   |
| Triatoma pseudomaculata | 60          | 100%     | 100%      | 83%    |
| Triatoma rubrovaria    | 61           | 100%     | 100%      | 58%    |
| Triatoma sherlocki     | 35           | 100%     | 58%       | 100%   |
| Triatoma sorrida       | 108          | 100%     | 100%      | 93%    |
| Triatoma tubuamaculata | 41           | 100%     | 100%      | 89%    |
| Triatoma vandae        | 32           | 100%     | 100%      | 83%    |
| Triatoma villiceps     | 50           | 100%     | 100%      | 100%   |
| Triatoma williami      | 25           | 80%      | 89%       | 98%    |
| **Total**              | **1812**     | **96%**  | **92%**   | **91%**|

TABLE 8. Classification results of the 12 Mexican species

| Species name            | No. of Images | Accuracy | precision | recall |
|-------------------------|--------------|----------|-----------|--------|
| Panstrongylus rufomargaritaria | 12        | 100%     | 100%      | 100%   |
| Triatoma berberi        | 31           | 100%     | 96%       | 100%   |
| Triatoma dimidiata H1   | 44           | 100%     | 88%       | 79%    |
| Triatoma dimidiata H2   | 40           | 95%      | 100%      | 88%    |
| Triatoma dimidiata H3   | 40           | 100%     | 93%       | 88%    |
| Triatoma gersaukeri     | 12           | 100%     | 100%      | 77%    |
| Triatoma longipenni     | 52           | 100%     | 80%       | 80%    |
| Triatoma mazzotti       | 23           | 70%      | 100%      | 100%   |
| Triatoma mexicana       | 46           | 100%     | 90%       | 100%   |
| Triatoma nitida         | 16           | 100%     | 100%      | 100%   |
| Triatoma pallidipennis  | 44           | 100%     | 90%       | 100%   |
| Triatoma phyllonotus    | 60           | 100%     | 82%       | 92%    |
| **Total**               | **410**      | **97%**  | **94%**   | **93%**|
This paper presented an approach to automatically identify kissing bugs and classify Brazilian and Mexican kissing bug species. By using CNN with transfer learning, our model made the feature selection, feature extraction, and classification automatically with very high accuracy. So, if new species are added to the original database, the model can be trained automatically to improve the performance. Moreover, a web application has been developed using Streamlit python library based on the trained model. The application accepts an uploaded image as an input and displays the predicted classification. The collected images will be available to the community. Future directions include automated prediction of the T. cruzi Parasite’s presence within kissing bugs based on its images.

REFERENCES

[1] Á. Moncayo and A. C. Silveira, “Current epidemiological trends of chagas disease in latin america and future challenges: Epidemiology, surveillance, and health policies,” in American Trypanosomiasis Chagas Disease. Elsevier, 2017, pp. 59–88.
[2] WHO, “Chagas disease,” https://www.who.int/news-room/fact-sheets/detail/chagas-disease-(american-trypanosomiasis), March 2020.
[3] V. Balouz, F. Agüero, and C. A. Buscaglia, “Chagas disease diagnostic applications: present knowledge and future steps,” in Advances in parasitology. Elsevier, 2017, vol. 97, pp. 1–45.
[4] D. Ordoñez, P. Fernández-Soto, A. M. Fernández-Martín, B. Crego-Vicente, B. Febrer-Sendra, J. G.-B. Diego, B. Vicente, J. López-Abán, M. Belhassen-García, A. Muro et al., “A trypanosoma cruzi genome tandem repetitive satellite dna sequence as a molecular marker for a lamp assay for diagnosing chagas’ disease,” Disease markers, vol. 2020, 2020.
[5] J. A. Pérez-Molina and I. Molina, “Chagas disease,” The Lancet, vol. 391, no. 10115, pp. 82–94, 2018.
[6] F. Abad-Franch, “A simple, biologically sound, and potentially useful working classification of chagas disease vectors,” Memorias do Instituto Oswaldo Cruz, vol. 111, no. 10, pp. 649–651, 2016.
[7] K. C. Borsatto, M. T. V. Azeredo-Oliveira, and K. C. C. Alevi, “Identification key for the chagas disease vectors,” Memorias do Instituto Oswaldo Cruz, vol. 117, no. 4, pp. 491–501, 2021.
[8] R. Curtis-Robles, L. D. Auckland, K. F. Snowden, G. L. Hamer, and S. A. Hamer, “Analysis of over 1500 triatomine vectors from across the us, predominantly texas, for trypanosoma cruzi infection and discrete typing units,” Infection, Genetics and Evolution, vol. 58, pp. 171–180, 2018.
[9] R. Curtis-Robles, S. A. Hamer, S. Lane, M. Z. Levy, and G. L. Hamer, “Bionomics and spatial distribution of triatomine vectors of trypanosoma cruzi in texas and other southern states, usa,” The American journal of tropical medicine and hygiene, vol. 98, no. 1, pp. 113–
[10] L. A. Shender, M. D. Lewis, D. Rejmanek, and J. A. Mazet, “Molecular diversity of trypanosoma cruzi detected in the vector triatoma protracta from california, usa,” PLoS neglected tropical diseases, vol. 10, no. 1, p. e0004291, 2016.

[11] R. Gurgel-Gonçalves, E. Komp, L. P. Campbell, A. Khalighifar, J. Mellenbruch, V. J. Mendoçanha, H. L. Owens, K. de la Cruz Felix, A. T. Peterson, and J. M. Ramsey, “Automated identification of insect vectors of chagas disease in brazil and mexico: the virtual vector lab,” PeerJ, vol. 5, p. e3040, 2017.

[12] A. Khalighifar, E. Komp, J. M. Ramsey, R. Gurgel-Gonçalves, and A. T. Peterson, “Deep learning algorithms improve automated identification of chagas disease vectors,” Journal of medical entomology, vol. 56, no. 5, pp. 1404–1410, 2019.

[13] Z. Ghasemi, S. Banitaan, and G. Al-Refaie, “Automated chagas disease vectors identification using data mining techniques,” in 2020 IEEE International Conference on Electro Information Technology (EIT), 2020, pp. 540–545.

[14] Z. Parsons and S. Banitaan, “Automatic identification of chagas disease vectors using data mining and deep learning techniques,” Ecological Informatics, p. 101270, 2021.

[15] L. Márcia de Oliveira, R. Nogueira de Brito, P. Anderson Souza Guimarães, R. Vitor Mastrângelo Amaro dos Santos, L. Gonçalves Diotaiauti, R. de Cássia Moreira de Souza, and J. Conceição Ruiz, “Triatokey: a web and mobile tool for biodiversity identification of brazilian triatomine species,” Database, vol. 2017, 2017.

[16] I. MathWorks, “Convolutional neural network),” https://www.mathworks.com/solutions/deep-learning/convolutional-neural-network.html, August 2020.

[17] L. Lu, Y. Zheng, G. Carneiro, and L. Yang, “Deep learning and convolutional neural networks for medical image computing,” Advances in Computer Vision and Pattern Recognition, 2017.

[18] Z. J. Wang, R. Turko, O. Shaikh, H. Park, N. Das, F. Hohman, M. Kahng, and D. H. Chau, “Cnn explainer: Learning convolutional neural networks with interactive visualization,” arXiv preprint arXiv:2004.15004, 2020.

[19] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, “A comprehensive survey on transfer learning,” Proceedings of the IEEE, 2020.

[20] OpenCV .org, “Fine-tuning using pre-trained models),” https://www.learnopencv.com/keras-tutorial-fine-tuning-using-pre-trained-models/, February 2018.

[21] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1251–1258.

[22] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” arXiv preprint arXiv:1409.1556, 2014.

[23] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[24] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2818–2826.

[25] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” arXiv preprint arXiv:1704.04861, 2017.
[26] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 4510–4520.

[27] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4700–4708.

[28] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, “Learning transferable architectures for scalable image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8697–8710.

[29] V. Kurama, “A review of popular deep learning architectures: Alexnet, vgg16, and googlenet,” https://blog.paperspace.com/popular-deep-learning-architectures-alexnet-vgg-googlenet/, July 2020.

[30] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in 2009 IEEE conference on computer vision and pattern recognition. Ieee, 2009, pp. 248–255.

[31] R. Thakur, “Step by step vgg16 implementation in keras for beginners,” https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686a6c, August 2019.

[32] P. Hridayami, I. K. G. D. Putra, and K. S. Wibawa, “Fish species recognition using vgg16 deep convolutional neural network,” Journal of Computing Science and Engineering, vol. 13, no. 3, pp. 124–130, 2019.

[33] D. Berrar, “Cross-validation,” Encyclopedia of bioinformatics and computational biology, vol. 1, pp. 542–545, 2019.

[34] A. C. J. Janssens and F. K. Martens, “Reflection on modern methods: revisiting the area under the roc curve,” International journal of epidemiology, vol. 49, no. 4, pp. 1397–1403, 2020.

[35] T. A. University, “Kissing bug and chages disease),” https://kissingbug.tamu.edu/, Aug 2020.

[36] B. Images, “Insect images),” https://www.insectimages.org/, October 2018.

[37] A. Treuille, “streamlit),” https://www.streamlit.io/, August 2020.

BASSAM ABDELGHANI received the B.S. degree in electrical and communication engineering from Fayoum University, Fayoum, Egypt, in 2011. He is currently pursuing the M.S. degree in Electrical and computer engineering from University of Detroit mercy, MI, USA. His research interest includes the Machine learning, Deep learning, and AI.

DR. SHADI BANITAAN is currently an Associate Professor and the Director of Computer Science and Software Engineering at the University of Detroit Mercy. His research interests include software engineering, machine learning, and data mining. He is a member of the Association for Computing Machinery (ACM), the Institute of Electrical and Electronics Engineers (IEEE), and the IEEE Computer Society. He received a B.S. degree in Computer Science from Yarmouk University, an M.S. degree in Computer and Information Sciences from Yarmouk University, and a Ph.D. degree in Computer Science from North Dakota State University. He worked as an instructor at the University of Nizwa from 2004 to 2009. He joined the University of Detroit Mercy in 2013.

DR. MINA MALEKI is an Assistant Professor of Software Engineering and Computer Science at the University of Detroit Mercy, MI, USA. Prior to this, she was a sessional instructor at the University of Windsor and a SOSCIP TalentEdge post-doctoral research fellow at the Cross-Border Institute (CBI) at the University of Windsor, Canada. Her main research interests are machine learning, deep learning, pattern recognition, and big data analysis.

AMNA MAZEN received the B.S. degree in electrical and Power engineering from Fayoum University, Fayoum, Egypt, in 2013. Received the M.S. degree in electrical and Power engineering from Fayoum University, Fayoum, Egypt, in 2018. She is currently pursuing the Ph.D. degree in Electrical and computer engineering from University of Detroit mercy, MI, USA. From 2019 to 2021, she was a Research Assistant with the Electrical and computer engineering from University of Detroit mercy, MI, USA. Her research interest includes the Robotics, Deep learning, and AI.

***