Institut supérieur d'électronique de Paris

Engineering program

Report

"To Bee or not to bee"

Students :

MERBOUCHE Mouloud
TSO Célia
DAHIM Miriam
BOUNOUAR Haylena

Classe : A2

Supervisors :

SUBLIME Jérémie
URIEN Hélène
OH Seoyoung

Course code : IG.2411 & IG.2412

Academic year 2024-2025

Contents

1 Introduction and objectives

Accurate insect classification plays a crucial role in biodiversity monitoring, pest management, and ecological research. With declining insect populations worldwide, automated identification systems can significantly accelerate species surveys and conservation efforts. This project addresses the challenge of classifying six insect types (bee, bumblebee, butterfly, dragonfly, hover fly, and wasp) from RGB images using computer vision and machine learning techniques.

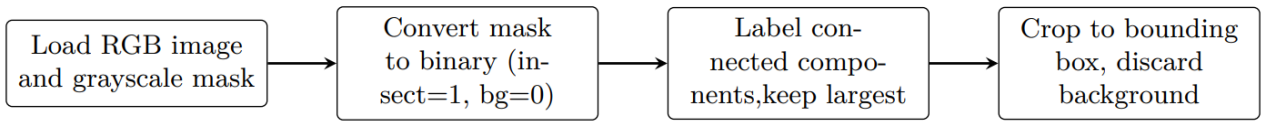
The primary objective is to develop and compare multiple classification approaches, from traditional feature engineering to deep learning, to identify the most effective method for automated insect recognition. We seek to extract discriminative features that capture morphological, textural, and color characteristics of different insect species while addressing the severe class imbalance present in the dataset. Through systematic evaluation of various supervised and unsupervised learning algorithms, we aim to understand their strengths and limitations in this challenging classification task, ultimately comparing classical machine learning approaches with modern deep learning techniques.

This report is structured as follows: Section 2 details the feature extraction pipeline, including preprocessing steps and the computation of mandatory and additional descriptors spanning shape, texture, and color domains. Section 3 presents comprehensive data visualization and exploratory analysis using PCA, t-SNE, and Isomap to understand the intrinsic structure of the feature space. Section 4 evaluates multiple machine learning approaches, beginning with feature selection via recursive elimination, followed by supervised methods including KNN, SVM, Random Forest, and Logistic Regression. We then explore unsupervised clustering to assess natural groupings before concluding with a deep learning implementation using MobileNetV2. Finally, we synthesize the results to identify the most promising approach for practical insect classification systems.

2 Feature Extraction

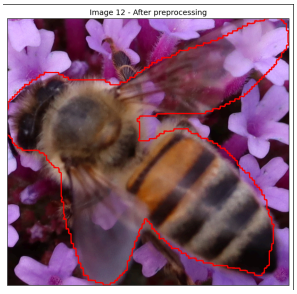
2.1 Preprocessing

Before extracting any features, we performed a thorough quality control to verify that each image in the training set had a corresponding mask. We discovered that the mask for image #154 was missing and therefore excluded this pair from our dataset, leaving 249 valid image-mask pairs. The following preprocessing pipeline was then applied to each remaining pair:



These steps ensure that subsequent feature computations are performed on a clean, tightly bounded region of interest. Figure 1 shows two representative outputs from this pipeline:

(a) Cleaned binary mask after largest-component selection (image #012).



(b) Cropped RGB region of interest aligned to the mask (image #012).

Figure 1: Illustrations of preprocessing results: (a) cleaned mask, (b) cropped image.

2.2 Mandatory Features

After preprocessing, we directly computed three categories of features on each of the 249 validated insect ROIs or on the full image depending on the features calculated :

- Area ratio (computed on the the full image) :

$$\text{ratio\_area} = \frac{\#\{\text{insect pixels}\}}{\#\{\text{total image pixels}\}}.$$

- Color statistics, for each channel $C \in \{R, G, B\}$ (computed on the ROI) :

$$\min(C),\ \max(C),\ \overline{C},\ \text{median}(C),\ \sigma(C).$$

- Median and standard deviation are included above as $\text{median}(C)$ and $\sigma(C)$ (computed on the ROI).

This is the distribution of the insect-to-image area ratio over all 249 samples :

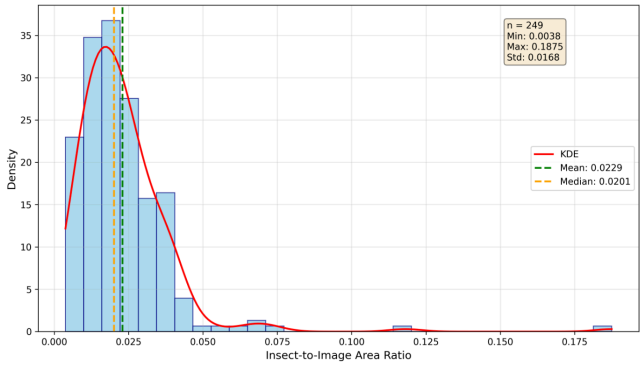


Figure 2: distribution of the insect-to-image area ratio over training set

The vast majority of ratios fall between 0.01 and 0.04, indicating that insects typically occupy only 1 %-4 % of the frame, with the highest density around 0.02. The mean value (0.0229) is slightly higher than the median (0.0201), revealing a right-skewed distribution caused by a small number of larger insects or very tight crops that push the ratio up to 0.12-0.18. The standard deviation (0.0168) confirms a moderate dispersion: while most specimens remain small, a few outliers occupy under 0.5 % or up to roughly 10 % of the image. Then, the area ratio proves to be a robust, normalized feature that effectively captures the scale of each insect and is therefore retained in our feature set.

This figure the distributions of the minimum, maximum, mean and standard deviation of pixel intensities for the R, G and B channels across all 249 insect masks :

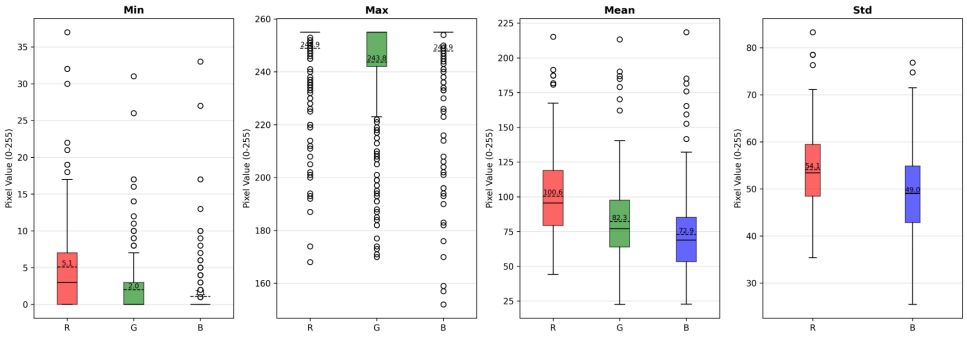


Figure 3: RGB Channel Statistics (min, max, mean, σ) across Dataset

The red channel exhibits the highest central tendency, with a median minimum around 5 and a median mean of approximately 100.6, reflecting the generally warm (reddish-brown) tones of the insects. In contrast, the green channel shows lower values overall (median mean $\simeq$ 82.3) and the smallest minima (median $\simeq$ 2), indicating darker or shadowed regions. The blue channel is intermediate (median mean $\simeq$ 72.9) but displays a wider spread in both minima and maxima, suggesting variable highlights or background bleed. The maximum values for all channels cluster near the upper end of the scale (median maxima: $R \simeq 248.9$, $G \simeq 243.8$, $B \simeq 247.9$), confirming that specular reflections often reach full intensity. Finally, the standard-deviation panel reveals that the red channel also has the greatest chromatic variability (median $\sigma \simeq 54.1$), while green and blue are slightly lower (median $\sigma \simeq 49.0$), signifying that red contrast is particularly informative for distinguishing texture and pattern on the insect exoskeleton.

2.2.1 Shape and symmetry features

We compute two geometrical descriptors that capture the overall body compactness and bilateral regularity of the specimen.

We seek the largest circle entirely contained within the binary mask, as a proxy for the core of the insect body (excluding legs, antennae and background protrusions). Denote by $D(x, y)$ the Euclidean distance transform of the mask (the distance from each pixel to the nearest background). Then the center (x^*, y^*) and radius r^* of the optimal circle solve

$$(x^*, y^*) \;=\; \arg\max_{(x,y)} D(x, y), \qquad r^* \;=\; D(x^*, y^*)\,.$$

In practice, we initialize the search at the mask centroid to ensure the circle lies within the main body, and we refine (x, y) by minimizing the negative distance via SciPy's Nelder-Mead solver [7]. Finally, we normalize the radius by the smaller side of the bounding box:

$$r_{\text{norm}} = \frac{r^*}{\min(\text{width}, \text{height})}.$$

We can observe the maximum and minimum value for the inscribed circle radius norm :

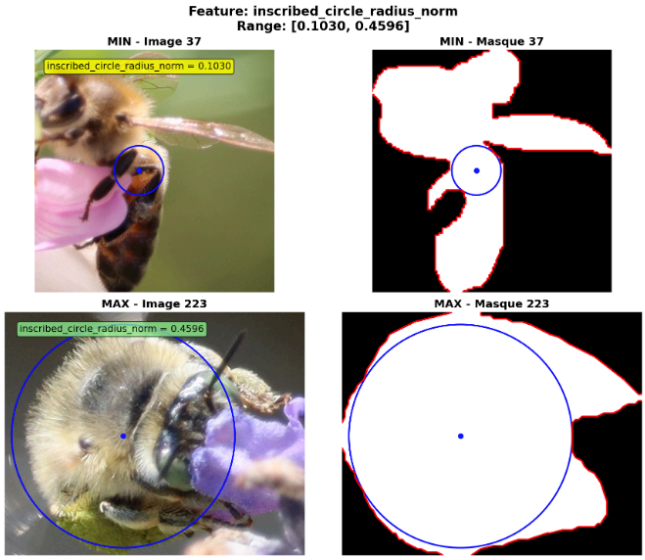


Figure 4: Maximum and minimum value for the inscribed circle radius norm

We complement our geometric description by seeking the optimal bilateral symmetry plane. To this end, three descriptors are computed:

- Minimum Symmetry Loss (L^*_{sym}): This is the lowest achieved value from our symmetry loss function. This function quantifies the dissimilarity (normalized sum of squared differences) between the grayscale object and its reflection across a candidate symmetry axis. The comparison is performed within a circular Region of Interest (ROI) determined by the object's largest inscribed circle. A lower loss value (closer to 0) indicates a better symmetry fit.
- Optimal Symmetry Angle (θ^*): This is the orientation (angle in degrees, typically $0° \leq \theta^* < 180°$) of the bilateral symmetry plane that minimizes the symmetry loss. It defines the principal axis of symmetry identified for the object.
- Symmetry Score (S_{sym}): This score provides a normalized measure of symmetry quality, ranging from 0 (indicating low symmetry with respect to the found plane) to 1 (indicating high or perfect symmetry). It is derived from the Minimum Symmetry

Loss (L^*_{sym}) as $S_{sym} = \max(0, 1 - L^*_{sym})$. A higher score indicates that the object is more closely bilaterally symmetric.

We can see here a example of results obtained :

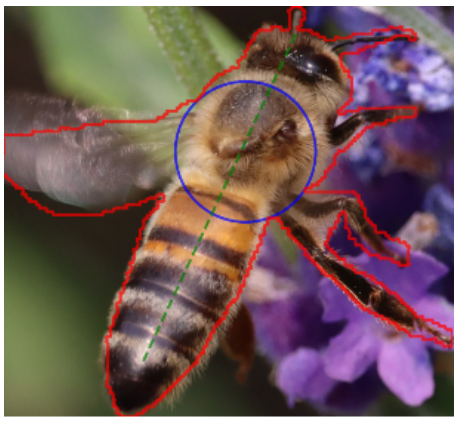


Figure 5: Combined view including the contour, the best plane of symmetry at $-25°$, and the optimal inscribed circle for image # 15

To evaluate the discriminative power of these two geometrical descriptors, we computed them for all 249 insects in our dataset and represented the results in a 2D feature space. Figure 6 below shows the distribution of insects with the normalized inscribed circle radius on the x-axis and the symmetry score on the y-axis, where each point is colored according to its bug type classification.

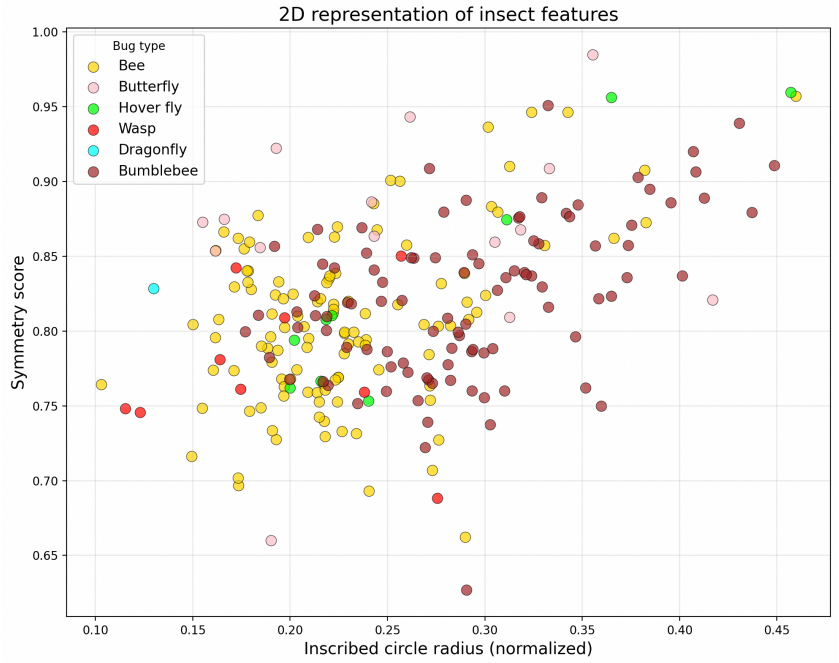


Figure 6: 2D representation of insect features showing inscribed circle radius (normalized) versus symmetry score. Each point represents one insect specimen, colored by bug type: Bee (yellow), Bumblebee (brown), Butterfly (pink), Hover fly (green), Wasp (red), and Dragonfly (cyan).

The visualization reveals that while these features capture meaningful morphological variations, their ability to distinguish between insect types is limited. The most problematic observation is the substantial overlap in the central region of the feature space, particularly between bees and bumblebees, which form an indistinguishable cluster approximately bounded by inscribed circle values of 0.20-0.35 and symmetry scores of 0.75-0.85.

This overlap encompasses roughly 60% of our dataset, making reliable classification challenging. Some separation is observable at the extremes: butterflies tend toward higher symmetry scores above 0.85, potentially reflecting their more regular wing patterns when viewed dorsally, while the single dragonfly specimen appears isolated with a lower inscribed circle value around 0.15, consistent with its elongated body shape. The symmetry score may partially capture differences in acquisition viewpoint, as insects photographed at oblique angles or in asymmetric poses would naturally exhibit lower bilateral symmetry scores.

However, the extensive interclass overlap and limited variance in both dimensions suggest that these two geometrical descriptors alone are insufficient for robust species classification, likely yielding accuracies below 50% in a practical classification system. This analysis indicates that while inscribed circle radius and symmetry score provide useful morphological characterization, additional features incorporating texture, color, or higher-order shape descriptors would be necessary to achieve satisfactory discrimination between insect species or acquisition views.

2.3 Added features

To further characterize the visual properties of the insects, we incorporated several advanced features beyond basic geometric and symmetry descriptors. These features aim to capture more nuanced visual properties and are grouped into morphological, textural (derived from the Gray Level Co-occurrence Matrix - GLCM), histogram-based, and color-based descriptors, as detailed below. This preliminary evaluation guided our selection process, ensuring that only the most informative features (presented below) were retained for modeling.

2.3.1 Morphological Features

These features describe the global shape and form of the insect's silhouette, extracted from its binary mask.

Hu moments [5] are a set of seven descriptors that remain invariant under translation, scaling, and rotation of an object. We extract all seven Hu moments from the insect contours, each capturing different geometric properties of the shape.

The moments are derived from central moments and provide a comprehensive shape signature. The first two moments (I_1 and I_2) characterize the overall dispersion and elongation of the shape, while higher-order moments (I_3 to I_7) capture more complex shape properties such as skewness and higher-order deformations. To stabilize their numerical range and enhance discriminative power, we apply a log-transform to each moment:

$$hu_i = -\operatorname{sign}(I_i) \log_{10} |I_i|$$

These moments effectively capture inter-class differences in insect morphology. For instance, the shape dispersion captured by the lower-order moments distinguishes between:

- Dragonflies with elongated, slender silhouettes (high dispersion)
- Bees and bumblebees with compact, rounded forms (low dispersion)
- Butterflies with intermediate dispersion due to broad wings but narrow bodies

The complete set of seven Hu moments provides a robust shape descriptor that helps our classifier discriminate between visually similar species like hoverflies and wasps, regardless of their position, size, or orientation in the image. Eccentricity [6] quantifies how much an

object deviates from a perfect circle. It is defined by fitting an ellipse to the binary mask and computing

$$\text{eccentricity} = \sqrt{1 - \frac{b^2}{a^2}},$$

where a and b are the ellipse's semi-major and semi-minor axes. Values range from 0 (circle) to 1 (line segment).

Because different insect groups have characteristic body and wing shapes, eccentricity becomes a strong discriminator. Below, we illustrate how eccentricity varies with mask shape, highlighting why this descriptor effectively separates elongated from rounded silhouettes.

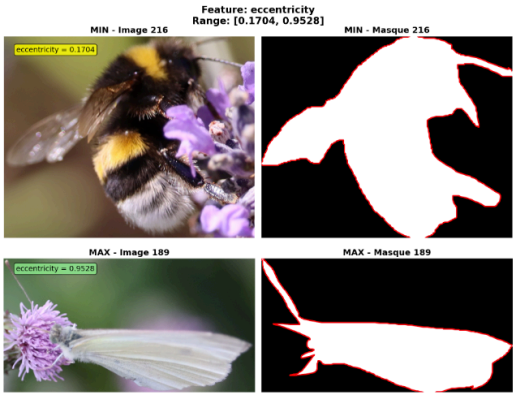


Figure 7: Maximum and minimum value for the eccentricity

2.3.2 Texture Features (GLCM)

Texture features are computed from the Gray Level Co-occurrence Matrix (GLCM). The GLCM characterizes the texture of an image by statistically sampling how often pairs of gray-levels occur at a given spatial offset. In our approach, the GLCM is computed on the grayscale representation of the insect, with pixel intensities first scaled down (to 16 levels) and then processed with a distance of 2 pixels at a 0-degree angle.

We extract four complementary texture descriptors from the GLCM:

Energy (also known as Angular Second Moment) measures the uniformity of the texture:

$$\text{Energy} = \sum_i \sum_j P(i,j)^2$$

This feature is particularly effective for identifying insects with uniform patterns like bee stripes or smooth butterfly wings.

Homogeneity quantifies the closeness of the distribution of elements in the GLCM to its diagonal:

$$\text{Homogeneity} = \sum_i \sum_j \frac{P(i,j)}{1 + |i-j|}$$

It captures the local uniformity in texture, helping distinguish smooth surfaces from rough ones.

Dissimilarity measures the local variations in the gray level co-occurrence matrix:

$$\text{Dissimilarity} = \sum_i \sum_j |i-j| \cdot P(i,j)$$

This feature is sensitive to texture variations, useful for differentiating complex patterns in dragonfly wings from simpler insect textures.

Correlation measures the linear dependency of gray levels on those of neighboring pixels:

$$\text{Correlation} = \sum_i \sum_j \frac{(i-\mu_i)(j-\mu_j)P(i,j)}{\sigma_i \sigma_j}$$

where μ and σ are the means and standard deviations of the row and column sums of P.

These four texture descriptors provide a comprehensive characterization of insect surface patterns, enabling effective discrimination between species with different textural properties in our classification pipeline.

2.3.3 Histogram-Based Features

From the grayscale pixel intensity distribution within the object's mask, we extract eight statistical descriptors that characterize the insect's appearance:

The mean (μ) and standard deviation (σ) capture the central tendency and spread of pixel intensities. The variance ($\sigma^2 = E[(X-\mu)^2]$) measures the dispersion of gray levels, with high values indicating contrasting patterns (like bee stripes) and low values suggesting uniform textures.

Skewness quantifies the asymmetry of the intensity distribution:

$$\text{Skewness} = \frac{E[(X-\mu)^3]}{\sigma^3}$$

Kurtosis measures the tail heaviness of the distribution:

$$\text{Kurtosis} = \frac{E[(X-\mu)^4]}{\sigma^4}$$

Entropy characterizes the randomness of pixel intensities:

$$\text{Entropy} = -\sum_i p_i \log_2(p_i)$$

where p_i is the probability of intensity level i.

Energy measures the sum of squared elements in the normalized histogram:

$$\text{Energy} = \sum_i p_i^2$$

Finally, the mode represents the most frequent intensity value in the histogram.

These features effectively capture different aspects of insect appearance. Figure 8 illustrates the discriminative power of variance, showing insects with maximum and minimum variance values from our database.

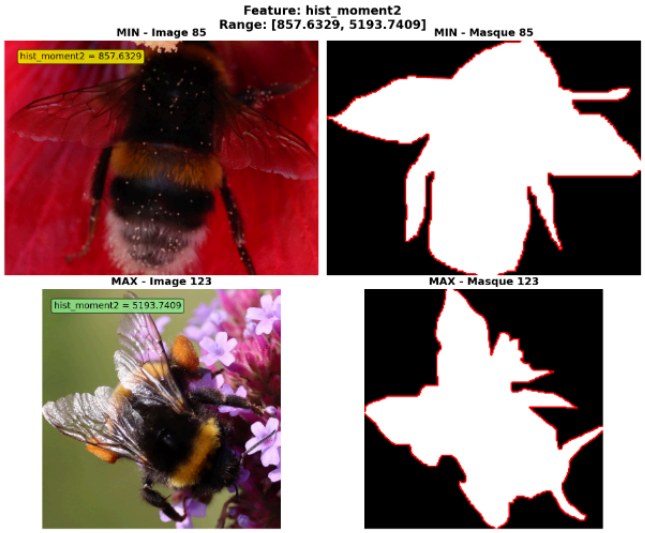


Figure 8: Maximum and minimum value for the variance

The clear difference in dynamic range between these examples demonstrates how histogram-based features can effectively distinguish between insects with different textural characteristics.

3 Data visualization

3.1 Mandatory visualization

Now that all the features have been extracted, we can begin by analyzing their distribution across the dataset, focusing on the differences between insect types and species. This will provide initial insights into how the features vary within and between classes. Following this, we will explore different projection techniques to better understand the relevance and contribution of each feature to the overall data structure.

We begin by examining the distribution of insect types within the dataset (with image # 154 remove).

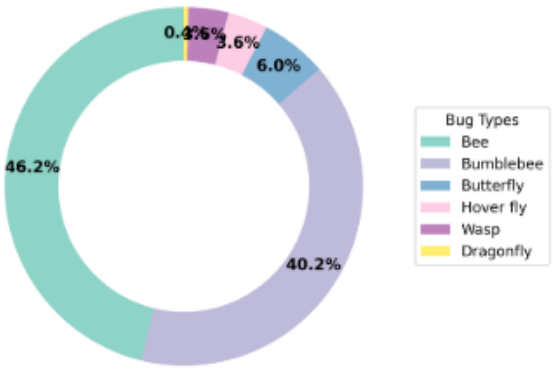


Figure 9: Bug Type Distribution

As shown in the donut chart, the majority of samples are bees (46.2%) and bumblebees (40.2%), together accounting for over 86% of the data. In contrast, other types such as Butterflies (6.0%), Hover flies (3.6%), Wasps (3.6%), and Dragonflies (0.2%) are significantly underrepresented. This strong class imbalance may influence model performance and should be kept in mind for later stages, particularly during training and evaluation.

Next, we analyze the distribution of species within each insect type.

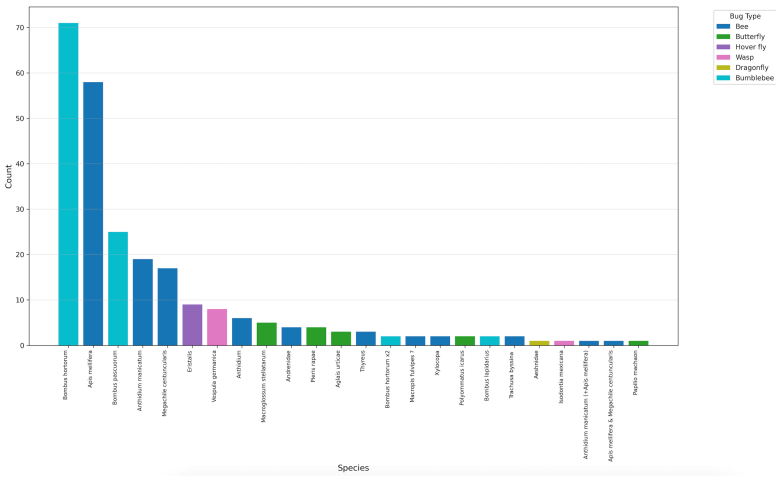
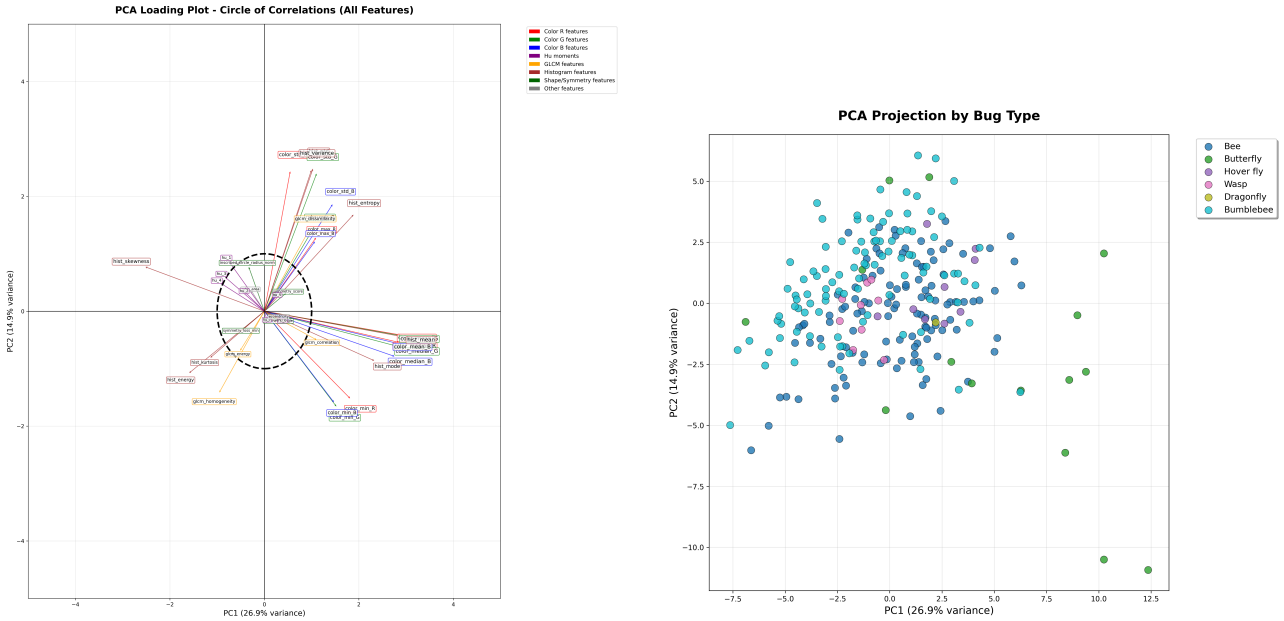


Figure 10: Distribution of species within each insect type.

As shown in the bar chart, the dataset is dominated by a few species such as Bombus hortorum (bumblebee), Apis mellifera, and Bombus pascuorum (bumblebee), which account for a significant proportion of the total samples. Most other species are represented by only

a few individuals, and some by just one or two samples. This reflects not only a strong class imbalance at the species level but also within each insect type. Such imbalance may limit the model's ability to generalize across less-represented species, especially when fine-grained classification is required.

After describing the class and species distributions, we now turn to a Principal Component Analysis (PCA) to evaluate how the extracted features structure the data. PCA serves two complementary purposes here: (i) to visualise the global organisation of samples in a reduced, two-dimensional space, and (ii) to identify which variables contribute most to the observed variance so that redundant or weak features can be removed.



(a) Circle of correlations

(b) PCA projection coloured by bug type

Figure 11: PCA of the 38 extracted features : (a) The loading plot shows how strongly each feature correlates with the first two principal components (PCs). Features whose arrows reach or exceed the dashed unit circle dominate the variance, those near the centre contribute little. (b) The score plot positions each sample in the PC_1-PC_2 plane and is coloured by insect type.

First we can observe that for explained variance, PC_1 and PC_2 together capture $\approx 42\%$ of the total variability (26.9 % and 14.9 %, respectively), giving a faithful though not exhaustive 2-D summary.

Moreover, for the dominant feature groups, RGB colour statistics (means, minima) exhibit the longest vectors along PC_1, indicating that colour drives the bulk of the variance. Histogram means/modes and key GLCM texture metrics (correlation, dissimilarity) also play a visible role, whereas most shape, Hu-moment, and basic geometry features lie inside the unit circle and thus explain little of the first two PCs.

In the score plot (Figure 11b) bees and bumblebees form a dense, overlapping core, confirming their visual similarity and the difficulty of separating them with colour alone. Minority classes sit on the outskirts, reinforcing the long-tail imbalance observed earlier.[1]

3.2 Added visualization

While PCA provides a useful linear approximation of the data structure, it may fail to capture complex, non-linear relationships between samples, especially when clusters are intertwined in high-dimensional space. To address this limitation, we now turn to two complementary non-linear dimensionality reduction techniques:t-distributed Stochastic Neighbor

Embedding (t-SNE) and lsomap. Both methods aim to preserve different aspects of the data geometry-local neighbourhoods in the case of t-SNE, and global geodesic distances for Isomap-offering deeper insight into the intrinsic data manifold and potentially revealing hidden groupings not visible through PCA.

First starting with t-SNE :

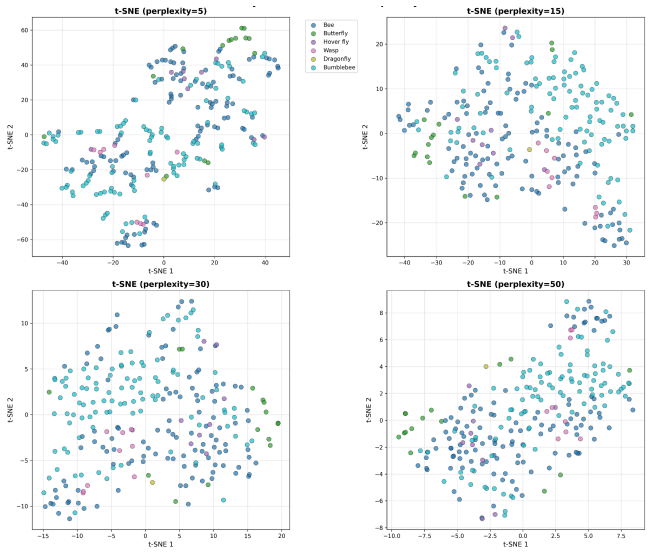


Figure 12: t-SNE projection with different perplexity values

Figure 12 shows the two-dimensional embeddings obtained with t-distributed Stochastic Neighbour Embedding for four perplexity values (5, 15, 30, and 50). Regardless of the neighbourhood size, two consistent patterns emerge. First, samples labelled as Bee and bumblebee (blue and cyan) form a dense, intertwined backbone occupying the centre of each map. The overlap confirms the strong visual similarity already hinted at by PCA and suggests that, with the current feature set, a classifier will struggle to draw a sharp boundary between these two dominant classes. Second, minority types Butterfly (green), Hover fly (purple), Wasp (pink), and Dragonfly (yellow) tend to populate the periphery of the manifold, but never merge into compact, isolated islands. This indicates that their visual signatures are moderately distinctive yet still partly embedded in the feature space spanned by the two main classes.

Increasing the perplexity smooths the embedding: at low perplexity (5) the manifold fragments into several filament-like chains, hinting at tiny, high-confidence neighbourhoods; from 15 upward the clouds merge, yielding a more global view that nevertheless preserves the broad dichotomy between central (bee-like) and peripheral (non-bee) regions. No perplexity value produces clearly separable clusters for all six insect types, meaning that nonlinear neighbours capture subtle structure but do not, on their own, resolve the class imbalance challenge.

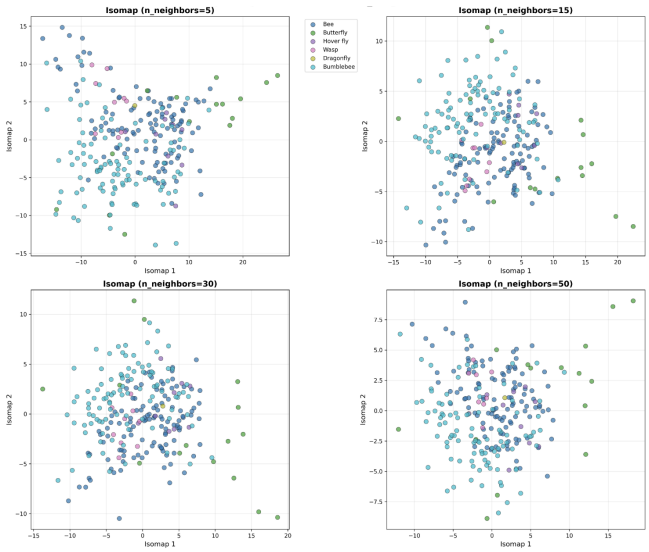


Figure 13: ISOMAP visualization with different neighbors values

Figure 13 reports the two leading Isomap coordinates obtained with four neighbourhood sizes (k=5, 15, 30, and 50). Unlike t-SNE, which emphasises local constellations, Isomap seeks to preserve global geodesic distances; the resulting maps therefore highlight the overall shape of the insect manifold rather than small, tight groups.

Across every k value the embeddings reveal a single, roughly circular core in which Bee and Bumblebee instances (blue, cyan) mix almost uniformly, mirroring their entanglement in the PCA and t-SNE plots. As k grows the core compacts: with k=5 elongated filaments remain, but from k=15 upward the structure stabilises into a dense disc, suggesting that the intrinsic dimensionality of the Bee-Bumblebee submanifold is low and well captured by global geodesic distances.

Minority classes occupy the perimeter yet do not crystallise into well-separated lobes. Butterflies (green) scatter on the positive Isomap_1 axis, while hover flies, wasps, and dragonflies appear as sparse border points whose positions fluctuate with k. This instability indicates that their inter-sample geodesic distances are not robustly defined in the current feature space.

Taken together, the Isomap results confirm that: the dominant Bee-like manifold is globally coherent but internally unsplit, making fine discrimination challenging.

4 Machine Learning and Deep Learning

4.1 Feature selection

Because the non-linear visualisations (PCA, t-SNE, Isomap) showed that our original feature space still entangles the six insect classes, we conducted a systematic feature-selection study to identify a compact, high-utility subset. The workflow was as follows:

1. All 38 extracted descriptors were pre-ranked by combining permutation importance with mutual information, computed on a stratified training sample.
2. We then applied recursive feature elimination (RFE) driven by an L2-regularised logistic regression (solver=liblinear, C=0.1, max_iter=300, random seed 42, one feature removed per step). Starting with the five highest-ranked variables, the next most informative feature was added iteratively. After each addition the model was refitted and its balanced accuracy was measured with repeated, stratified 5-fold cross-validation (shuffling enabled, same seed).
3. The mean accuracy and its standard deviation were plotted against the number of retained features, as shown in Figure 14.

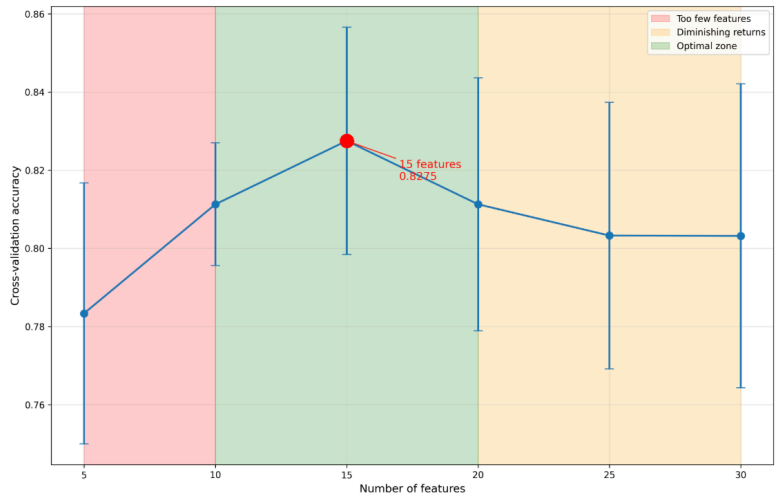


Figure 14: Model performance as a function of feature count.

The curve reveals three regimes: underfitting with too few features (red), a broad optimal window for 10-17 variables (green), and diminishing returns beyond 20 (orange). The maximum is reached at 15 descriptors, yielding a cross-validated balanced accuracy of 0.827 $\pm$ 0.030. We therefore lock the RFE mask at this point and carry forward only these fifteen variables, discarding the rest to lower dimensionality and curb overfitting in subsequent classification experiments.

4.2 Supervised methods

4.2.1 2 supervised methods that are neither deep learning nor ensemble learning

KNN After discarding the Dragonfly class (fewer than five images) the remaining 248 photographs were split stratified 80/20 ($n_{\text{train}} = 198$, $n_{\text{test}} = 50$). All 15 RFE-selected features were z-standardised. A `GridSearchCV` (stratified 5-fold) explored:

$$k \in \{3, 5, 7, 9, 11\}, \quad \text{weights} \in \{uniform, distance\}, \quad \text{metric} \in \{euclidean, cosine\}.$$

The best validation setting was $k = 5$, uniform weights, Euclidean distance, reaching a macro-F_1 of 0.607 ± 0.08 on the cross-validation folds. We obtain this classification report :

```
Classification Report:
              precision    recall  f1-score   support

         Bee       0.77      0.74      0.76        23
  Bumblebee        0.70      0.95      0.81        20
  Butterfly        0.00      0.00      0.00         3
  Hover fly        0.00      0.00      0.00         2
       Wasp        1.00      0.50      0.67         2

   accuracy                           0.74        50
  macro avg        0.50      0.44      0.45        50
weighted avg       0.68      0.74      0.70        50
```

Figure 15: Classification report metrics for KNN

On the held-out set the classifier obtained accuracy 0.74 and macro-F_1 0.45. KNN still struggles with minority categories, as illustrated by the confusion matrix in Figure 16: Bee and Bumblebee dominate the nearest-neighbour space, leaving Butterfly, Hover fly and Wasp almost unrecognised.

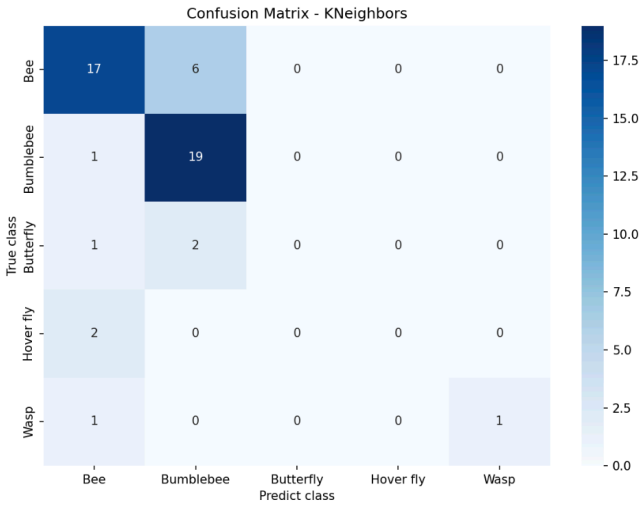


Figure 16: Confusion matrix for the KNN classifier.

With only fifteen dimensions the curse of dimensionality is not a major concern. Instead, performance is limited by the extreme class imbalance, and the intrinsic similarity between bees and bumblebees, whose dense cloud in feature space envelops the few minority samples.

SVM Using the same train/test split and scaling procedure, we fitted a C-SVC with class balancing (`class_weight=balanced`). The grid search covered

$$C \in \{10^{-3}, \ldots, 10^{2}\}\ (10 \text{ log steps}),\ \gamma \in \{scale, auto\},\ \text{kernel} \in \{rbf, linear\}.$$

The optimal model employed an rbf kernel with $C = 1$ and $\gamma = scale$, scoring 0.686 ± 0.113 macro-F_1 in cross-validation.

We obtained these results :

```
Classification Report:
              precision    recall  f1-score   support

         Bee       0.86      0.78      0.82        23
   Bumblebee       0.76      0.95      0.84        20
   Butterfly       1.00      0.67      0.80         3
   Hover fly       0.00      0.00      0.00         2
        Wasp       1.00      1.00      1.00         2

    accuracy                           0.82        50
   macro avg       0.72      0.68      0.69        50
weighted avg       0.80      0.82      0.80        50
```

Figure 17: Classification report for SVM

On the test set the SVM improves to accuracy 0.82 and macro-F_1 0.69. Figure 18 shows far fewer false positives for bees/bumblebees and the only perfect detection of the scarce wasps.

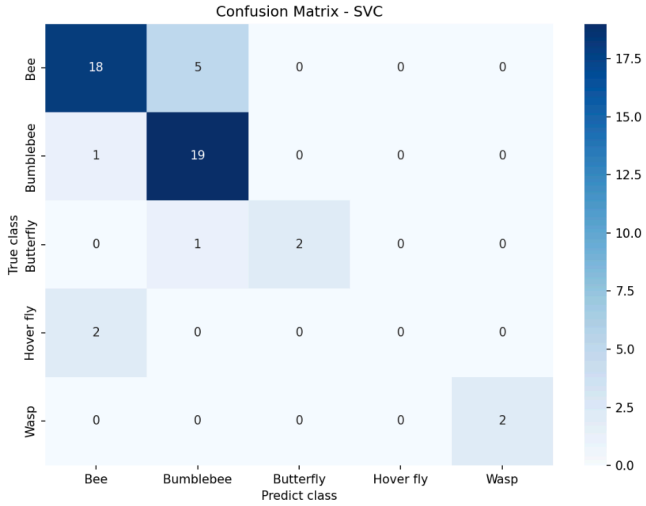


Figure 18: Confusion matrix for the SVM classifier.

The kernel SVM benefits from the same modest dimensionality yet can carve non-linear decision boundaries that separate minority classes more sharply than KNN. Nevertheless, misclassifications of Hover flies and the tiny Butterfly cluster persist, confirming that additional data-or feature engineering tailored to fine-grained texture and wing-shape cues-is required for perfectly balanced performance. Under identical preprocessing and evaluation protocols the balanced SVM outperforms KNN by 19 % macro-F_1 and 8 % accuracy, mainly by rescuing minority classes such as Wasp and reducing Bee-Bumblebee cross-talk. The improvement is consistent across CV folds, making the SVM the preferred classical baseline.

4.2.2 1 supervised ensemble learning method

RF As the required ensemble-learning baseline, we trained a balanced Random-Forest (RF) on the same 80/20 split and 15-feature input used by KNN and SVM. Because tree models are scale-invariant, no standardisation was applied. A `GridSearchCV` (stratified 5-fold, macro-F_1 scoring) scanned

$$\begin{aligned} n_{\text{trees}} &\in \{50, 100, 200, 300\}, \\ max\_depth &\in \{\text{None}, 10, 20, 30, 50\}, \\ min\_samples\_leaf &\in \{1, 2, 4\}, \\ max\_features &\in \{\sqrt{}, \log_2, \text{None}\}. \end{aligned}$$

The 5-fold procedure proved crucial: with a single train/test split this instability would have remained hidden and the apparently decent accuracy of a lucky split could have been mistaken for robust performance.

Classification Report:

	precision	recall	f1-score	support
Bee	0.79	0.83	0.81	23
Bumblebee	0.77	1.00	0.87	20
Butterfly	0.00	0.00	0.00	3
Hover fly	0.00	0.00	0.00	2
Wasp	0.00	0.00	0.00	2
accuracy			0.78	50
macro avg	0.31	0.37	0.34	50
weighted avg	0.67	0.78	0.72	50

Figure 19: Classification report (test split) for the Random-Forest.

On the test partition the ensemble reached only macro-$F_1 = 0.34$ at a seemingly strong accuracy 0.78[1] (Figure 20). The high accuracy is deceptive: all tail classes receive zero recall.

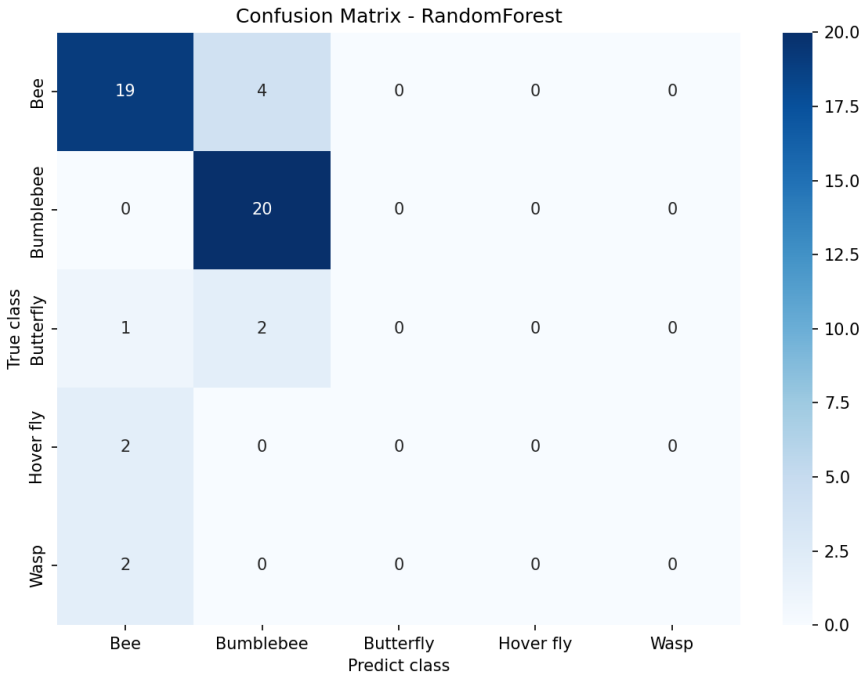


Figure 20: Confusion matrix for the Random-Forest classifier.

- All bumblebees (20/20) and most bees (19/23) are correctly labelled, accounting for the 0.78 accuracy.
- Every Butterfly, Hover-fly and Wasp is absorbed by a majority class, yielding $F_1 = 0$ for three categories.
- Bee $\rightarrow$ Bumblebee confusion falls (4 vs. 6 with KNN), but at the expense of ignoring the tail classes entirely.

Despite class-balancing weights, the forest overfits to the dense Bee/Bumblebee manifold-rare-class votes are drowned in correlated trees. Enlarging the forest or adding cost-sensitive weighting could help, but the key lesson is that ensemble size and balanced priors alone do not ensure robustness when minority support drops to two samples.

[1] Exactly 19/23 bees and all 20 bumblebees are recovered, inflating accuracy despite the collapse on minority classes.

While Random-Forest secures the second-best raw accuracy, its macro-F_1 lags behind both SVM and KNN owing to total failure on three classes, and the k-fold analysis warns that this gap is structural rather than an artefact of a particular data split.

4.2.3 At least 1 supervised method studied in the optimization & AI

Logistic Regression A linear, L_2-regularised multinomial Logistic Regression was trained on the same stratified 80/20 split and z-normalised 15-feature input used for the other classical models. Hyper-parameters were optimised with a `GridSearchCV` (stratified 5-fold, macro-F_1 scoring):

$$C \in \{10^{-3}, \ldots, 10^{3}\},\ \textit{solver} \in \{\text{lbfgs}, \text{liblinear}, \text{saga}\},\ \textit{penalty} = L_2.$$

The grid selected liblinear with $C = 1$, yielding a cross-validated macro-F_1 of 0.63 ± 0.10.

Classification Report:

	precision	recall	f1-score	support
Bee	0.71	0.74	0.72	23
Bumblebee	0.74	0.85	0.79	20
Butterfly	0.50	0.33	0.40	3
Hover fly	0.00	0.00	0.00	2
Wasp	0.00	0.00	0.00	2
accuracy			0.70	50
macro avg	0.39	0.38	0.38	50
weighted avg	0.65	0.70	0.67	50

Figure 21: Classification report for Logistic Regression (test split).

On the test set LR attains accuracy 0.70 and macro-F_1 0.38. The detailed confusion matrix is shown in Figure 22.

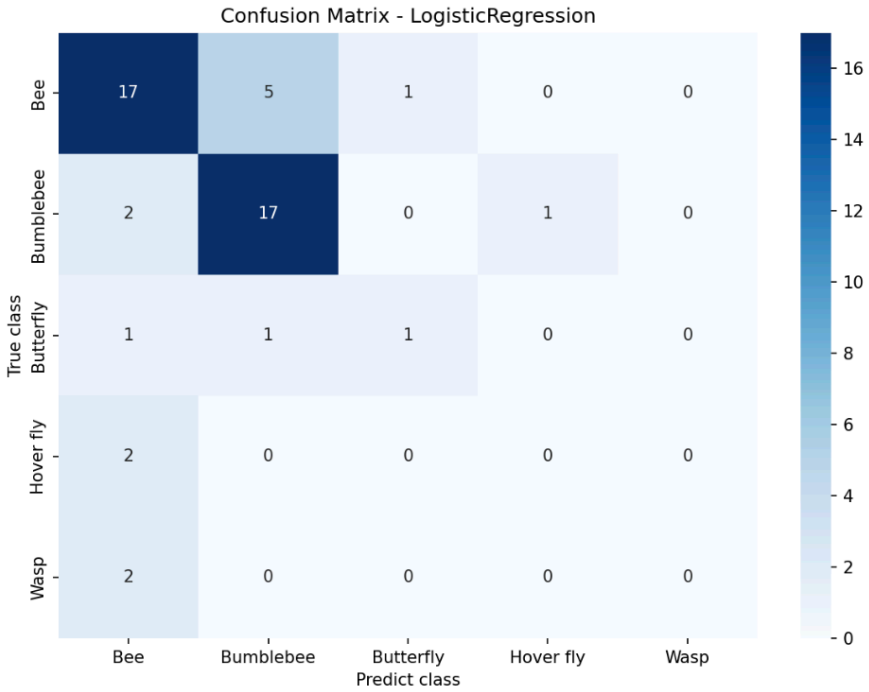


Figure 22: Confusion matrix for the Logistic Regression classifier.

- Bee vs. Bumblebee: 5 bees and 2 bumblebees are mis-swapped, confirming that a linear boundary is still unable to disentangle their feature overlap.

- Butterfly gains one correct hit (1/3) unlike RF or KNN, yet its recall remains poor, and Hover-fly and Wasp remain entirely undetected.
- The gap between CV macro-F_1 (0.63) and test macro-F_1 (0.38) highlights fold-to-fold variance, with so few minority samples, even a single misplacement can swing the score by ± 0.1.

Despite balanced class weights and tuned regularisation, the linear LR model captures mostly the coarse Bee/Bumblebee separation, achieving a macro-F_1 slightly above Random-Forest (0.34) yet well below SVM (0.69). Its limited capacity confirms that non-linear decision surfaces, or richer feature engineering, are required to rescue the rare insect categories.

4.3 Unsupervised methods

Following the supervised classification analysis, we now explore unsupervised clustering approaches to investigate the natural groupings within our insect dataset. Unlike supervised methods that leverage labeled data, clustering algorithms seek inherent structure in the feature space without prior knowledge of bug types. This analysis serves two purposes: (i) validating whether the engineered features capture meaningful inter-class differences, and (ii) assessing how well unsupervised groupings align with the true taxonomic categories (insect categories).

We evaluate two complementary clustering algorithms: K-Means for its computational efficiency and interpretability, and hierarchical clustering for its ability to reveal nested taxonomic relationships. Both methods operate on the same 15 RFE-selected features used in supervised learning, with $k = 5$ clusters corresponding to the number of valid bug types after filtering classes with fewer than five samples (Dragonfly excluded).

4.3.1 K-Means Clustering

We applied K-Means clustering to the z-standardised 15-dimensional feature vectors from all 248 valid images. The algorithm minimises within-cluster sum of squares (WCSS) through iterative centroid updates:

$$\min_{\mathbf{C}} \sum_{i=1}^{n} \min_{j=1}^{k} \|\mathbf{x}_i - \mathbf{c}_j\|^2$$

where $\mathbf{c}_j$ represents the j-th cluster centroid. Key parameters included:

- $k = 5$ clusters, matching the number of remaining insect classes
- `n_init=50` random initialisations to mitigate local minima
- `random_state=42` for reproducibility
- Euclidean distance metric in the standardised feature space

The choice of $k = 5$ was motivated by domain knowledge rather than purely data-driven methods (e.g., elbow method), as we aim to assess how well unsupervised clusters correspond to known taxonomic categories.

The K-Means algorithm converged to a stable partition with the following performance metrics:

- Silhouette Score: 0.167 (range: [-1, 1])
- Davies-Bouldin Index: 1.565 (lower is better)
- Adjusted Rand Index: 0.169 (vs. true labels)

Figure 23 visualises the clusters projected onto the first two principal components, which explain approximately 35% of the total variance. The relatively low silhouette score indicates substantial overlap between clusters, while the modest ARI suggests limited correspondence with true bug types.

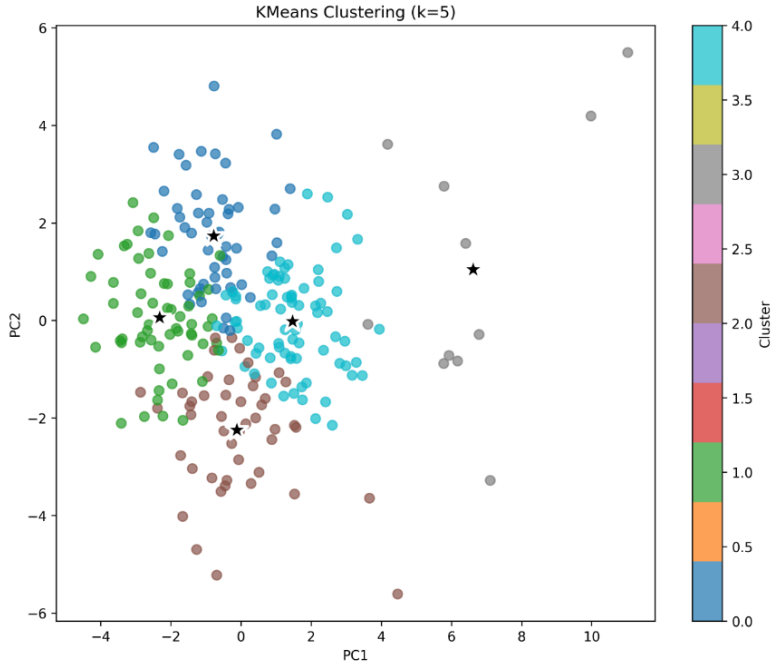


Figure 23: K-Means clustering results projected onto the first two principal components. Cluster centers are marked with black stars.

The poor clustering metrics reveal fundamental challenges in unsupervised insect categorisation.

The silhouette score of 0.167 indicates that many points lie near cluster boundaries, suggesting continuous rather than discrete distributions in feature space. This aligns with our supervised findings where bees and bumblebees showed substantial feature overlap.

Despite no explicit class weighting in K-Means, the algorithm naturally gravitates toward dense regions dominated by majority classes (bees/bumblebees), potentially merging minority classes into these larger clusters.

The ARI of 0.169 confirms that unsupervised clusters poorly match true bug types. This suggests that either (i) the 15 selected features do not capture the visual cues humans use for taxonomic classification, or (ii) insect categories form a continuum rather than distinct clusters in our feature space.

The Davies-Bouldin Index of 1.565 further corroborates poor cluster separation, with high intra-cluster variance relative to inter-cluster distances. This metric is particularly sensitive to the spherical cluster assumption of K-Means, which may be violated for elongated distributions like those observed in the PCA projection.

These results underscore that while our engineered features suffice for supervised classification (SVM achieving 0.69 macro-F_1), they do not naturally partition into taxonomically meaningful groups without labeled guidance. This gap between supervised and unsupervised performance suggests that discriminative boundaries learned by classifiers exploit subtle feature combinations invisible to distance-based clustering.

4.3.2 Hierarchical Clustering

As a complementary approach to K-Means, we employed Agglomerative Hierarchical Clustering with Ward's linkage criterion. This bottom-up algorithm recursively merges the clos-

est pairs of clusters, minimising within-cluster variance at each step:

$$d_{\text{Ward}}(C_i, C_j) = \frac{n_i n_j}{n_i + n_j} \|\bar{x}_i - \bar{x}_j\|^2$$

where n_i, n_j are cluster sizes and $\bar{x}_i$, $\bar{x}_j$ are cluster centroids. The configuration employed Ward linkage for minimal within-cluster variance, cut the dendrogram at $k = 5$ clusters to match the number of valid bug types, and operated on Euclidean distances in the standardised feature space. Unlike K-Means, this deterministic algorithm requires no random initialisation. Ward's method was chosen for its tendency to produce compact, spherical clusters of similar size, which aligns with the assumption of distinct insect categories.

The hierarchical clustering yielded a silhouette score of 0.138, a Davies-Bouldin Index of 1.787, and an Adjusted Rand Index of 0.114 when compared to true labels. These metrics are marginally lower than those obtained with K-Means, suggesting even weaker cluster cohesion and taxonomic correspondence. Figure 24 shows the hierarchical partition in PCA space, revealing a different cluster structure compared to K-Means.

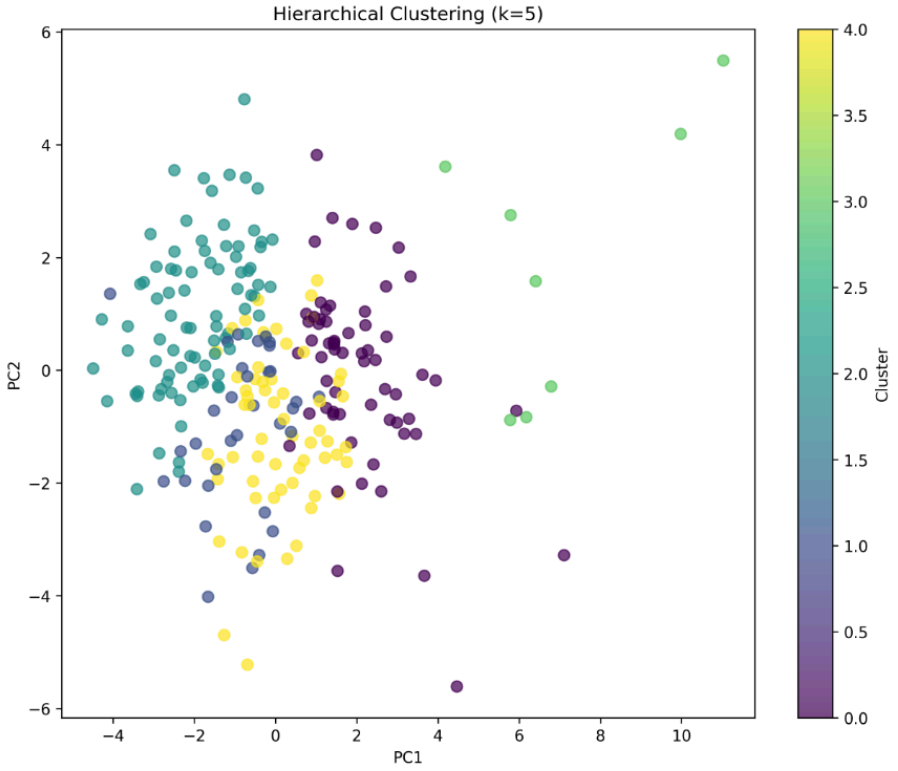


Figure 24: Hierarchical clustering results (Ward linkage) projected onto principal components. The colour gradient indicates cluster assignments.

The hierarchical approach exhibits similar limitations to K-Means but with distinct characteristics. Unlike K-Means' tendency toward spherical clusters, hierarchical clustering with Ward linkage can capture more elongated structures. However, the lower silhouette score (0.138 vs. 0.167) suggests this flexibility did not improve cluster quality in our feature space. The higher Davies-Bouldin Index (1.787 vs. 1.565) indicates worse cluster separation, possibly due to the greedy nature of agglomerative clustering which cannot recover from early suboptimal merges. Most notably, the ARI of 0.114 represents even poorer correspondence with true bug types than K-Means, suggesting that the hierarchical structure imposed by Ward linkage does not reflect natural taxonomic relationships in our feature representation.

4.3.3 Synthesis of Unsupervised Results

Both clustering algorithms produce notably similar results, with silhouette scores below 0.17 and ARI values indicating minimal correspondence with true taxonomic categories. This consistency across different algorithmic approaches suggests that our 15-dimensional feature space does not naturally partition into distinct insect classes.

The performance gap between supervised and unsupervised methods is particularly revealing. While SVM achieved 0.69 macro-F_1 with labeled data, both clustering approaches fail to recover meaningful taxonomic groupings (ARI < 0.17). This indicates that discriminative boundaries between insect types exist but require supervision to identify-they are not apparent through distance-based clustering alone.

These findings suggest that our engineered features, though effective for supervised classification, do not inherently encode the taxonomic structure of the dataset. Successful insect classification in this feature space thus depends on labeled training examples rather than unsupervised discovery.

4.4 Bonus : Deep learning

To evaluate the potential of deep learning methods, we implemented a convolutional architecture based on MobileNetV2. [4]

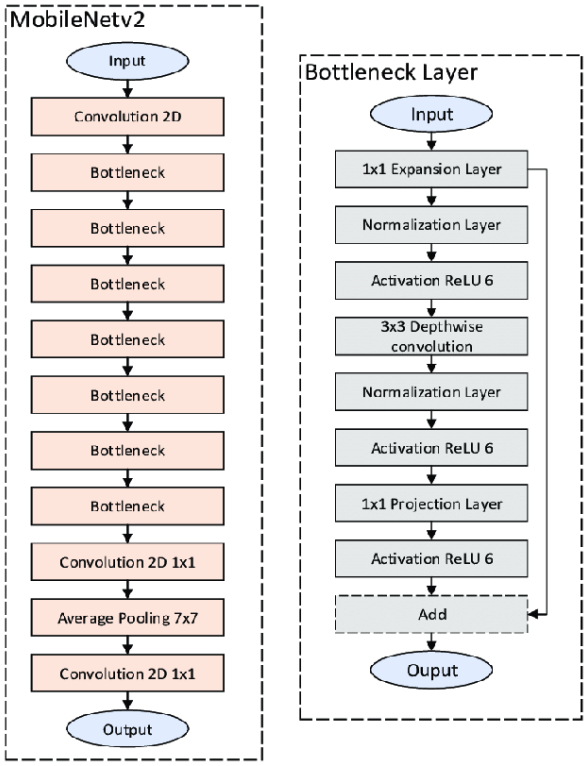


Figure 25: MobileNet-v2 Architecture [2]

It's lightweight pretrained CNN originally trained on ImageNet [3]. The model was adapted for multiclass insect classification using masked RGB images.

Each image was resized to 224×224 pixels and normalized to $[0, 1]$. A binary segmentation mask was applied to isolate the insect from the background. Class labels were encoded using a `LabelEncoder`, and transformed into one-hot vectors for multiclass softmax classification.

The model consists of a frozen MobileNetV2 base, followed by global average pooling, one dense layer (128-512 units), dropout (rate between 0.2 and 0.5), and a softmax output.

Hyperparameters were tuned via `Keras Tuner` (`RandomSearch`) with 10 trials and validation accuracy as the objective. The optimizer (Adam or RMSProp) and learning rate (10^{-4} to 10^{-2}, log scale) were also included in the search space.

The selected architecture was trained for 50 epochs with the base frozen. Then, the last 30 layers of MobileNetV2 were unfrozen for fine-tuning using a reduced learning rate (divided by 10). The best model (based on validation accuracy) was saved using `ModelCheckpoint`.

On the validation set, the best model achieved an overall accuracy of 0.88 and a macro-F_1 score of 0.69.

	precision	recall	f1-score	support
Bee	0.84	0.91	0.88	23
Bumblebee	0.90	0.95	0.93	20
Butterfly	1.00	1.00	1.00	3
Hover fly	0.00	0.00	0.00	2
Wasp	1.00	0.50	0.67	2
accuracy			0.88	50
macro avg	0.75	0.67	0.69	50
weighted avg	0.85	0.88	0.86	50

Figure 26: Classification report for the deep learning model.

- Bee and Bumblebee are well separated, with F_1 scores of 0.88 and 0.93 respectively, despite their high feature overlap.
- Butterfly achieves perfect classification ($F_1 = 1.00$), even with very limited samples.
- Hover fly is entirely missed ($F_1 = 0.00$), and Wasp is partially detected (recall = 0.50, precision = 1.00).

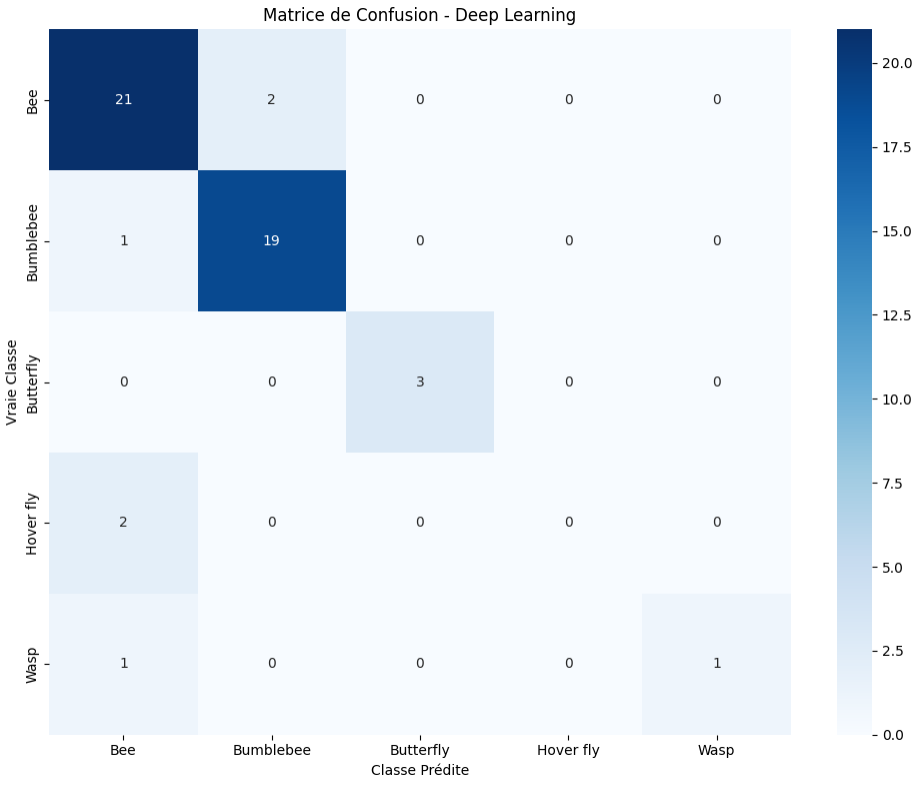


Figure 27: Confusion matrix for the deep learning classifier.

As shown in Figure 27, the model effectively classifies Bee and Bumblebee instances,

while minority classes such as Hover fly and Wasp are overwhelmed by the majority class distributions, highlighting the limitations of class imbalance.

Comparison to Classical Methods

- The deep model outperforms all classical baselines in terms of accuracy (0.88 vs. 0.82 for SVM, 0.74 for KNN, 0.78 for RF).
- In terms of macro-F_1, it ties with the best classical model (SVM: 0.69), but remains significantly ahead of KNN (0.45), Random Forest (0.34), and Logistic Regression (0.38).
- Unlike Random Forest or KNN, it successfully classifies Wasp and Butterfly, although Hover fly remains undetected across all models.

In conclusion, despite the limited number of training examples for certain classes, the deep learning model based on MobileNetV2, augmented with masking and fine-tuning, delivers the highest accuracy and ties for best macro-F_1. It captures fine-grained featuresthat classical models cannot easily leverage. This makes it the most promising candidate for insect classification.

5 Bonus: Species-Level Classification

Following the analysis of "bug type" classification, an additional experiment was conducted to assess the feasibility of predicting the more granular "species" column using the same deep learning framework. The objective was to determine if the MobileNetV2-based architecture could distinguish between individual insect species, a considerably more challenging task due to a higher number of classes and potentially greater inter-class similarity with fewer distinguishing features.

The methodology mirrored that of the "bug type" classification. The process began with loading the dataset, which initially consisted of 250 images. One image was subsequently ignored due to a missing mask file, resulting in 248 usable images for species classification. The "species" column from the dataset was targeted as the label for training. This initial dataset comprised 25 distinct species labels. However, a filtering step was applied to remove species with fewer than two samples to ensure a minimum level of representation for each class during training and validation. This filtering led to the removal of 5 species ('Aeshnidae', 'Anthidium manicatum (+Apis mellifera)', 'Apis mellifera & Megachile centuncularis', 'Isodontia mexicana', 'Papilio machaon'), which each had only one sample. Consequently, the final dataset used for training contained 244 images distributed across 20 species. This curated dataset was then split into training (195 images) and validation (49 images) sets. A new label encoding process was performed on these 20 species, and the MobileNetV2 architecture, with its pre-trained base, global average pooling, dense layer, dropout, and softmax output, was employed. Hyperparameter tuning using Keras Tuner, followed by initial training with a frozen base and subsequent fine-tuning of the top 30 layers of MobileNetV2, was performed as previously described.

Upon training and validation, the model achieved a validation accuracy of 0.6735 and a validation loss of 1.2730 for the species classification task. The detailed performance metrics for each species are presented in the classification report (Figure 28).

	precision	recall	f1-score	support
Aglais urticae	1.00	1.00	1.00	1
Andrenidae	0.00	0.00	0.00	1
Anthidium	0.00	0.00	0.00	1
Anthidium manicatum	1.00	0.50	0.67	4
Apis mellifera	0.43	0.83	0.57	12
Bombus hortorum	0.93	1.00	0.97	14
Bombus pascuorum	0.67	0.80	0.73	5
Eristalis	0.00	0.00	0.00	2
Macroglossum stellatarum	1.00	1.00	1.00	1
Megachile centuncularis	0.00	0.00	0.00	4
Pieris rapae	1.00	1.00	1.00	1
Thyreus	0.00	0.00	0.00	1
Vespula germanica	0.00	0.00	0.00	2
accuracy			0.67	49
macro avg	0.46	0.47	0.46	49
weighted avg	0.58	0.67	0.61	49

Figure 28: Classification report for the deep learning model on species classification.

The macro average F_1-score achieved was 0.46, and the weighted average F_1-score was 0.61. The confusion matrix (Figure 29) provides a visual representation of the classification performance across the various species.

Figure 29: Confusion matrix for the deep learning classifier on species.

Observing the classification report and confusion matrix, it is evident that the model performs well for certain species with sufficient representation and distinct features, such as Bombus hortorum ($F_1 = 0.97$). Species like Aglais urticae, Macroglossum stellatarum, and Pieris rapae also show perfect scores, though this is based on very limited samples (support of 1 each) and thus may not generalize. However, a significant number of species, particularly those with few samples like Andrenidae, Anthidium, Eristalis, Megachile centuncularis, Thyreus, and Vespula germanica, were entirely misclassified ($F_1 = 0.00$). Apis mellifera achieved a reasonable recall (0.83) but lower precision (0.43), indicating it was often predicted correctly when present, but other species were frequently misclassified as Apis mellifera. The confusion matrix reveals that many misclassifications occur towards the more populous classes like Apis mellifera and Bombus hortorum.

Comparing these results to the "bug type" classification, there is a noticeable decrease in performance. The accuracy dropped from 0.88 for "bug type" to 0.67 for "species," and the macro F_1-score decreased from 0.69 to 0.46. This reduction in performance is expected. The species-level task, even with 20 classes post-filtering (of which 13 appeared in the validation set shown in the report), involves a larger set of categories compared to 6 "bug types". Many of these species remain highly underrepresented in the dataset. Furthermore, distinguishing between species within the same broader "bug type" (e.g., different species of bees) is inherently more difficult due to subtle visual differences. The model's struggle highlights the significant impact of class imbalance and the increased complexity of fine-grained classification with limited data per class. While the MobileNetV2 architecture demonstrated promise, its effectiveness for species-level identification is severely hampered by these data limitations, leading to substantially lower and less reliable classification results than for broader bug categories. **Thus, for the predictions that we will have to make on the Test dataset, we will use the MobileNetV2 deep learning model that we have already developed.**

6 Bonus : Bug Type Classification with Raw Images

Further exploring the impact of preprocessing, an additional experiment focused on classifying the original "bug type" categories using raw images, thereby omitting the segmentation mask application. The aim was to assess how the MobileNetV2-based model would perform when tasked with learning directly from images that include the natural background, and to compare this with the performance achieved using masked images as detailed in the main body of the report.

The methodology remained consistent with the previous deep learning experiments in terms of model architecture, hyperparameter tuning strategy with Keras Tuner, and training procedure (initial training with a frozen base followed by fine-tuning). The key difference was in the data preprocessing pipeline, where images were directly resized and normalized without any mask-based foreground extraction. This meant the model was exposed to the entire visual content of each image.

For this "bug type" classification task on raw images, the model achieved a validation accuracy of 0.8000 and a validation loss of 0.7801. The detailed per-class performance is presented in the classification report (Figure 30).

	precision	recall	f1-score	support
Bee	0.85	0.74	0.79	23
Bumblebee	0.78	0.90	0.84	20
Butterfly	0.75	1.00	0.86	3
Hover fly	0.50	0.50	0.50	2
Wasp	1.00	0.50	0.67	2
accuracy			0.80	50
macro avg	0.78	0.73	0.73	50
weighted avg	0.81	0.80	0.80	50

Figure 30: Classification report for the deep learning model on "bug type" classification using raw images.

The macro average F_1-score was 0.73, and the weighted average F_1-score was 0.80. The confusion matrix (Figure 31) illustrates the model's predictions across the different bug types.

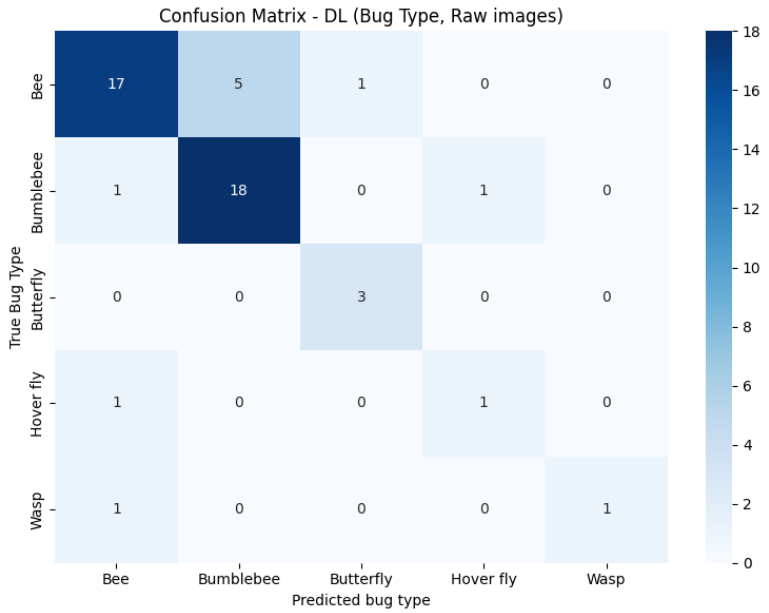


Figure 31: Confusion matrix for the deep learning classifier on "bug type" using raw images.

The results indicate a strong performance for "Bumblebee" ($F_1 = 0.84$) and "Butterfly" ($F_1 = 0.86$), with "Bee" also achieving a good F_1-score of 0.79. The "Butterfly" class, despite its small support (3 samples), was perfectly recalled. "Hover fly" ($F_1 = 0.50$) and "Wasp" ($F_1 = 0.67$) had lower scores, likely due to their very limited number of samples (2 each). The confusion matrix shows that "Bee" instances were sometimes misclassified as "Bumblebee" (5 instances).

Interestingly, when comparing these results to the initial "bug type" classification performed with masked images (which achieved an accuracy of 0.88 and a macro F_1-score of 0.69, as mentioned previously), the model trained on raw images yielded a slightly lower accuracy (0.80 vs. 0.88). However, the macro F_1-score for the raw image model was higher (0.73 vs. 0.69). A higher macro F_1-score suggests a potentially more balanced performance across classes, especially benefiting minority classes, as it averages the F_1-scores per class without considering class support.

This counter-intuitive improvement in macro F_1-score despite a drop in overall accuracy warrants consideration. Several factors might contribute to this. First, the quality of the segmentation masks used in the initial experiment could play a role. Imperfect masks might inadvertently obscure useful features or introduce noise, whereas the raw image allows the model to learn from all available pixels. Second, the background in the raw images, while potentially distracting, might also contain subtle contextual cues that the convolutional network learns to leverage or effectively ignore, leading to the development of more robust features. MobileNetV2, having been pre-trained on diverse ImageNet images with complex backgrounds, is inherently capable of handling such contextual information. Third, training on more varied raw images might act as a form of natural data augmentation, potentially improving generalization for some classes compared to cleaner, but possibly overly uniform, masked images. It is plausible that for certain classes, particularly minority ones, the additional context or the model's ability to learn to segment implicitly leads to a better balance of precision and recall, thereby boosting their individual F_1-scores and consequently the macro average. While the highest accuracy was achieved with masks, the improved macro F_1-score with raw images suggests that for tasks where balanced performance across all classes is critical, training on raw images could be a viable or even preferable strategy, provided the model can effectively learn to distinguish object from background.

7 Conclusion

This study evaluated various insect classification methods, from traditional feature engineering to deep learning, to assess their potential and limitations for biodiversity monitoring. The analysis covered broader "bug type" identification, finer-grained species-level classification, and the impact of image preprocessing (masked vs. raw images).

Among classical methods using an optimized set of engineered features, SVM with an RBF kernel emerged as the top performer (82% accuracy, 0.69 macro-F_1). However, unsupervised clustering indicated limited natural class separation within this engineered feature space (ARI < 0.17), suggesting inherent difficulties in distinguishing classes based solely on these features.

Deep learning models based on MobileNetV2 showed considerable promise. For "bug type" classification, using masked images achieved the highest overall accuracy (88%) and a strong macro-F_1 score (0.69). Intriguingly, an experiment using raw images for the same task yielded a slightly lower accuracy (80%) but an improved macro-F_1 score (0.73). This suggests that while image masking can guide models to higher accuracy, processing raw images might foster a better balance in performance across classes, possibly by allowing the model to learn more robust features or leverage subtle contextual information. However, the deep learning approach faced significant hurdles when applied to finer-grained species-level classification, where performance dropped considerably (accuracy $\approx$ 67%, macro-F_1 $\approx$ 0.46), primarily due to the increased number of classes and severe data imbalance. This issue of class imbalance proved to be a critical limiting factor across all deep learning experiments, particularly affecting underrepresented categories.

For the predictions that we will have to make on the Test dataset, we will use the MobileNetV2 deep learning model that we have already developed.

References

[1] Zihan Chen et al. *Figure 1: A comparison between the balanced, imbalanced, and long-tailed data distributions over a dataset with 7 classes*. https://www.researchgate.net/figure/A-comparison-between-the-balanced-imbalanced-and-long-tailed-data-distributions-over-a_fig1_361657075. From the preprint: Towards Federated Long-Tailed Learning. 2022.

[2] Revella Eshaya and Maiwan Abdulrazzaq. "HANDWRITTEN CHARACTER RECOGNITION IN ASSYRIAN LANGUAGE USING CONVOLUTIONAL NEURAL NETWORK". In: *Science Journal of University of Zakho* 12 (Mar. 2024), pp. 105–115. DOI: 10.25271/sjuoz.2024.12.1.1189.

[3] ImageNet Team. *ImageNet*. https://www.image-net.org/. Large-scale image database organized according to the WordNet hierarchy.

[4] Keras Team. *MobileNet, MobileNetV2, and MobileNetV3 - Keras*. https://keras.io/api/applications/mobilenet/. Keras Applications Documentation.

[5] Satya Mallick and Krutika Bapat. *Shape Matching using Hu Moments (C++/Python)*. Consulté le 30 mai 2025. 2018. URL: https://learnopencv.com/shape-matching-using-hu-moments-c-python/.

[6] scikit-image contributors. *Measure region properties*. 2025. URL: https://scikit-image.org/docs/0.25.x/auto_examples/segmentation/plot_regionprops.html.

[7] SciPy Community. *scipy.optimize.minimize(method='Nelder-Mead') SciPy v1.15.3 Manual*. https://docs.scipy.org/doc/scipy/reference/optimize.minimize-neldermead.html. Documentation officielle de SciPy.