**Ahmed Samir 8120**

**Youssef Awad 8179**

**Amr Samir 8211**

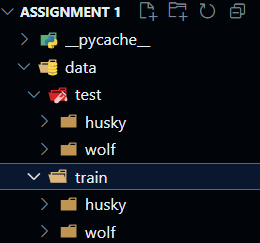
**Introduction**

The objective is to build a binary image classification system to distinguish between husky and wolf images. The tasks include data loading, splitting, preprocessing (resizing, normalization, augmentation), model creation with hyperparameter tuning, and evaluation on a test set. Bonus tasks involve using Grad-CAM and addressing data leakage.

**2. Dataset Description**

The dataset contains images of huskies and wolves, divided into two classes: "husky" and "wolf". Initially, all images were in the data/train directory. A custom splitting function created an 80/20 train-test split, resulting in:

* Training set: 80 images (40 huskies, 40 wolves).
* Test set: 20 images (10 huskies, 10 wolves).  
  The final structure is data/train/husky, data/train/wolf, data/test/husky, and data/test/wolf.



**3. Methodology**

**3.1 Data Loading**

The load\_data function loads images, converts them from BGR to RGB, resizes them to 256x256 pixels, and normalizes pixel values to [0, 1]. Labels are assigned as 0 (husky) and 1 (wolf). The function successfully loaded 80 training images and 20 test images.

**3.2 Data Splitting**

The dataset was split into training and test sets with an 80/20 ratio using scikit-learn’s train\_test\_split with stratify to ensure balanced classes. Files were moved to data/test/husky and data/test/wolf.

Training Sample : 

Test Sample : 

**3.3 Data Preprocessing**

Preprocessing included:

* **Resizing**: All images were resized to 256x256 pixels.
* **Normalization**: Pixel values were scaled to [0, 1].
* **Augmentation**: Using albumentations, the following transformations were applied to all training images:
  + Rotation: ±20 degrees.
  + Horizontal Flip: 50% probability.
  + Random Resized Crop: Scale between 0.8 and 1.0.  
    The augmented images doubled the training set size to 160 images (80 original + 80 augmented).

**3.4 Model Creation and Hyperparameter Tuning**

A Logistic Regression model was implemented using scikit-learn. Hyperparameter tuning was performed with GridSearchCV over:

* C: [0.01, 0.1, 1, 10, 100].
* penalty: ["l1", "l2"].
* max\_iter: [100, 500, 1000].  
  The model was trained on the extended training set (160 images), with 5-fold cross-validation to select the best parameters.

**3.5 Evaluation**

The best model was evaluated on the test set (20 images). A placeholder test\_path = 'path/to/test' was added for custom test evaluation, which skips evaluation if the placeholder is detected, ensuring "Run All" compatibility.

**4. Results**

**4.1 Hyperparameter Tuning Results**

The table below summarizes the cross-validation results for all hyperparameter combinations:

| **C** | **Penalty** | **Max Iter** | **Mean CV Accuracy** | **Std CV Accuracy** |
| --- | --- | --- | --- | --- |
| 0.01 | l1 | 100 | 0.5000 | 0.0000 |
| 0.01 | l2 | 100 | 0.9125 | 0.0848 |
| 0.01 | l1 | 500 | 0.5000 | 0.0000 |
| 0.01 | l2 | 500 | 0.9125 | 0.0848 |
| 0.01 | l1 | 1000 | 0.5000 | 0.0000 |
| 0.01 | l2 | 1000 | 0.9125 | 0.0848 |
| 0.10 | l1 | 100 | 0.5000 | 0.0000 |
| 0.10 | l2 | 100 | 0.8875 | 0.1146 |
| 0.10 | l1 | 500 | 0.5000 | 0.0000 |
| 0.10 | l2 | 500 | 0.8875 | 0.1146 |
| 0.10 | l1 | 1000 | 0.5000 | 0.0000 |
| 0.10 | l2 | 1000 | 0.8875 | 0.1146 |
| 1.00 | l1 | 100 | 0.8750 | 0.1046 |
| 1.00 | l2 | 100 | 0.9000 | 0.1225 |
| 1.00 | l1 | 500 | 0.8750 | 0.1046 |
| 1.00 | l2 | 500 | 0.9000 | 0.1225 |
| 1.00 | l1 | 1000 | 0.8750 | 0.1046 |
| 1.00 | l2 | 1000 | 0.9000 | 0.1225 |
| 10.00 | l1 | 100 | 0.8250 | 0.1649 |
| 10.00 | l2 | 100 | 0.9000 | 0.1225 |
| 10.00 | l1 | 500 | 0.8250 | 0.1649 |
| 10.00 | l2 | 500 | 0.9000 | 0.1225 |
| 10.00 | l1 | 1000 | 0.8250 | 0.1649 |
| 10.00 | l2 | 1000 | 0.9000 | 0.1225 |
| 100.00 | l1 | 100 | 0.8250 | 0.1649 |
| 100.00 | l2 | 100 | 0.8875 | 0.1392 |
| 100.00 | l1 | 500 | 0.8250 | 0.1649 |
| 100.00 | l2 | 500 | 0.8875 | 0.1392 |
| 100.00 | l1 | 1000 | 0.8250 | 0.1649 |
| 100.00 | l2 | 1000 | 0.8875 | 0.1392 |
|  |  |  |  |  |

**4.2 Model Performance**

* **Best Hyperparameters:** {'C': 0.01, 'max\_iter': 100, 'penalty': 'l2'}
* **Test Accuracy:** 0.95
* **Classification Report:**  
  precision recall f1-score support  
  husky 1.00 0.90 0.95 10  
  wolf 0.91 1.00 0.95 10  
  accuracy 0.95 20  
  macro avg 0.95 0.95 0.95 20  
  weighted avg 0.95 0.95 0.95 20
* **Confusion Matrix:**  
  [[ 9 1]  
  [ 0 10]]

**5. Discussion**

The Logistic Regression model achieved a test accuracy of 95%, indicating strong performance. The l2 penalty consistently outperformed l1, with the best C value of 0.01 yielding a mean cross-validation accuracy of 0.9125.