Malaria Cells Image GLM Classifier

Advanced Statistical Modeling for Big Data
09/01/2023

Donato Gallo
0222400837

Mattia Lamberti
0222400838

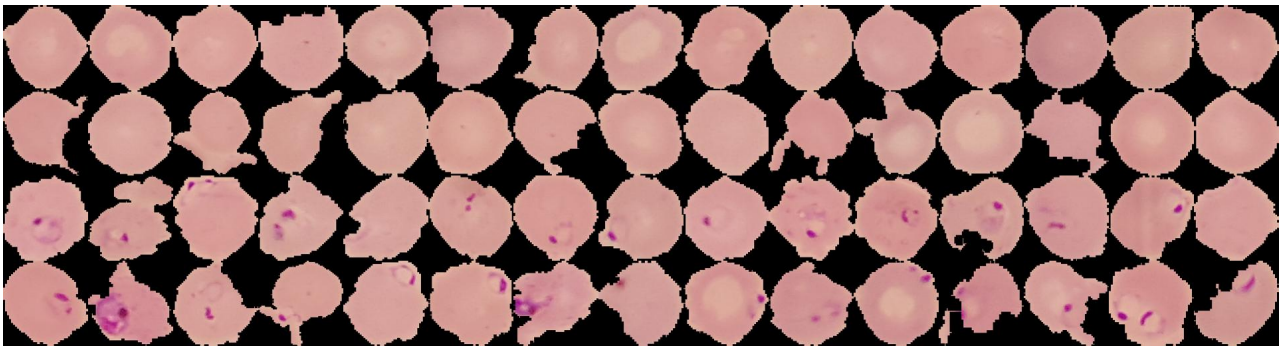


Figure 1: Sample of 60 random images from the dataset. The first two rows are uninfected cells, the bottom ones malaria infected. Generally, infected cells can be visually distinguished by the darker parasitic stain.

1 Introduction

Malaria is a lethal mosquito-borne infectious disease. It is the leading cause of morbidity and mortality in many nations, in fact, it is estimated by the World Health Organization (WHO) to be the cause of death of about 1 million people around the world, overall infecting 300 to 500 million, every year. Although many modern advancements in the field of malaria diagnosis, manual microscopy for the examination of blood smears is currently "the gold standard", despite relying on special training and considerable expertise, which often lack in the rural areas where malaria is endemic, causing manual microscopy not to be a reliable screening method [2]. An automated system capable of such visual diagnosis may be the solution for this problem.
At its core, the problem consists of distinguishing between non-parasitic cells and the malarial parasites infected ones using visual information (i.e. images), which we can interpret as a classification problem, in which the ultimate goal is to assign a label ("infected", "uninfected") to unseen cells images. In this sort of applications, in the recent years, deep convolutional neural networks (CNNs) are generally considered to be the best-performing approach to the

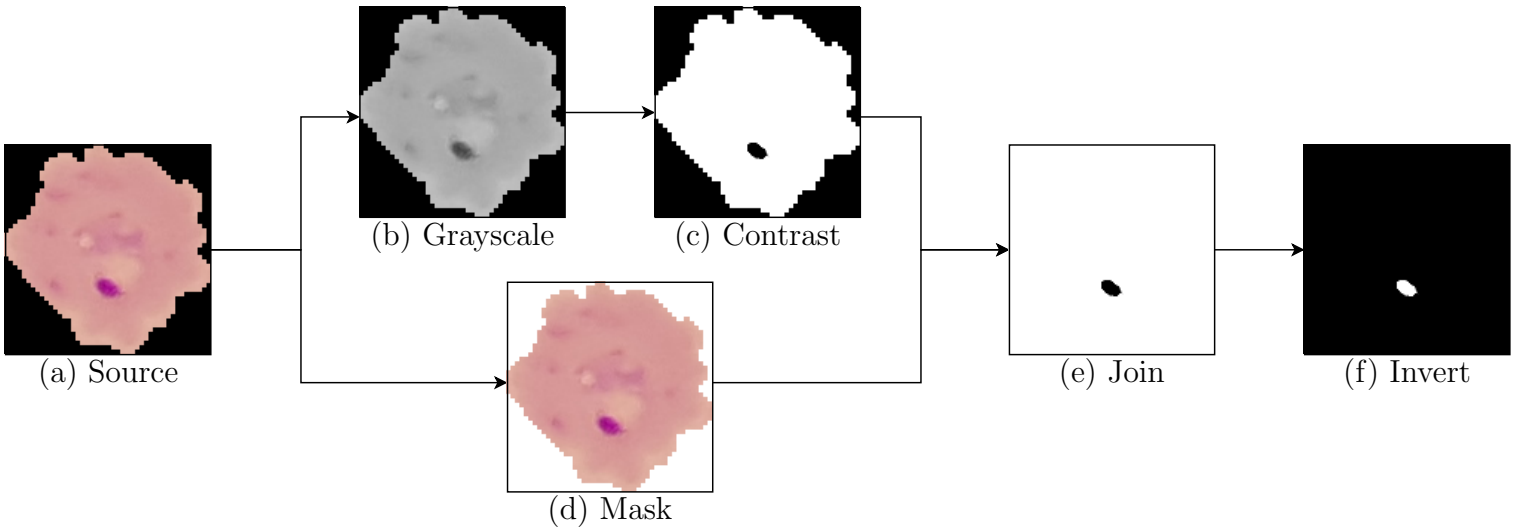


Figure 2: Visual representation of the preprocessing pipeline.

problem. In this project, however, we examine the possibility of alternative methodologies based on much simpler Generalized Linear Models (GLMs), investigating their pros and cons.

2 Data

The dataset used in this project is made-up of $27,558$ close-up images of cells belonging to two classes: infected and uninfected, acquired on standard light microscopy equipment. Infected malaria cells can be visually distinguished by a darker parasitic stain on them, which is absent for uninfected cells, the latter presenting a more homogeneous appearance in color, as can be seen in Figure 1.
Source images come in different sizes in the RGB color space (3 channels), averaging about $150 \times 150 \times 3$ pixels. The dataset is perfectly balanced between classes, containing $13,779$ (50%) images for each class. The dataset is publicly available on Kaggle.

3 Preprocessing

Convolutional Neural Networks are widely used in the areas of computer vision and imaging because of their automatic feature-extraction capabilities. When using a more traditional approach this aspect is generally lost. In order to fill this gap, preprocessing procedures are crucial to achieve better performance.
Based on the previous observation of the infected cells being distinguishable due to their darker spot, we tried different techniques in order to enhance and make it more visible, thus maximising the signal-to-noise ratio in this regard.
Referencing Figure 2, the source images **(a)** are first converted to 1 channel grayscale **(b)**, thus dropping 2 of the 3 channels, vastly reducing dimensionality. The contrast is then enhanced **(c)**, making the dark stain much more evident and reducing the non-stain-related noise. Simultaneously to steps (b) and (c), a mask **(d)** is created from source images to convert the black background to a white one. Mask and contrast enhanced images are joined together **(e)** to obtain a black stain on white background image. Although these images can

Label	$p_{1,1}$	$p_{1,2}$	...	$p_{32,32}$
0	0	0	...	0
0	0	0	...	0
...	...	...	...	...
1	0	250	...	0

Table 1: Example of the matrix-form dataset. Each row is a vectorized image.

Preprocessing	Test accuracy (%)
0-1 encoding	63.1
Grayscale	67.2
Grayscale, contrast	66.4
Grayscale, contrast, mask	89.5
Grayscale, contrast, mask, inverting	89.9

Table 2: Preprocessing pipelines test accuracy. Last row corresponds to the full process described in section 2.

be perfectly usable, we ultimately inverted the color space resulting in a white stain on black background **(f)**, optimizing the memory storage space required for the images, due to the white being coded as 255 while black 0, therefore requiring many fewer bits.
Finally, in order to use more classical statistical learning methodologies, all of the images are required to be of the same dimension and available in a matrix form in which each row is an image (observation), and each column a pixel, thus making up the final dataset. With respect to the first point, we resized all the images to 32×32 (clearly, 1 channel) by bicubic under-scaling, reducing the dimensionality even further. One could argue that resizing prior to the preprocessing steps may be much more computationally efficient, because it would imply working in reduced dimensionality from the beginning. However, we found out, that this may eventually cause graphical artifacts to the final image. As for the matrix-building process, each image is vectorized in a row vector of length 1.025, in which the first element is the label coded as 0 for uninfected and 1 for infected, while the remaining 1.024 are the $p_{i,j}$ for $i, j = 1, ..., 32$ pixels. These row vectors are ultimately row-binded together to obtain the final matrix of size $27,558 \times 1.025$, an example of which can be seen in Table 1.

4 Modeling

As mentioned before, we explored the possibility of image classification via GLM. Since our response variable is binary, we assume the model's random component to follow a Bernoulli distribution, and the link function to be a logit, simply resulting in a logistic regression.
First of all, we tried several preprocessing techniques, including various combinations of grayscale conversion, contrast enhancement, 0-1 bit color conversion, masking and color space inversion. In order to evaluate performance we splitted the data between train and test sets with proportions 0.7 and 0.3 respectively, and considered test accuracy to select the most appropriate preprocess pipeline. The results are shown in Table 2. We selected the pipeline described in section 3.
In general, this pipeline sets to zero almost every pixel with the exception of the parasitic

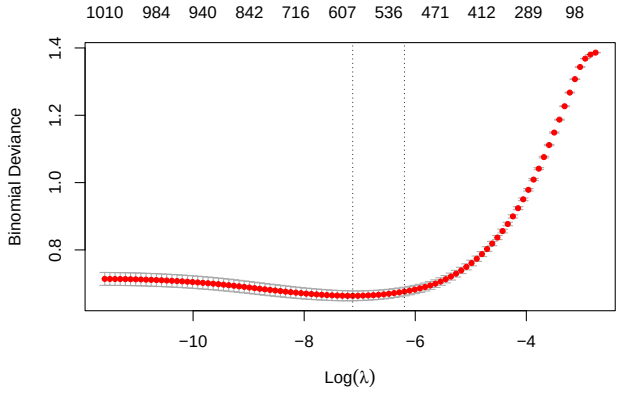


Figure 3: Cross validation binomial deviance for various values of λ. The vertical lines refer to the minimized and one-standard-error selected ones.

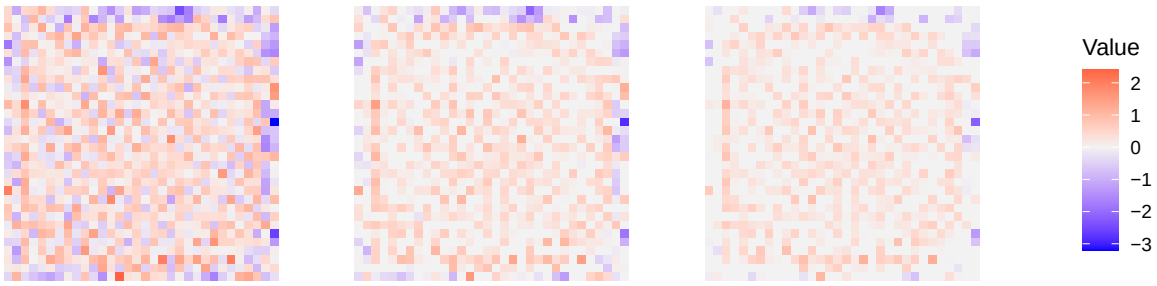
stain (in Figure 2 a quite ideal situation is shown, in which every non-stain pixel is extremely close to zero). This may eventually cause multicollinearity problems. To avoid this, we added a small uniform noise to every pixel. Despite generally lowering the signal to noise ratio, this has a great impact on the computational efficiency of the estimate process, making the optimization algorithm converge much faster.

4.1 Regularization

In almost every case, the cells appear to be shaped more or less in a rounded way, consequentially, the parasitic stain almost never occupies cornering regions of space, therefore making the pixels in the corners not so relevant. Furthermore, due to the relatively elevated dimensionality of the model (a high number of predictive variables), we are willing to reduce the variance of estimates, in exchange for an increase in bias. For these reasons, to avoid making decisions about which variables to include in the final model, we opted, in a more data-driven way, for a regularized approach. In order to completely set to zero as many coefficients as possible we Lasso-regularized the previous best performing model. The optimum problem then can be written as:

$$\hat{\beta} = \arg\min_{\beta} \; -\ell(\beta) + \lambda \sum_{j=1}^{p} |\beta_j|$$

where $\ell()$ is the log-likelihood function for a Bernoulli distribution. The value of λ controls the strength of the regularization, making its selection crucial. We performed 10-folds Cross-Validation (CV) for its choice, selecting the value of λ which minimizes the binomial deviance. To select an even more parsimonious model, we also considered the λ chosen by the one-standard-error rule. CV results are shown in Figure 3.
The minimized λ presents 607 non-zero coefficients (41% less), and achieves a test accuracy of 89.6%, while the one-standard-error rule selected one has 536 non-zero coefficients (48% less) and a test accuracy of 89.4%. Since the performance of the models are practically the same, we selected the more parsimonious, the one obtained with the one-standard-error λ.



(a) No penalty (b) λ min (c) λ one-se

Figure 4: Matrix representation of the coefficients for the three models.

4.2 Coefficients matrix form representation

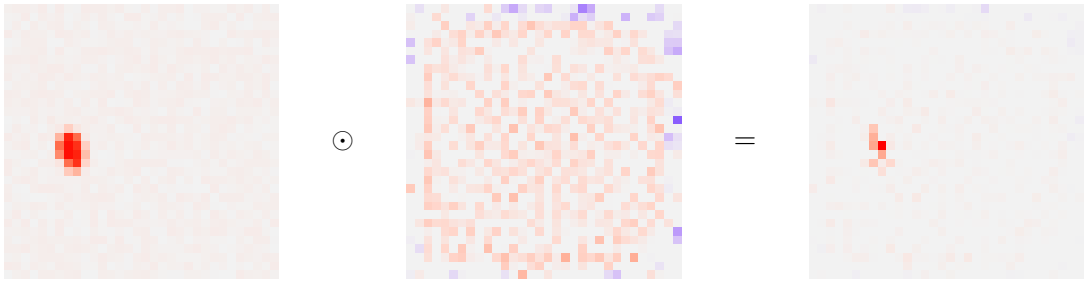
Among the advantages of using a simple GLM model over a CNN, a topic which will be more broadly discussed later, there is the possibility of easily interpreting the coefficients. In Figure 4, coefficients from each model have been rearranged in a 32×32 matrix, such that each coefficient represents the contribution of a single pixel over the probability that an image belongs to one class or the other. For ease of interpretation, we considered the simple transformation $(\exp(\beta_j) - 1) \cdot 100$, which is interpretable as the estimated percent change in odds in favour of the event "$y =$ infected" when $p_{i,j}$ is increased by one unit. As we outlined before, cornering pixels get progressively set to zero the more parsimonious the model we chose. A circular structure also emerges from the positive coefficients in all of the models, becoming more evident when increasing the penalization, and presumably represents the position in which we expect to observe the stains. Likewise, negative coefficients on the edges of the image penalize noise in that region.

Rearranging coefficients in this way is also quite convenient from a computational point of view, since we can compute predicted probabilities for unseen images by simply pairwise multiplying (Hadamard product) the pixels matrix of the image with the coefficients matrix, without the need for vectorizing the new image to a row vector and transpose. Formally:

$$\hat{\mu}_0 = \frac{\exp(\hat{\beta}_0 + \sum_{i,j=1}^{32} \hat{M}_{i,j})}{1 + \exp(\hat{\beta}_0 + \sum_{i,j=1}^{32} \hat{M}_{i,j})}$$

where β_0 is the model's intercept, $\hat{M} = X_0 \odot \hat{B}$ is the 32×32 matrix of the Hadamard product between X_0, the unseen preprocessed image in its matrix form (32×32), and $\hat{B}$ is the coefficient matrix described above.

We visually described the product in Figure 5. In subfigure (a) there is an example of a preprocessed unseen image from the test set (X_0). In (b) there is the coefficients matrix from the one-se model ($\hat{B}$). Subfigure (c) represents the product matrix $\hat{M}$, which can be interpreted as the signal captured by the model. As an analogy to a CNN context, we could think of the matrix $\hat{B}$ as a single filter, while $\hat{M}$ could represent the resulting pixelmap.



(a) X_0 (b) $\hat{B}$ (c) $\hat{M}$

Figure 5: Example of the pairwise product between the preprocessed image X_0 and the coefficients matrix $\hat{B}$.

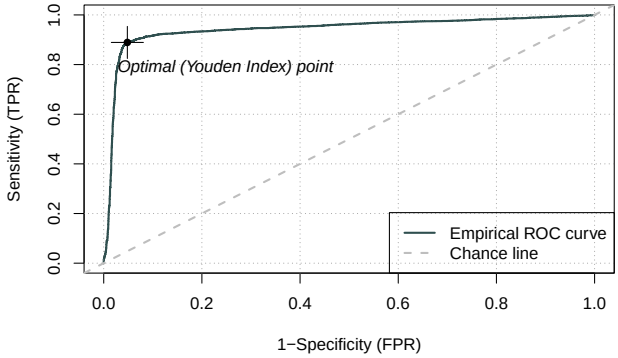


Figure 6: ROC Curve

4.3 ROC Curve

Ultimately, we tried to improve the predictive performance of the model by adjusting the decision rule for class assignment, which is normally by default set to 0.5. We did this by analyzing the Receiver Operating Characteristic (ROC) Curve. We reported an area under the curve (AUC) of 0.946, which is quite optimal, as can be seen in Figure 6. As for the decision rule cutoff, we selected the one identified by the Youden index, which optimizes the trade-off between true-positive and false-positive rate, in this case 0.311.
In Tables 3 and 4 we reported classification results as confusion matrices. Adjusting the cutoff improves the overall accuracy from 89.9% to 92.2% on the test set.

	$y = 0$	$y = 1$
$\hat{y} = 0$	0.485	0.090
$\hat{y} = 1$	0.014	0.409

Table 3: Confusion matrix, decision rule cutoff 0.5. Overall accuracy 0.899.

	$y = 0$	$y = 1$
$\hat{y} = 0$	0.474	0.052
$\hat{y} = 1$	0.026	0.448

Table 4: Confusion matrix, decision rule cutoff 0.311. Overall accuracy 0.922

5 CNN Comparison and conclusions

As long as the comparison with networks goes, we trained a simple CNN on the raw unpreprocessed train set to have 2 convolutional layers (16 filters 3×3 and 4 filter 3×3) and one final fully connected layer (8 neurons). This simple CNN scores an accuracy of 94.8%. More complex deep convolutional architectures are able to reach 95.7% on the test set [1].
At the expense of this small reduction in accuracy, our GLM logistic model offers several advantages. First of all, we have the interpretation of the coefficient, which is absolutely lacking in the CNN. In the second place, computational advantages are astonishing. In general, a classical GLM approach is commonly faster when evaluating new unseen data, and much faster when training. In our case, however, the GLM model requires vectorizing the original image, thus making the whole prediction dataflow in practice much slower than the network, which operates on raw data in matrix form. Our proposed way of computing prediction, discussed in section 4.2, however, bypasses this problem. We measured computational times to evaluate the whole test set on the same single core processor at 3.8GHz, for every methodology. Standard GLM vectorizing approach takes about 21.2 seconds. CNN takes 1.5 seconds, 14 times faster, while our matrix GLM approach takes 0.06, about 350 times faster than the classical GLM and 25 times faster than the CNN. This mainly depends on the fact that pairwise product is generally extremely optimized for modern hardware.
In conclusion, working with images is generally a hard computational task. It frequently requires large amount of storage capacity, high degrees of parallelism, moreover, processing data in real time may be an even harder task, often times requiring expensive hardware. The procedure discussed in this work aims to provide a faster and more accessible methodology for image classification, taking advantage of much simpler but very effective Generalized Linear Models.

References

[1] Negi Alok, Kumar Krishan, and Prachi Chauhan. Deep learning-based image classifier for malaria cell detection. *Machine Learning for Healthcare Applications*, pages 187–197, 2021.

[2] F Boray Tek, Andrew G Dempster, and Izzet Kale. Computer vision for microscopy diagnosis of malaria. *Malaria journal*, 8(1):1–14, 2009.