Université du Québec à Montréal
Laboratoire de Statistique et Actuariat

Technical Report
2^{nd} year summer internship

Submitted by :
DAVILA Andréa

ICM Student
at
Mines Saint-Étienne

Internship supervisors :

Professor BEAULAC Cédric
Postdoctoral Researcher MOINDJIÉ Issam

September 15, 2025

Acknowledgements

I am sincerely grateful to my supervisors, Cédric Beaulac and Issam Moindjié, for giving me such an amazing first experience in a research environment. Thank you for your support and enthusiasm towards my work, it truly helped me gain confidence in my abilities and encouraged me to pursue a career in research. I had really missed feeling this intellectually stimulated by my studies.

I am also thankful for the time you devoted to my project, and I greatly enjoyed our weekly meetings. I especially appreciated the freedom you gave me, I never felt pressured or judged, only supported. That made this experience all the more rewarding.

This internship was funded by my supervisor Cédric's personal research fund, part of the budget allocated to the Statistics Laboratory by the University of Quebec at Montreal. This opportunity would not have been possible without it, and I am truly grateful that Cédric trusted me with this project and made this unforgettable experience possible.

Contents

1 Host Organization Overview 2

1.1 The University 2

1.2 The Laboratory of Statistics and the Team 2

2 Abstract 3

3 Scientific Problem and Assigned Tasks 3

3.1 Scientific Problem 3

3.2 Specific Objectives and Success Criteria 4

3.3 Expected Deliverable 4

4 Dataset 5

4.1 Class Imbalance 5

5 Segmentation of the lesions 6

5.1 Main challenges of the segmentation 6

5.2 Hair removal 6

5.2.1 Hair removal pipeline 6

5.3 Segmentation algorithm and Grayscale channel choices 7

5.4 Border extraction 8

6 Functional approach for statistical shape Analysis 8

6.1 Pipeline for Shape Alignment 8

6.1.1 Definitions 8

6.1.2 Alignment Problem 9

6.1.3 Fourier Representation and Numerical Optimization 10

6.1.4 illustration of the procedure 10

7 Modeling 11

7.1 ANN 11

7.1.1 Pipeline used and Features retained 11

7.2 CNN 13

7.3 Hyperparameter space 13

8 Experimentation and Results 14

8.1 Experimentation Protocol 14

8.2 Hardware used 14

8.3 Results 15

9 Conclusion 16

9.1 Discussion and Challenges 16

10 Appendix 18

10.1 Stratified Nested CV tabular results 18

10.1.1 ANN 18

10.1.2 CNN 19

References 19

1 Host Organization Overview

This internship was carried out as part of my Master of Engineering, at École des Mines de Saint-Étienne. It took place at UQÀM (Université du Québec à Montréal), within the Laboratory of Statistics, under the supervision of Professor Beaulac Cédric and Postdoctoral Researcher Moindjié Issam.

1.1 The University

As its name suggests, UQÀM is located in Montreal and is a public University welcoming 39 427 students, including 30 185 undergraduates, 6 433 graduate students, and 2 116 doctoral students.

1.2 The Laboratory of Statistics and the Team

Its research axes include:

- stochastic modelling and applications.
- biostatistics and statistical genetics.
- actuarial and financial risk analysis, including general insurance, provisioning, ruin theory, and credit risk.
- Image processing using functional data analysis and statistical results *(current field of interest of my supervisors).*

Within the team, the work focuses more specifically on statistical and machine-learning methods for high-dimensional data (like images). The team consists of approximately 20 researchers, PhD students, and engineers, and participates in projects such as the Chair Co-operators in Actuarial Risk Analysis (CARA), the research laboratory Quantact in actuarial and financial mathematics, and the centre STATQAM (Statistics and Data Science, where I was assigned).

2 Abstract

In the field of medical imaging, the community is particularly interested in the shapes of certain organs or lesions, as features such as **asymmetry, border irregularity, and texture** are often correlated with **cancerous pathologies**. Consequently, the ability to quantify shape characteristics from images is crucial for supporting accurate and early diagnosis.

The state-of-the-art approach in this domain relies on representing shapes as pixel-based images, and applying feature extractors based on deep learning. Specifically **Convolutional Neural Networks (CNNs)**. CNNs are capable of learning abstract patterns from images, which can be used to differentiate shapes according to specific classes, which is essential since using raw pixel matrices alone is generally insufficient. However, CNNs demand high-end GPUs with large amounts of VRAM, and their training can be extremely time-consuming.

As an alternative to CNNs, my supervisors developed a pipeline based on **Functional Data Analysis** [1], applied on curves. This allows shapes in images to be represented in ways other than simple pixel grids. My role was to **implement and apply this pipeline to a classification task**, using the extracted features as inputs to a Multilayer Perceptron (MLP), and to compare its performance against that of CNNs.

The research question guiding this work could be: *Can Statistical Shape Analysis provide shape-based features for lesion classification that match or surpass the performance of CNNs, while being computationally more efficient ?*

The objectives of the internship were to:

- Learn about Functional data analysis and Statistical shape analysis.
- Build a segmentation pipeline for the HAM10000 dataset's moles.
- Implement my supervisors' shape analysis pipeline.
- Design and compare ANN and CNN architectures on classifying moles of the HAM10000 dataset.

3 Scientific Problem and Assigned Tasks

3.1 Scientific Problem

The studied problem concerns the classification of moles from the HAM10000 dataset, using shape information only. It is motivated by the need of an alternative approach to the classic pixel based representation and CNNs ; an approach with much more interpretability (we know what each feature exactly represent) and accessibility. Several major challenges arised:

- **Challenge 1:** mole segmentation - which is complex considering the variability of mole shapes, constrast and colors.
- **Challenge 2:** understanding the whole shape analysis pipeline and implementing it efficiently.

- **Challenge 3:** building a robust experimentation protocol for model comparison, considering the tremendous need for VRAM induced by training CNNs.

3.2 Specific Objectives and Success Criteria

Objectives

1. **Compare performance**: Evaluate and benchmark different model architectures on the given task, focusing on predictive accuracy and robustness.
2. **Assess practicality**: Take into account additional factors such as accessibility of the architectures/pipelines used, model complexity, and training difficulty.

Success Criteria

- Demonstrated ability to **quantitatively compare performance** across models.
- Clear analysis of the **trade-offs** between accuracy, computational cost and model interpretation.
- Delivery of a **well-documented methodology**, facilitating reproducibility and potential re-use by the research community.

3.3 Expected Deliverable

- **A report** — with the ambition of publishing an open-access article accompanied by a public code repository.

4 Dataset

In order to compare ANN and CNN feature extraction, we decided to use the **HAM10000 dataset (Human Against Machine)**, which contains samples of dermoscopies gathered for the *ISIC 2018 Challenge* (a data challenge). The **HAM10000** dataset comprises **10,015 dermatoscopic images** of pigmented skin lesions across **seven** diagnostic categories, including *melanoma* and *benign nevi.* Images were collected from diverse clinical sources, with ground truth established via histopathology, expert consensus, or follow-up. Its size, diversity, and high-quality annotations make HAM10000 a standard benchmark for training and evaluating deep learning models in skin lesion classification.

We focused only on using shape-based features for mole classification. Doing so, the most relevant categories were ***melanoma*** and ***benign nevi***. Other lesion types often differ primarily in color, texture, or do not even have a proper shape, making them less informative for shape-focused analyses. We decided to restrict the dataset to those two classes to ensure that the models learn discriminative geometric patterns relevant to malignancy.

4.1 Class Imbalance

After removing the classes irrelevant for a shape analysis, we are left with:

- 1,115 images labeled as melanoma (**16.1 %** of the dataset)
- 6,711 images labeled as benign nevi (**83.39 %** of the dataset)

This is a major class imbalance, that will be handled in our modeling by using a weighted loss. The weights will be calculated by the `compute_class_weight` function from `sklearn`. For a dataset with n_{samples} total samples, n_{classes} classes, and n_c samples in class c, the class weights are given by:

$$w_c = \frac{n_{\text{samples}}}{n_{\text{classes}} \cdot n_c}$$

This assigns higher weights to underrepresented classes, thereby increasing their influence in the loss function, which makes the models adjust more when misclassifying the samples of the underrepresented class.

5 Segmentation of the lesions

In order to extract the border of the lesion, we need to segment it first. Here we will be detailing the automated segmentation pipeline we applied to the dataset. **We chose simple automated thresholding techniques such as Otsu's thresholding technique [2] and KMeans** and kept the best performing one based on our qualitative appreciation of the results. We avoided supervised learning methods (U-net) too costly and which would be equivalent to basically doing the work manually...

5.1 Main challenges of the segmentation

The main obstacles to a decent segmentation included:

- Circular frame from the lens used for the dermoscopy.

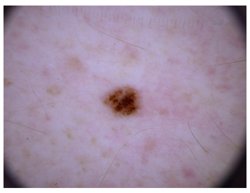


Figure 1: Mole with lens frames.

- Variability in the taints of the moles.

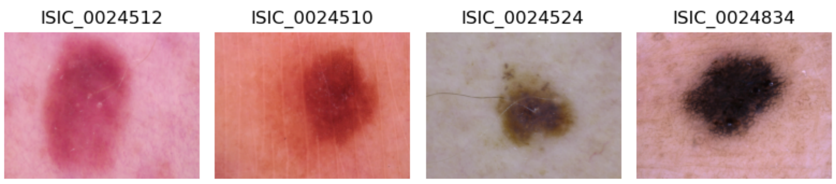


Figure 2: Samples from the HAM10000 with varying taints.

- Presence of hair in the image, which dark taints (similar to moles) can interfere with automated segmentation (see subsection 5.2).

5.2 Hair removal

Hair can mimic the edges of the lesion and therefore has to be removed for accurate border detection of the mole.

5.2.1 Hair removal pipeline

We followed the choice of *Kerem Gencer*[3] and applied our transformations to the red channel. The crucial step of this image processing pipeline is the blackhat filtering applied to the red channel, which emphasizes dark structures on a bright background. The extra steps aim to clean the resulting binarized mask. Finally, we inpaint the white regions in the original image using *Alexandru Telea's fast marching method (2004)* [4].

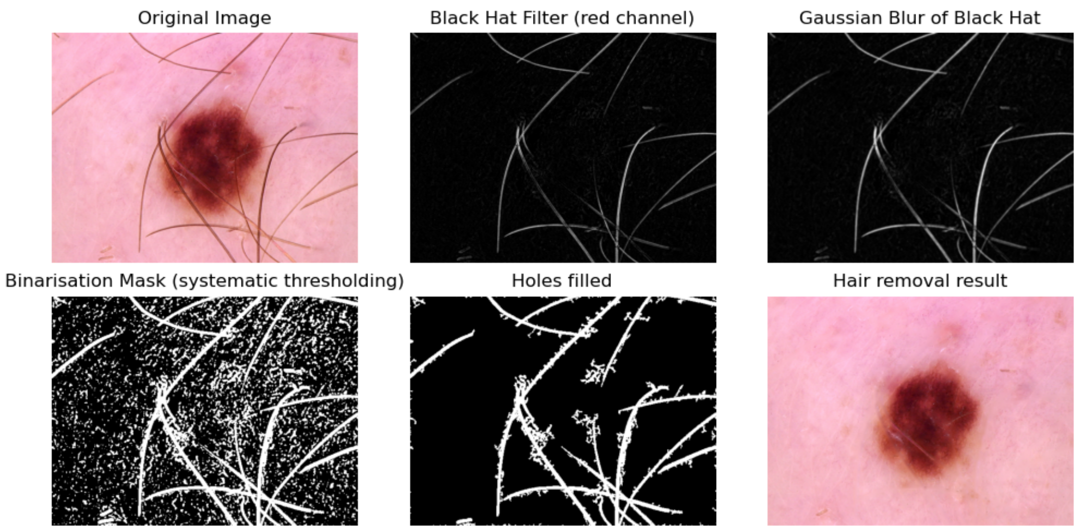

Original Image
Black Hat Filter (red channel)
Gaussian Blur of Black Hat
Binarisation Mask (systematic thresholding)
Holes filled
Hair removal result

Figure 3: steps included in the hair removal pipeline applied to the red channel of the image.

This pipeline showed satisfying hair removal with basic and computationally efficient operations.

5.3 Segmentation algorithm and Grayscale channel choices

Due to the variety of the colors amongst a single class, we found it relevant to tailor the grayscale channels automatically, in order to capture the maximum variance possible in a single channel for each image, before applying Otsu's thresholding algorithm [2].
Hence, we applied **PCA** to an **elliptical ROI (Region Of Interest)** of the hair-removed image (in order to hide the lens black frame), to all available channels (Red, Green, Blue). We constructed the **grayscale channel using the first principal component**, scaled to 0-255 (min-max scaling), with additional morphological operations in order to clean the mask and to **keep the largest connected component** (in terms of area, approximated to the number of pixels).

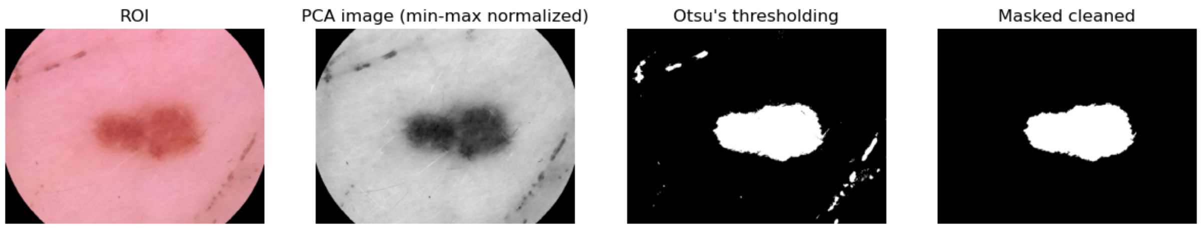


Figure 4: Segmentation pipeline.

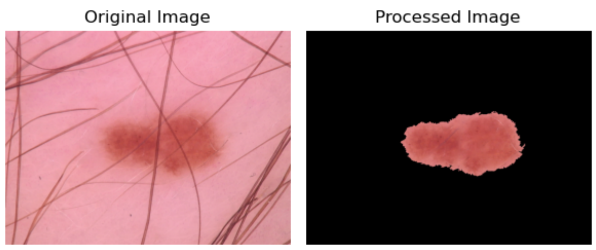


Figure 5: Initial image vs. Final image.

5.4 Border extraction

The next step was to extract the border from the segmentation mask. We decided to use the Marching Squares algorithm and choose the largest contour detected (largest in terms of perimeter approximated to the number of pixels detected).

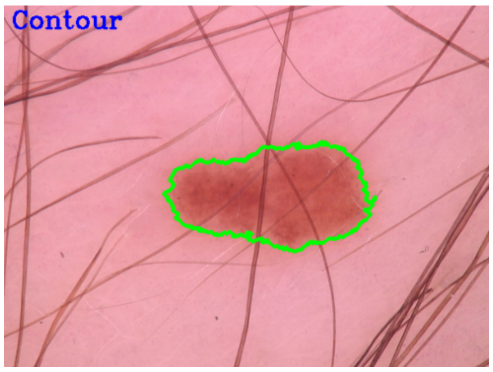


Figure 6: Initial image vs. Final image.

With the border extracted, we can now extract information using statistical shape analysis results.

6 Functional approach for statistical shape Analysis

In this section, we will quickly introduce the pipeline developed by Beaulac, Moindjié and Descary in *A functional approach for curve alignment and shape analysis* [1]. Hence, we will apply it to the border extracted from the moles, which is the shape under study. The alignment procedure will provide crucial continuous variables which can help for mole classification.

6.1 Pipeline for Shape Alignment

6.1.1 Definitions

The latent curve $\widetilde{\mathbf{X}}$ represents the *shape* of a random planar curve X, in the sense of Kendall (1989), i.e. the geometric information that remains after removing translation, scaling, rotation and reparametrization.

A planar curve $X : [0,1] \to \mathbb{R}^2$ is modeled as a deformed version of a latent shape $\widetilde{X}$:

$$X(t) = \rho\, O\, \widetilde{X} \circ \gamma(t) + T \quad t \in [0,1]$$

where the deformation parameters are:

- $\rho \in \mathbb{R}^+$: scaling factor
- $T \in \mathbb{R}^2$: translation vector
- $O \in SO(2)$: planar rotation
- $\gamma \in \Gamma$: reparametrization function

Γ is defined as the family of circular shifts. The parameter δ formalizes the starting point of the trajectory of the considered shape :

$$\Gamma = \Big\{\gamma_\delta(t) = \mathrm{mod}(t - \delta, 1),\ \delta \in [0,1]\Big\}$$

The functional space $H = L^2([0,1]) \times L^2([0,1])$ is equipped with the inner product

$$\langle f, g\rangle_H = \int_0^1 f^{(1)}(t) g^{(1)}(t)\, dt + \int_0^1 f^{(2)}(t) g^{(2)}(t)\, dt$$

so that shapes can be normalized to lie in the unit sphere

$$S_\infty = \{f \in H \mid \|f\|_H = 1\}$$

The *elastic distance* between two pre-shapes $f, g \in S_\infty$ is then defined by

$$d(f,g) = \min_{\theta \in [0,2\pi], \delta \in [0,1]} \|f \circ \gamma_\delta - O_\theta g\|_H$$

This distance is referred to as *"elastic"* since, by construction, it is invariant under rotation and reparametrization:

$$d(f,g) = d\Big(O_{\theta_1} f \circ \gamma_{\delta_1},\ O_{\theta_2} g \circ \gamma_{\delta_2}\Big) \qquad \forall(\theta_1, \delta_1),\ (\theta_2, \delta_2) \in [0, 2\pi] \times [0,1]$$

6.1.2 Alignment Problem

The alignment task consists in estimating the deformation parameters $(\rho, T, \theta, \delta)$ for each observed curve X in order to recover its underlying shape $\widetilde{X}$.

Translation

The translation is removed by centering the curve:

$$T = \int_0^1 X(t)\, dt$$

Scaling

The scaling factor is obtained from the norm of the difference:

$$\rho = \|X - T\|_H$$

We then define the normalized curve, centered and scaled.

$$X^*(t) = \frac{1}{\rho}\Big(X(t) - T\Big)$$

Rotation and Reparametrization

Given a *reference shape* μ, rotation and reparametrization are estimated by solving

$$(\hat{\theta}, \hat{\delta}) = \arg\min_{\theta, \delta} \|X^* - O_\theta \mu \circ \gamma_\delta\|_H^2$$

6.1.3 Fourier Representation and Numerical Optimization

To make the minimization tractable, both X^* and μ are projected on a truncated Fourier basis of dimension M:

$$X^*(t) = \sum_{k=1}^{M} \alpha_k \psi_k(t), \qquad \mu(t) = \sum_{k=1}^{M} u_k \psi_k(t) \quad \alpha_k, u_k \in \mathbb{R}^2$$

The coefficients are arranged into matrices $\alpha, u \in \mathbb{R}^{M\times 2}$.
The reparametrization γ_δ acts through a block-diagonal orthogonal matrix $\beta(\delta)$, while the rotation O_θ acts globally. The alignment criterion becomes

$$(\hat{\theta}, \hat{\delta}) = \arg\min_{\theta,\delta} \ \|\alpha - O_\theta u \beta(\delta)\|_F^2$$

where $\|\cdot\|_F$ is the Frobenius norm:

$$\|A\|_F^2 = \sum_{i,j} A_{ij}^2$$

Algorithm

1. Grid search over $\delta \in [0, 1]$ (e.g. step size 0.01).
2. For each δ, solve for the optimal rotation $\hat{\theta}_\delta$.
3. Select $(\hat{\theta}, \hat{\delta})$ minimizing the Frobenius error.

6.1.4 illustration of the procedure

In this example, we extracted the X and Y coordinates from a contour sampled according to our border extraction pipeline (1.4).

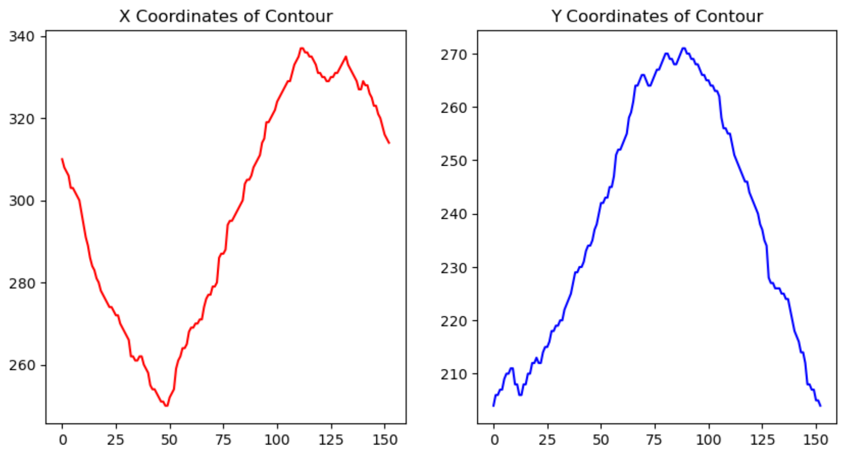


Figure 7: X and Y coordinates extracted from the contour of a mole.

We applied a known rotation and reparametrization to it, $\theta = \pi/4, \quad \delta = 0.1$, with $M = 101$.
Then used our pipeline in order to find those parameters. It managed to find exact solutions (error of 1e-15): $\hat{\theta} = \theta$ and $\hat{\delta} = \delta$

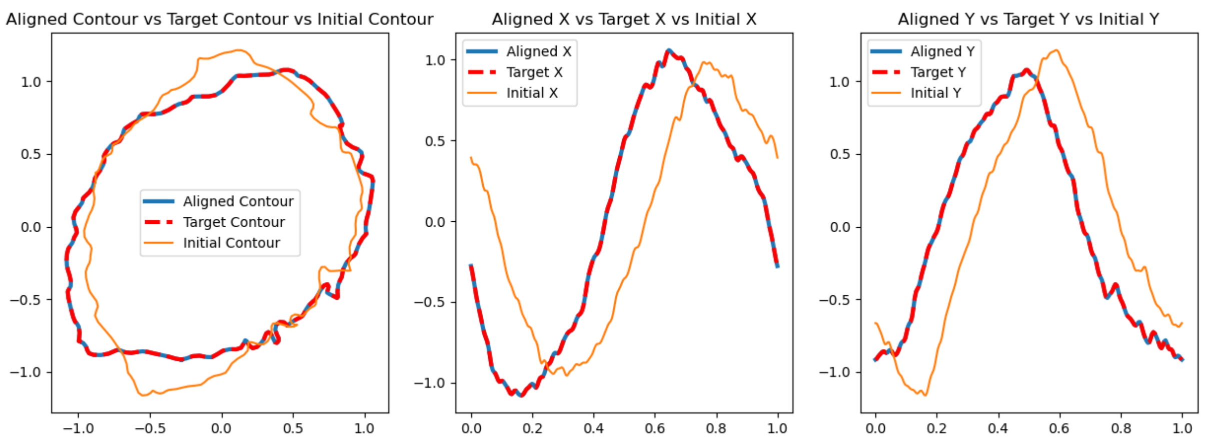


Figure 8: Result of the alignment procedure.

7 Modeling

7.1 ANN

Our approach is to use the alignment pipeline as a feature extractor, quantifying characteristics of mole shapes, aligning them on an arbitrary reference, and **passing the resulting coefficients to a Multi-Layer-Perceptron**.

7.1.1 Pipeline used and Features retained

We applied first the pipelines discussed in the sections 1.2 and 1.3 to our dataset, then proceeded to apply the alignment pipeline (2.1).
We aligned every contour on an arbitrary reference mole with a well-defined, regular shape of the 'nv' (non cancerous) class.

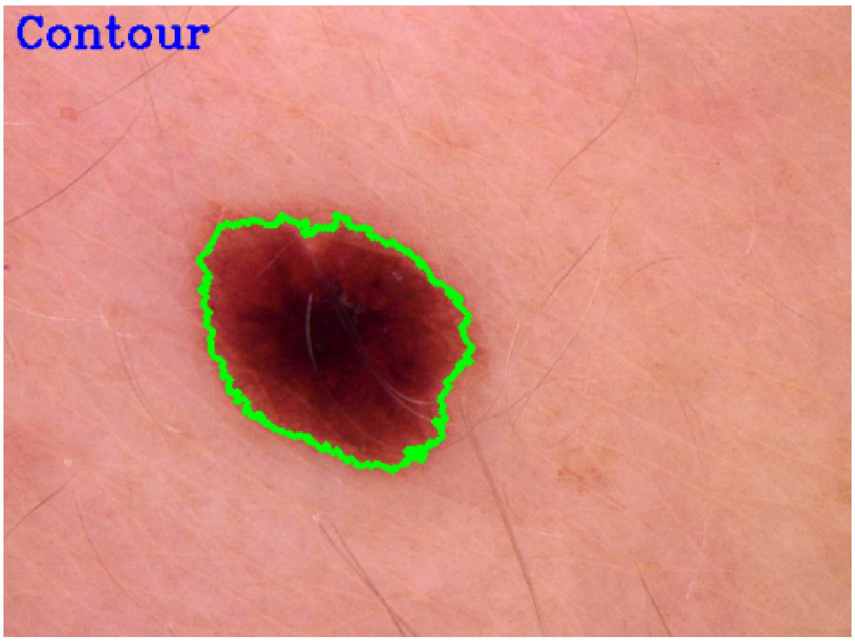


Figure 9: Reference we aligned other contours on - ISIC_0024803

Our basis expansion used $\mathbf{M} = \mathbf{101}$ Fourier functions to approximate the moles' contour.

Hence, the features we retained for the ANN are the following :

- **$\mathbf{2M} = \mathbf{202}$ aligned Fourier coefficients (float):** 101 for X and 101 for Y coordinates. These include the *Translation component* $\mathbf{T}$ (continuous component of the Fourier coefficients).
- **ρ (float) scaling factor:** norm of the coefficients without the constant Fourier term.
- **θ (float):** rotation angle applied to align the contour, in radians.
- **δ (float):** the reparametrization coefficient, which defines the aligned starting point of the trajectory (can be considered as the required phase shift used for alignment).
- $\min_{\theta,\delta} \|\alpha - O_\theta u\beta(\delta)\|_F^2$ **(float):** the Frobenius distance between the aligned contour and the reference.

The **ANN pipeline** is illustrated in the following flowchart :

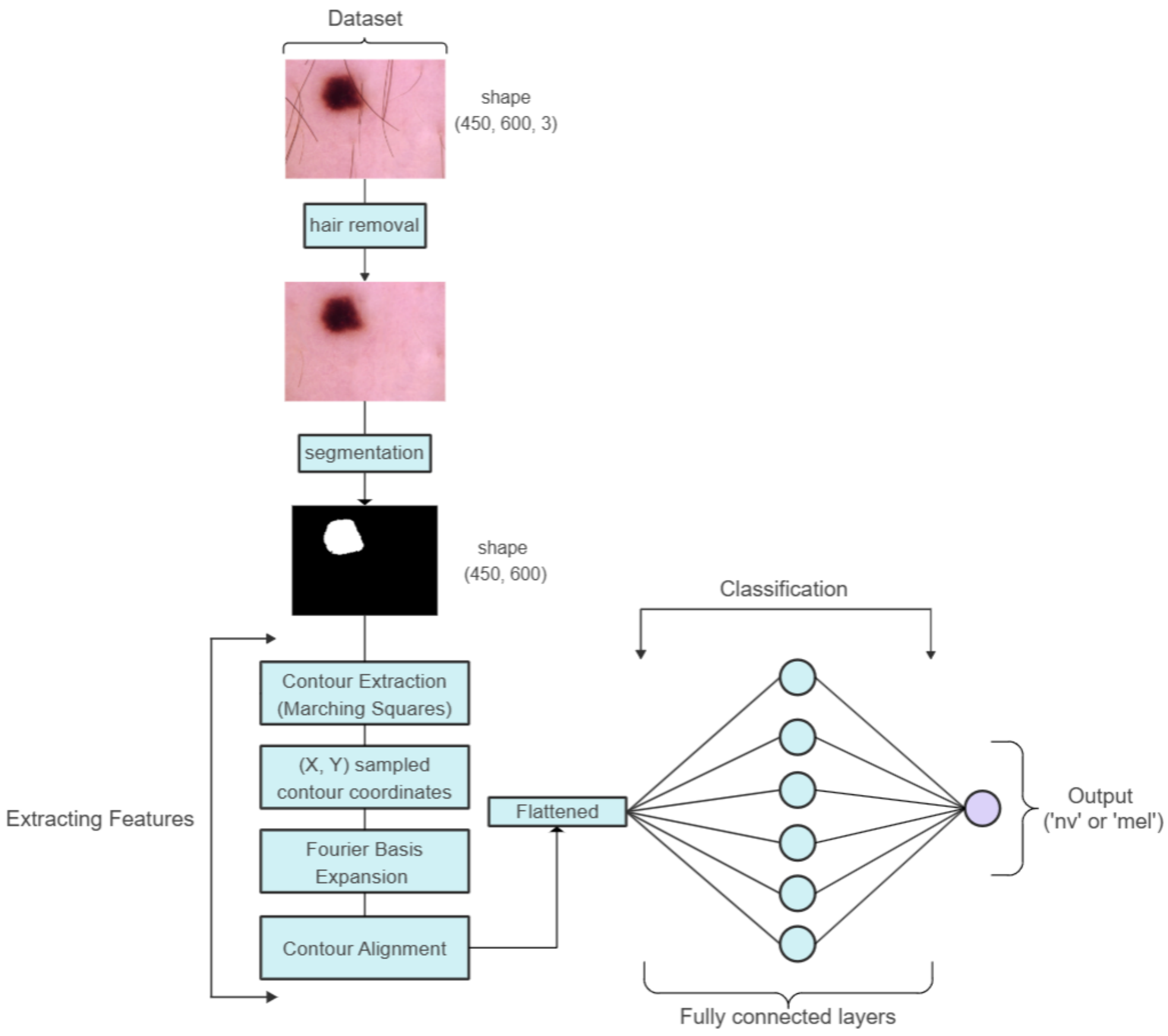


Figure 10: ANN flowchart.

7.2 CNN

We treat the CNN as a feature extractor, applied to a pixel representation of images. As such, we will feed to the CNN the full-size segmentation masks produced by the hair removal and segmentation pipelines, for a fair comparison with the ANN approach.

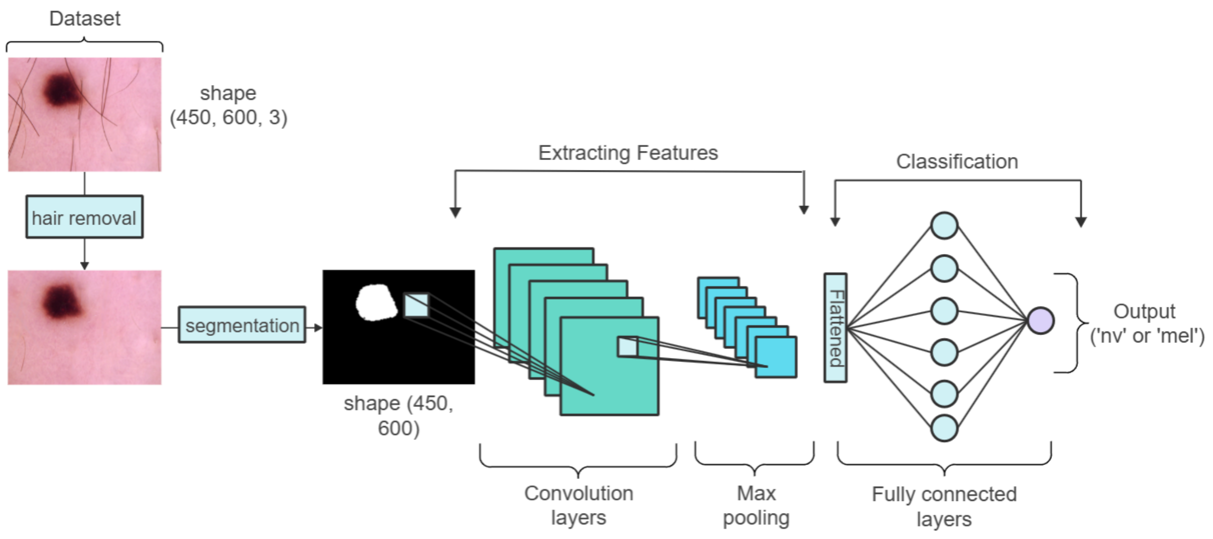


Figure 11: CNN flowchart.

7.3 Hyperparameter space

To define appropriate hyperparameter spaces for each architecture (CNN and ANN), we first performed **qualitative tests** followed by a **Bayesian Search with 100 iterations** for each model type. Based on these preliminary results, we selected hyperparameter ranges that consistently yielded good performance.

For the **CNN**, the search space included:

- Convolutional depth: 2–4 layers with filter sizes $\{16, 32, 64\}$, kernel size = (3, 3)
- Dense layer depth: 1–3 layers with units $\{32, 64, 128\}$
- L_2 regularization: 10^{-4}–10^{-2} (log scale)
- Dropout rates: 0.1–0.4
- Learning rate: 5×10^{-4}–2×10^{-3} (log scale)

For the **ANN**, the search space included:

- Dense depth: 1–3 layers with units $\{16, 32, 64, 128\}$
- L_2 regularization: 10^{-4}–10^{-2} (log scale)
- Dropout rates: 0.1–0.4
- Learning rate: 5×10^{-4}–2×10^{-3} (log scale)

Both architectures used **ReLU** activations, He-normal initialization, batch normalization, and were optimized with **Adam** on a **weighted binary cross-entropy loss**, with **AUC** as the evaluation metric, and a **batch-size of 32**.

8 Experimentation and Results

8.1 Experimentation Protocol

We evaluated our feature extractors using a **nested stratified cross-validation** with **n = 20 outer folds**. For each outer test fold, the remaining folds were used for **hyperparameter tuning** via an **inner cross-validation** with **10** randomly sampled configurations from a reduced search space. The best model was retrained on the 19 training folds and evaluated on the held-out test fold. We collected the resulting test metrics across folds to perform statistical comparisons.

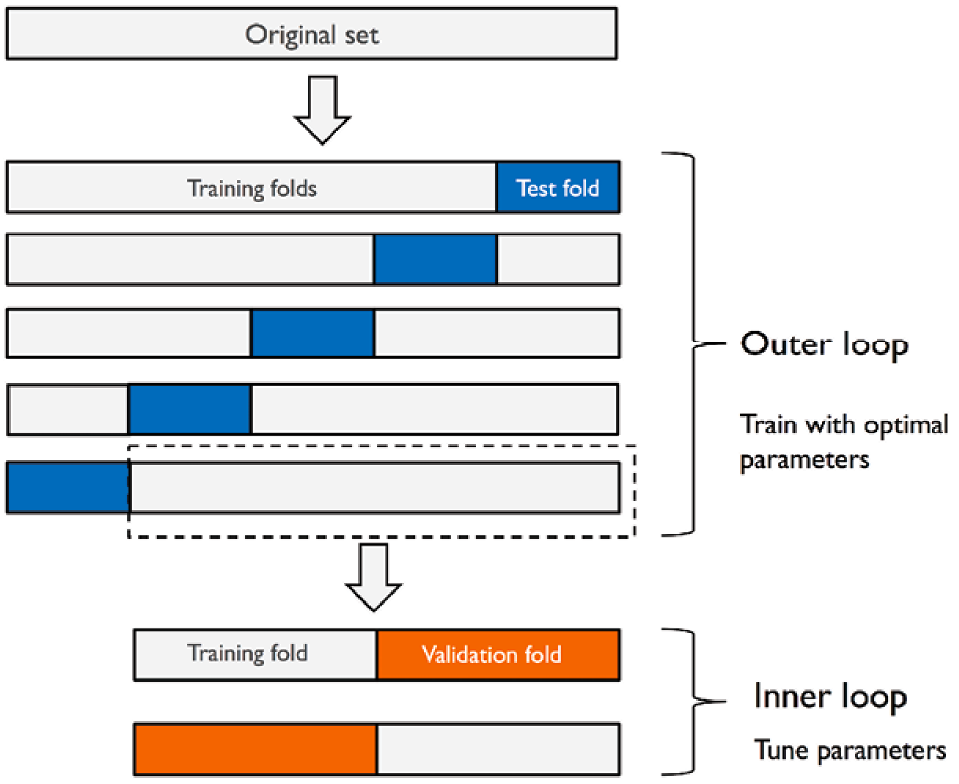
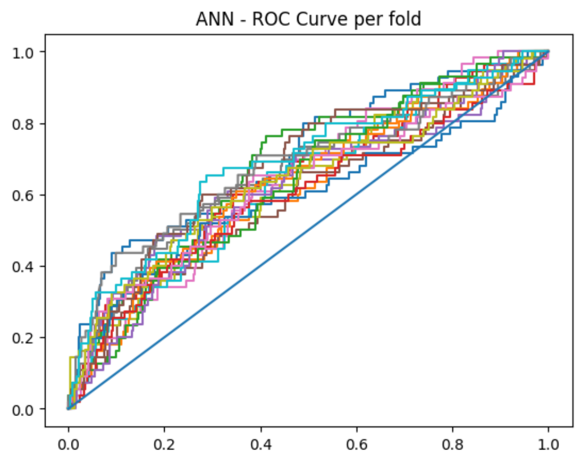


Figure 12: Nested Cross-Validation Chart.

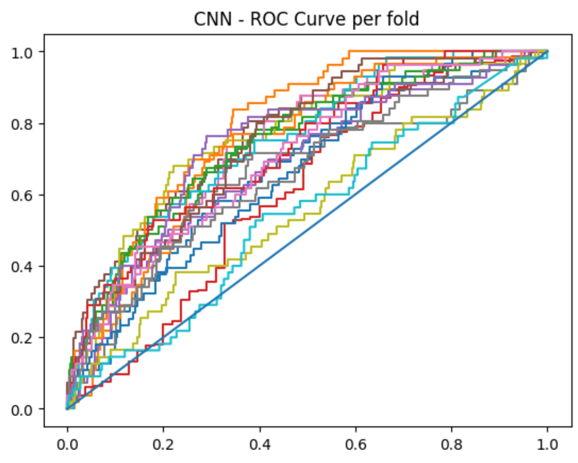
8.2 Hardware used

The training was conducted on Google Colab using an NVIDIA A100 GPU with 80 GB of VRAM. The complete process, including both training and evaluation, required a total runtime of 17 hours and 19 minutes.
The ANN architecture had a modest memory footprint of 0.7 GB, whereas the CNN consumed as much as 66.7 GB of VRAM, despite both models being trained with a **batch size of 32**. Full-resolution images were used for the experiments (450x600).

8.3 Results



(a) ROC Curves per fold - ANN



(b) ROC Curves per fold - CNN

Figure 13: Comparison of ROC curves per fold for ANN and CNN architectures.

Model	avg-AUC	avg-Bal.Accuracy	avg-F1	avg-Training time (s)
ANN	0.640 ± 0.031	**0.572 ± 0.036**	**0.837 ± 0.020**	**31.7 ± 6.92**
CNN	**0.697 ± 0.069**	0.544 ± 0.049	0.814 ± 0.016	174 ± 82.9
Naive	0.5	0.5	0.791	0

Table 1: Stratified-Nested-CV Results overview (The naive model simply predicted the majority class for all instances).

These results indicate that **CNNs achieve a higher AUC on average**, but at the cost of **substantially higher instability**, with over **twice the AUC standard-deviation of ANNs**. On the other hand, **Balanced Accuracy and F1 score are slightly higher on average for ANNs**, with **similar standard deviations for both models**. However, ANNs require on average nearly **5.5 times less training time** (including data standardization) than CNNs, while **using almost 95 times less VRAM** (0.7GB for ANNs vs. 66.7GB for CNNs).

To assess the statistical significance of the performance differences between models, **we computed the differences of metrics (AUC, Balanced Accuracy, F1) across all folds**. These difference vectors were then **bootstrapped 10,000 times** to estimate their **empirical distributions**. For each metric, we calculated the **2.5%** and **97.5% quantiles** of the bootstrap distribution to obtain **95% confidence intervals. Differences for which the interval did not include 0 were considered statistically significant, whereas intervals containing 0 indicated no significant difference**. This method for constructing confidence intervals was first introduced by *Bradley Efron* in 1982 [5] and has since been widely discussed in the literature, including practical explanations such as that provided by *the University of Virginia* [6].

Metric	95% CI (Bootstrap)	Significance	Comment
AUC	(-0.0924, -0.0171)	Significant	CNN performs significantly better
Balanced Accuracy	(-0.0031, 0.0574)	Not significant	No significant difference
F1 Score	(-0.0031, 0.0574)	Not significant	No significant difference

Table 2: Statistical comparison of ANN vs CNN using 95% bootstrap confidence intervals. If the interval does not include 0, the difference is considered statistically significant.

Our results indicate that among the evaluated metrics, only the **AUC shows a statistically significant difference**, with the **CNN outperforming the ANN**. For **Balanced Accuracy and F1 Score**, the confidence intervals include zero, suggesting that there is **no evidence of a meaningful performance difference between the two models for these metrics**.

9 Conclusion

This internship was an incredible opportunity for me to explore Functional Data Analysis and its application to Statistical Shape Analysis, through both Ramsay's book [7] and my supervisors' work [1]. It was also a first step in the field of medical image analysis for me, which I am passionate about.

The primary objective was to explore whether Statistical Shape Analysis and functional representations of contours could provide features that match or surpass the performance of CNNs, while being much more interpretable and computationally efficient. We showed that the **CNN architecture only had a slight edge regarding the AUC metric**, but at the cost of a **high variance** in the performance. **F1 and Balanced Accuracy showed no significant difference** between our approaches, all while the **ANN approach required less than a fifth of the training time of the CNNs** and nearly **95 times less VRAM**.

This result is promising, as it could make image-based models more accessible ! Additionally, the alignment pipeline can be applied to entire images, not just curves, which could open the door to numerous classification applications, especially in the field of medical imaging.

9.1 Discussion and Challenges

During this work, I faced several significant challenges. Training **CNNs** on full-resolution images (450x600) demanded **enormous computational resources**. I have lost weeks trying to train the CNN architecture on a A100 40Gb GPU, trying to simulate a high batch-size through complicated techniques, unsuccessfully. It was only at the very last moment that luckily, Google Colab offered a new A100 GPU offering 80 GB of VRAM. Still, those notebooks frequently crashed after around 12 hours of training. These interruptions led to a lot of time lost and necessitated careful management of experiments and checkpoints.

The segmentation task was also far from trivial: variability in lesion colors, presence of hair, and circular dermoscopy frames complicated border extraction. Despite these difficulties, the pipelines for hair removal, grayscale channel construction, and border extraction allowed for reasonably accurate shape representations. I took time to explore various algorithms and possibilities from other papers covering the segmentation for the same dataset. It was really frustrating to see that **most researchers used U-net Deep learning based architecture for the segmentation**, which I did not like at all since it required basically to do the work manually (to build a training dataset for segmentation).

In terms of achievements, I successfully implemented the full pipeline: automated segmentation, hair removal, feature extraction via functional shape analysis, and classification with a Multilayer Perceptron. I conducted a thorough comparison against CNNs using nested stratified cross-validation and bootstrapped confidence intervals under the advice from my supervisors.

Looking forward, several extensions could enrich this work. Segmentation remains imperfect, and few approaches could improve our modeling. **Exploring ensemble strategies, such as combining multiple ANNs or integrating hybrid ANN + CNN architectures**, could further increase predictive accuracy without dramatically increasing computation time, and I would have liked to try those if I had not encountered the problem related to Colab's crashes.

Overall, this internship showed me that working in a research environment is **extremely stimulating**, under the condition that the people you work with show recognition and enthusiasm towards your work. It is pleasing that they know what you're doing and discuss the issues you encounter in order to find new ideas together. However, it proved to be **quite lonely** ; the moments I preferred during this summer were definitely the ones I spent talking to my supervisors during our weekly meetings. I think I should have spent more time trying to meet the other members of the lab, instead of only focusing on getting my job done on my own.

10 Appendix

10.1 Stratified Nested CV tabular results

10.1.1 ANN

Fold	Model	AUC	Bal.Accuracy	F1	Training time (s)
1	ANN	0.703	0.655	0.837	33.0
2	ANN	0.626	0.569	0.814	44.2
3	ANN	0.680	0.541	0.808	34.9
4	ANN	0.648	0.533	0.810	36.8
5	ANN	0.634	0.588	0.822	31.7
6	ANN	0.590	0.578	0.797	32.4
7	ANN	0.638	0.556	0.798	44.6
8	ANN	0.657	0.580	0.806	27.0
9	ANN	0.633	0.573	0.804	35.9
10	ANN	0.628	0.562	0.785	36.6
11	ANN	0.594	0.576	0.821	31.1
12	ANN	0.629	0.547	0.801	31.8
13	ANN	0.628	0.530	0.798	24.7
14	ANN	0.594	0.535	0.800	26.3
15	ANN	0.620	0.543	0.784	19.4
16	ANN	0.676	0.543	0.797	24.4
17	ANN	0.626	0.591	0.755	18.4
18	ANN	0.680	0.649	0.837	27.8
19	ANN	0.640	0.555	0.817	36.3
20	ANN	0.689	0.629	0.844	37.5

Table 3: Metrics measures in each test fold (n = 20) for the ANN

10.1.2 CNN

Fold	Model	AUC	Bal.Accuracy	F1	Training time (s)
1	CNN	0.663	0.493	0.787	249.
2	CNN	0.768	0.496	0.790	120.
3	CNN	0.743	0.590	0.826	122.
4	CNN	0.590	0.500	0.794	94.6
5	CNN	0.680	0.556	0.805	298.
6	CNN	0.758	0.598	0.842	120.
7	CNN	0.732	0.507	0.795	263.
8	CNN	0.668	0.509	0.796	147.
9	CNN	0.751	0.500	0.790	82.8
10	CNN	0.750	0.518	0.799	107.
11	CNN	0.659	0.565	0.781	90.8
12	CNN	0.765	0.565	0.822	296.
13	CNN	0.749	0.675	0.786	112.
14	CNN	0.727	0.506	0.797	181.
15	CNN	0.740	0.611	0.813	377.
16	CNN	0.750	0.576	0.823	225.
17	CNN	0.709	0.578	0.818	168.
18	CNN	0.640	0.541	0.817	82.7
19	CNN	0.554	0.500	0.793	143.
20	CNN	0.539	0.500	0.793	209.

Table 4: Metrics measures in each test fold (n = 20) for the CNN

References

[1] Beaulac Cédric Moindjié Issam-Ali and Descary Marie-Hélène. *A functional approach for curve alignment and shape analysis*. 2025.

[2] Nobuyuki Otsu. "A Threshold Selection Method from Gray-Level Histograms". In: *IEEE Transactions on Systems, Man, and Cybernetics* 9.1 (1979), pp. 62–66. DOI: 10.1109/TSMC.1979.4310076.

[3] Kerem Gencer. "Enhanced Skin Lesion Classification through Hair Artifact Removal and Transfer Learning Models". In: *Black Sea Journal of Engineering and Science* 8 (2025), pp. 1007–1021. DOI: 10.34248/bsengineering.1665621.

[4] Alexandru Telea. "An Image Inpainting Technique Based on the Fast Marching Method". In: *Journal of Graphics Tools* 9.1 (2004), pp. 25–36. DOI: 10.1080/10867651.2004.10487596. URL: https://doi.org/10.1080/10867651.2004.10487596.

[5] Bradley Efron. *Bootstrap Methods: Another Look at the Jackknife*. Springer, 1982.

[6] University of Virginia Library. *Bootstrap Estimates of Confidence Intervals*. Accessed: 2025-09-15. 2025. URL: https://library.virginia.edu/data/articles/bootstrap-estimates-of-confidence-intervals.

[7] J. O. Ramsay and B. W. Silverman. *Functional Data Analysis*. 2nd. New York, NY: Springer, 2005.