Université de Liège

Faculté des Sciences Appliquées

Département d'Électricité, Électronique et Informatique

Institut Montefiore

Machine Learning for Landmark Detection in Biomedical Applications

Thèse présentée par

Rémy Vandaele

Pour le titre de Docteur en Sciences
(orientation Informatique)

supervisée par Pierre Geurts

co-supervisée par Raphaël Marée

Année académique 2018-2019

Abstract

Machine Learning aims at developing models able to accurately predict an output variable given the value of some input variables by using a dataset of observed (input, output) pairs. In the recent years, the development of new Machine Learning algorithms as well as the increase of computing capabilities have made these methods very popular and successful to address various image processing related tasks.

One of these tasks is landmark detection, which consists in finding the coordinates of one or several interest points in images. Landmark detection finds many applications in computer vision. In this thesis, we focus on two of them, both related to bioimaging. The first is morphometrics, where landmark coordinates are used to measure the size and the shape of body parts. The second is image registration, where the coordinates of the landmarks are used to compute the deformation between two images.

During this thesis, we have developed an automated landmark detection algorithm combining tree-based machine learning models with multi-resolution pixel descriptors. Starting from an algorithm used for cephalometric landmark detection, we have progressively extended it in order to fit the needs of morphometric analyzes, where a wide variety of image datasets and body types are observed. We carefully analyzed the behavior of our algorithm in order to provide detailed insights about its performance on new image datasets. We then extended our landmark detection algorithm to 3D images and used it to perform CT-CBCT rigid registration. Finally, we studied the relevance of using post-processing steps based on the landmark shape structure given the specificities of biomedical applications.

Throughout this work, we evaluated our method on four different datasets: three datasets concerning 2D morphometrics, and one concerning 3D image registration. On these datasets, we showed that our algorithm could reach state of the art performance while providing additional genericity regarding its application on datasets containing different types of images.

Acknowledgements

I would like to thank Pierre Geurts and Raphaël Marée for their precious support. Their knowledge and advices guided me through my research. With their dedication and incredible patience, they became an essential part in the realization of this work.

This thesis was started with a FNRS-Télévie grant. I would like to thank the organization for having offered me this great opportunity.

I would like to thank Philippe Martinive, Akos Gulyban, Philippe Coucke and Sébastien Jodogne for helping me to connect the dots between two different worlds. It was difficult, but the challenge was worth it and could not have been completed without their assistance.

Many thanks to my colleagues, from both Cytomine and the Montefiore Research Unit, for their help and insightful discussions.

The computations realized during this thesis would still be running without the efficient work of the teams managing the computing infrastructures of the SEGI and the CÉCI. Special thanks to Alain Empain for his help.

Thanks to the thesis committee for their evaluation of this work.

Finally, I would like to thank my family and friends for their support and understanding. Their presence during the good and bad times was key. Thanks to my brother Arnaud for his advices and careful reading of this manuscript.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Imaging techniques are a widespread technology in many different fields of science. They are crucial for the analysis of bodies and objects of interest: they allow to make accurate observations that are potentially impossible to carry out without such tools or techniques. For example, environmental scientists study multi-spectral images to analyze the effects of global warming, mechanical engineers study the quality of their equipment through computed tomography (CT) scans, biological experts use imaging techniques to study the development of microscopic bodies, and oncologists use PET-scans to detect potential tumors lying inside their patients. With the increase of computer performances bringing more and more images with always higher resolution and additional informations to extract, it became a necessity to use automated image processing techniques in order to replace or assist the experts in the process of their analysis. The types of tasks that can be automated are as various as there are research fields: it goes from image classification, to object recognition and segmentation. In this thesis, we focus on one of these particular image processing tasks: the automated detection of anatomical landmarks for medical and biological applications.

1.1 Anatomical Landmark Detection

1.1.1 Presentation of the problem

In the remaining of this work, we use the term **anatomical landmark** to describe a position in an image or in a volume. This position corresponds to a specific location in a particular body displayed in the image. A landmark is defined by a name (usually the anatomical name of the body location), and 2D (x, y) or 3D (x, y, z) coordinates in the image. In general, several landmarks are annotated on a single image.

An example is given in Figure 1.1 where the top left corner is considered as the origin $(0, 0)$. Three landmarks have been annotated on the body of the fox: (1) the tip of its nose, located at coordinates $(737, 247)$, (2) the upper corner of its right ear, located at coordinates $(682, 119)$, and (3) the upper corner of its right ear, located at coordinates $(790, 120)$. It

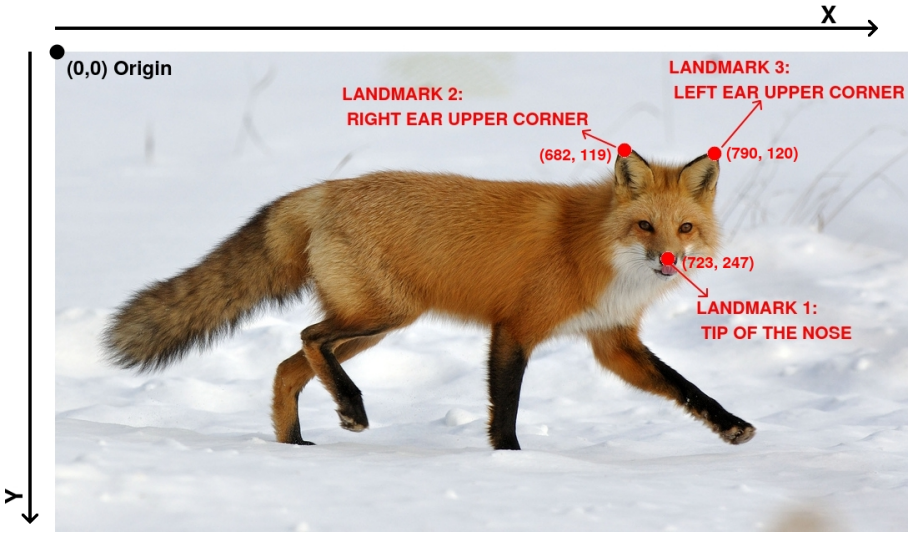


Figure 1.1: Three landmarks on the body of a fox.

is important to stress the difference between *anatomical landmarks* (as in Figure 1.1) and *fiducial landmarks* which are markers consisting of physical objects purposely placed in order to highlight a specific location by responding to the imaging technique used.

For a specific landmark, given its name, the process of **landmark detection** consists in finding its coordinates in the image. The task of detecting several landmarks on a new image can be performed quite easily and accurately by a human operator: with some typical examples, or just by using his prior knowledge, a human will be able to easily detect the landmarks on a new image. However, this easy task quickly becomes unmanageable with the increase of the number of images, their size as well as the number of landmarks to detect. When large datasets of images and landmarks are used, it becomes less and less realistic for a human to perform this task in a completely manual fashion. Not only this task can take precious time from an expert who could instead focus on more interesting matters, but the redundancy also adds to the painfulness of the task. These difficulties can be at the origin of a decrease of the accuracy in the detection, or even lead to confusion between the different landmarks to annotate. This is the reason why the development of (semi-)automated landmark detection algorithms is of great importance.

1.1.2 Applications

Landmark detection is useful in different fields. In this section, we present the three main applications using landmark detection, illustrated in Figure 1.2.

- In **video and image recognition**, landmarks are annotated on human faces. These landmarks are then detected and used in order to describe the given faces and help in the recognition procedure. This topic is highly popular due to the accessibility of the datasets, with large number of annotated images. As we show in Figure 1.2

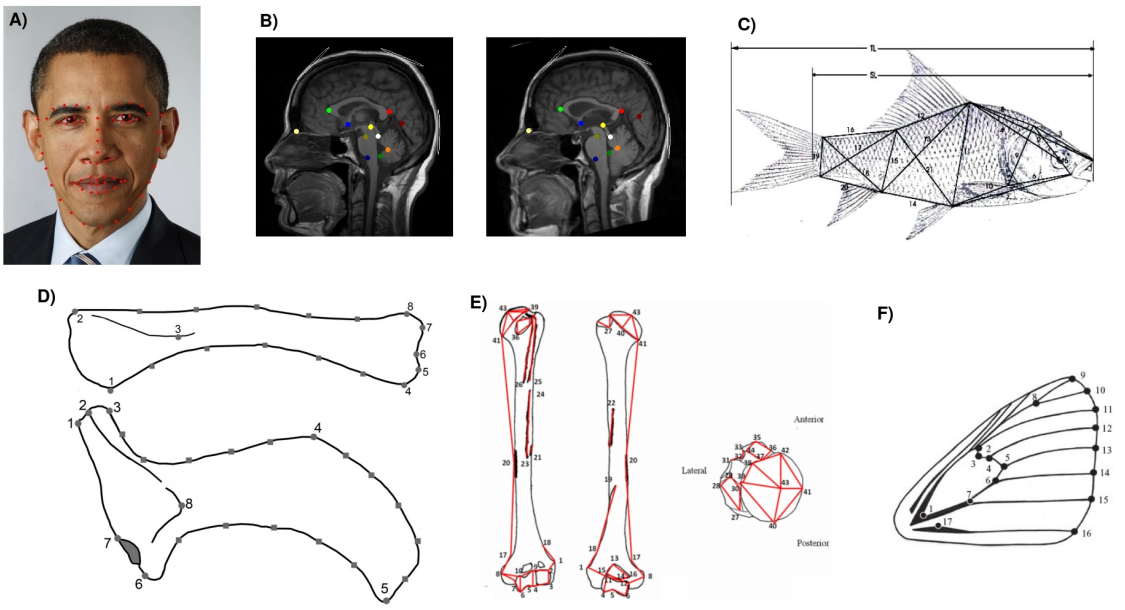


Figure 1.2: Landmark detection for A) Face recognition B) Image registration and Morphometric analysis on C) Fish images, D) Dinosaurs skeletons, E) Human humeri, F) Butterfly wings.

A), typical datasets use an important number of landmarks in order to be able to segment the face in the image.

- Landmark detection can also be used in **image registration**. Image registration is the problem of registering images in the same coordinate system. It therefore requires to find the deformation between the images to register. This problem can be solved with landmark detection by computing the deformation with the help of the coordinates of corresponding anatomical landmarks. An example is given in Figure 1.2 B), where the deformation between the corresponding landmarks could be computed, and applied in order to register the images.

- Landmark detection is also used in (bio-)**morphometrics**. Morphometrics is a set of analysis performed in several areas in which the morphology of a given body is studied through the analysis of distances, angles and shapes inside the body of interest. In order to perform these measurements, landmarks need to be annotated on the bodies that are studied. Morphometrics is widely used in applications related to biological studies. In Figure 1.2 C) to F), we present some morphometric applications: morphometric analysis can be used to study the impact of drugs on classical bodies (such as zebrafishes), extract digging rules from the study of dinosaur bones, investigate the morphology of neanderthal's humeri, and even monitor the evolution of butterflies.

The focus of this work is on medical and biological applications, which means morphometric

studies as well as image registration.

1.1.3 Landmark detection by supervised learning

The automated detection of landmarks is not an easy task for an algorithm. Indeed, given the body and the images on which the algorithms need to be applied, the visual appearances of the landmarks can be very different. For example, the conditions in which the images are taken can change the illumination in the image. The orientation of the body can also vary, and thus the orientation and the position of the landmarks. This variation can also come directly from the differences in the morphology of the studied body. In the example presented in Figure 1.1, we could try to detect the three landmarks on adult and young foxes, but they might have different types and shapes of ears and noses. It is therefore not easy for a developer to establish an algorithm with a set of rules that would be able to directly find the landmarks on each image. A natural approach to deal with such situations is the use of methods able to automatically generate this set of decision rules using a dataset of manually annotated images. Supervised machine learning focus on such methods.

Let X be the set of observations (input space), and Y be the corresponding set of annotations (output space). Supervised machine learning is a term used to describe the set of methods that, from a dataset made of N observations coming from X and their annotations in Y, computes a function $f : X \to Y$ able to predict an output (annotation) $y \in Y$ from an input $x \in X$ with the highest possible accuracy. If the output space Y consists of categorical values (unordered), the method can also be described as a classification algorithm. If not, it can be described as a regression algorithm. Supervised machine learning algorithms consider that the input space X consists of continuous and/or categorical values.

There are many approaches that are studied in supervised machine learning algorithms: some use a (non-)linear model, where the weights of the models are found by solving an optimization problem on the training data (Linear Regression, Support Vector Machines, Neural Networks), some others try to divide the space of the observations into subregions sharing the same type of output (C4.5, Random Forests), while some others simply use proximity criterions to decide which are the most relevant observations in the training dataset to get information from (Nearest Neighbors).

Supervised Machine Learning methods could potentially be used in most areas as long as observations and their corresponding outputs can be modeled using numbers or categories. The applications of these methods range from the prediction of equipment failures from diverse statistics (energy consumption, usage, location,...) to the recognition of speech from audio signals or to the diagnosis of potential diseases from DNA sequences or gene expression measurements.

It is in particular widely used in the area of image processing: supervised learning methods are developed in a wide range of applications going from recognizing characters or numbers [56, 49] to automated industrial inspection [52] or face recognition [2, 92]. It has also been

applied to applications close to the automated landmark detection problem [95, 80, 21], but with relative success, due to lacks of genericity, accuracy and prediction speed.

1.2 Objectives

Recent studies [64] showed that machine learning approaches based on the classification of sub-windows using Random Forests [12, 38] can bring additional genericity to automated image classification methods. It means that with this kind of methods, it is not necessary to make assumptions on the type of images on which the method is applied and that hyper-parameter tunings can be reduced to a minimum.

Given that we want to apply our method on datasets of different types, our first goal in this thesis is to analyze the application of a similar type of approach in order to build a robust *visual model* [1] able to automatically detect each of the landmarks using the available visual information. This model will not make any prior assumptions about the visual appearances of the landmarks or about their spatial configuration.

The second goal of this thesis is to test different approaches aiming at correcting the landmark positions predicted by a fully visual model by using models of the structure formed by the landmark positions. Indeed, one of the founding hypothesis of this work is that the *landmark structure* [2] formed in typical biomedical applications is harder to model, when compared to popular face detection applications, due to the scarcity of available annotated data and to the smaller number of landmarks that define the structure.

Although we will evaluate our methods on specific datasets, we will develop them so that they are as generic as possible and can be used for other types of data (like different types of morphometric studies or image registration problems). This genericity can be expressed in two different ways:

- The local visual information around the landmarks can vary: no assumptions are made about the physical location of the landmark.
- Potential users need to be able to autonomously use the landmarks detection methods on their own datasets. In particular, they need to know how the methods need to be tuned accordingly. They should also know what are the potential solutions (e.g., adding more manual annotations or landmarks or decreasing image variability) to improve the performance of the algorithm.

Note that, although no hypothesis will be made on the visual appearance of the landmarks, we will nevertheless assume that the images are already roughly registered, both in terms of scaling and orientation. To achieve the second goal, we will put some effort throughout

[1] In this work, we use the term **visual model** to denote landmark detection models that only exploit visual features to predict the position of each landmark individually, independently of the other landmarks and their relative positions.

[2] In this work, we use the term **landmark structure** to denote all statistical or geometrical relationships that exist between the different landmark positions and that could be exploited to improve landmark predictions.

the thesis on the systematic exploration of the influence of the different method parameters and we will also carry out experiments on different types of datasets. Finally, to make our methods accessible to everyone, we will implement them within the Cytomine platform, an open-source web-based image processing software.

1.3 Contributions

During this thesis, we proposed several contributions regarding biomedical imaging research in automated landmark detection:

- We participated in the 2014 ISBI challenge of cephalometric landmark detection with a preliminary version of our visual detection algorithm, presented in Chapter 3. With this version, we ranked #2 over nine teams. This participation resulted in the publication of a supplementary material [87] to the co-authored paper with the competing methods [91] and a presentation of the method during the corresponding 2014 ISBI Workshop [3].
  - **R. Vandaele**, *R. Marée, S. Jodogne, and P. Geurts.* Automatic cephalometric x-ray landmark detection challenge 2014: A tree-based algorithm. *Proceedings of the ISBI International Symposium on Biomedical Imaging, Automatic Cephalometric X-Ray Landmark Detection Challenge, 2014.* [87].
  - *C.-W. Wang, C.-T. Huang, M.-C. Hsieh, C.-H. Li, S.-W. Chang, W.-C. Li,* **R Vandaele**, *R. Maree, S. Jodogne, P. Geurts, C. Chen, G. Zheng, C.-W. Chu, H. Mirzaalian, G. Hamarneh, T. Vrtovec, and B. Ibragimov* Evaluation and comparison of anatomical landmark detection methods for cephalometric x-ray images: A grand challenge. *IEEE Transactions on Medical Imaging, 34(9):1890:1900, 2015.* [91].
- We improved the previous 2D landmark detection method, and extended it to morphometric studies by evaluating it on three different datasets, thus offering a more generic approach. A preliminary study was presented at the ICSIA 2015 conference, then a thorough study was published as a journal paper in Nature Scientific Reports [85].
  - **R. Vandaele**, *J. Aceto, M. Muller, F. Péronnet, V. Debat, C.-W. Wang, C.-T. Huang, S. Jodogne, P. Martinive, P. Geurts, and R. Marée.* Landmark detection in 2d bioimages for geometric morphometrics: a multi-resolution tree-based approach. *Scientific Reports , 8(1):538, 1 2018.* [85].
- We implemented our method as well as two other 2D landmark detection methods in the bio-imaging Cytomine software [4], published in Bioinformatics [66]. This implementation is open-source, and the software is available to everyone interested in landmark detection.

[3] `http://www-o.ntust.edu.tw/~cweiwang/ISBI2015/challenge1/`
[4] `http://www.cytomine.be`

  - *R. Marée, L. Rollus, B. Stévens, R. Hoyoux, G. Louppe,* **R. Vandaele**, *J.-M. Begon, P. Kainz, P. Geurts, and L. Wehenkel.* Collaborative analysis of multi-gigapixel imaging data using cytomine. *Bioinformatics , 2016.* [66].

- We extended our method to 3D images for CT-CBCT registration. Initially, we presented a poster at the 2015 ESTRO Conference. A later study using additional data was presented at the 2017 VISAPP conference, where it was published in the proceedings of the conference [86]

  - **R. Vandaele**, *F. Lallemand, P. Martinive, A. Gulyban, S. Jodogne, P. Coucke, P. Geurts, and R. Marée.* Automated multimodal volume registration based on supervised 3d anatomical landmark detection. *In Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP) , 2017.* [86] .

The latest research results presented in this document regarding the study and comparison of post-processing methods using the landmark structure (Chapter 6) is under consideration for a future publication. This publication could also include a study concerning the use of this method in a semi-supervised context.

1.4 Outline of the thesis

The remaining of this thesis is structured in six chapters as follows:

- **Chapter 2** presents the main principles underlying machine learning and state of the art methods in both machine learning and landmark detection.

- **Chapter 3** describe a preliminary version of our algorithm, developed in the context of the 2014 ISBI challenge for landmark detection on cephalometric images. This first version introduces the concept of multi-resolution features for landmark detection.

- **Chapter 4** presents the final version of our 2D visual landmark detection algorithm for morphometric studies, applied on three different datasets of images, and provides a complete analysis of its parameters in order to help potential users to apply it on their own datasets.

- **Chapter 5** extends the method presented in Chapter 4 to 3D images in the context of image registration. We also compare this image registration extension to state of the art image registration methods.

- **Chapter 6** presents advanced analysis and extensions to our 2D landmark detection method. It is divided in three parts: First, we study the impact of the number of training images on algorithm performances. Second, we study post-processing methods that refine the landmark positions initially predicted by the visual model. Finally, the last part of the chapter studies the possible intervention of human observers during the automatic annotation process.

- **Chapter 7** is the conclusion chapter. It offers some insights about the future work that could be done in order to further improve our method. We also offer some perspectives about the use of deep neural network approaches for landmark detection, a method that is currently offering interesting results in most areas where machine learning is used, especially in computer vision.

Chapter 2

Supervised learning and landmark detection methods

In this chapter, we review the main concepts underlying the studies presented in this thesis. First, the different notions concerning the domain of supervised machine learning are presented in Section 2.1. Secondly, we introduce in Section 2.2 the machine learning algorithms used in order to develop our landmark detection algorithm. We also make a brief description of the multi-layer perceptrons. After a brief introduction on supervised machine learning for image processing tasks in Section 2.3, we review the main landmark detection algorithms that existed prior to this work in Section 2.4, and then we detail in Section 2.5 two algorithms that were developed alongside this work. These algorithms, based on machine learning, were analyzed and reimplemented for the studies and comparisons performed in the following chapters.

2.1 Supervised machine learning

Machine Learning is a research field in artificial intelligence that focuses on the development of techniques and algorithms that can make a computer learn from data. These algorithms and techniques are based on theories coming from the fields of mathematics, statistics and computer sciences. It is closely related to **data mining**, which uses machine learning tools to analyze datasets.

The methods developed in machine learning are divided into two main categories: supervised and unsupervised. **Unsupervised machine learning** focus on developing methods for finding common patterns, groups and/or outliers in the datasets. **Supervised machine learning** uses datasets made of pairs (observation, output). The goal is then to produce, from these datasets, a model able to predict the output of a new observation with the highest possible confidence. As we explained in Chapter 1, the landmark detection methods presented in this thesis are based on supervised machine learning.

A toy example of a supervised machine learning dataset is given in Table 2.1. In this sce-

	Observation				Output
	Temperature	Tension	BMI	Heart rate	Has the flu?
#1	34	16	17	70	Yes
#2	38	12	21	89	Yes
#3	37	10	25	76	No
#4	35	15	35	95	No
#5	38	13	28	80	Yes

Table 2.1: Toy example of a supervised machine learning dataset: patient data for flu prediction

nario, we have five pairs (observation, output). One observation corresponds to a patient, and its output corresponds to a binary answer, if he has the flu or not. The patient is described using four measurements: his temperature, tension, BMI (Body Mass Index) and heart rate. By applying a supervised machine learning on this dataset, we want to create a model able to predict if a new patient described by these four measurements has the flu, with the highest confidence possible. In machine learning, the measurements of an observation are usually described using the term **descriptors** or **features**.

Supervised machine learning is itself divided into two main categories, depending on the type of the output of the dataset. If the output is an unordered category (this is the case for the example presented in Table 2.1), we can use the term **(supervised) classification**. In the case where the output is an ordered value (in Table 2.1 the output could for example be replaced with the severity of the flu), then the term **(supervised) regression** is used.

With the increase of data storage and computing capacities, datasets can consist of hundreds of thousands of observations, and be characterized by thousands of features. Human operators are generally not able to have a global understanding of the behavior of the datasets with such quantities of data. They are therefore assisted by machine learning algorithms. These algorithms can be used in two contexts. The first one is to automate a prediction task, which would mean predicting if a patient has the flu in our toy example. The second one is to use the model in order to understand which are the most relevant features for the prediction, which in our toy example would mean finding the most relevant symptoms of the flu.

The concepts of supervised machine learning can be described more formally. A typical dataset of observations X with its annotations Y is described in Equation 2.1. In machine learning, this dataset is also called **learning set or LS**. This dataset consists of N pairs (observation, output). The observations are described by vectors $x_i \in \mathbb{R}^m$ composed of m continuous feature values. Although this continuity of the features is not assumed in every supervised machine learning algorithm, this assumption will stay true along this thesis. For some cases, the corresponding output values y_i of the observations can also belong to a multi-dimensional space C. In supervised machine learning, the aim is to find a function

$f : \mathbb{R}^m \to C$ minimizing the error of the prediction of new observations.

$$X = \{x_i\}, Y = \{y_i\}, i \in [1, N], x_i \in \mathbb{R}^m, y_i \in C \tag{2.1}$$

However, it is complicated to directly estimate the quality of a supervised machine learning model for new observations. This is even the main problem that a machine learning algorithm has to tackle, and it is called the **bias-variance trade-off**. If a model focuses too much on minimizing the prediction error on the learning set, the risk is that the model will also fit random noise present in the learning set: outliers (exceptional observation), measurement variations or mistakes. The model created will then be unable to correctly generalize on new observations and will give unstable predictions on new observations (variance). This problem is called **overfitting**. At the opposite, if the machine learning algorithm does not focus enough on reducing the error on the learning set, it will not be able to find the best model of the problem, and will give inaccurate predictions (bias).

With our toy example, we can create a simple model classifying patients as not having the flu only if they have a temperature between 35° and 37°. While this model works on our dataset, we can easily suspect that it can not generalize to new patients: other sicknesses can be the cause of the temperature, and other symptoms can highlight a flu. Because this model is able to correctly classify the learning set, but is not generalizable, this model is described as having a high variance.

It is also quite simple to build a model with high bias: in our toy example, we can define a model that outputs the most represented category in the learning set: people with a flu. While this model will predict new patients with a non-zero accuracy, it will have learned nothing from the patient's features, and will be unable to accurately predict if a patient is sick or not.

While this problematic trade-off is approached differently by the different families of machine learning algorithms, the potential overfitting of a model also needs to be directly treated at the **validation** of the parameters of the model. For most machine learning algorithms, it is a bad choice to test the accuracy of a model with the observations used to build it. Indeed, the model is likely to overfit the learning sample and estimating its accuracy on this sample will give an optimistic estimation. This is why the learning sets are generally divided into three partitions:

- The machine learning model is built using the **training set**.
- The different parameters of the model are chosen by optimizing its performance on the **validation set**.
- The quality of the model built using the training set, with its parameters validated and tuned over the validation set is finally evaluated over the **test set**. This data, that were not used for building the model, nor to validate its parameters, thus minimizes potential overfitting at the evaluation of the model.

However, one could argue that the parameters of the model are then overfitted on the validation set: indeed, the parameters are tuned in order to work well on this dataset.

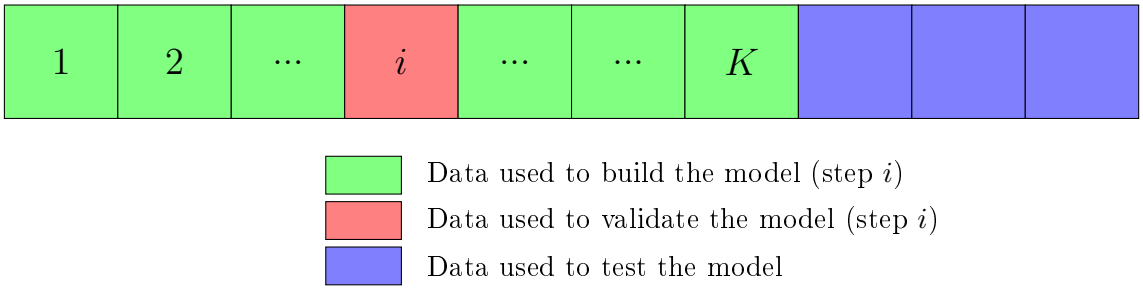


Figure 2.1: Visual representation of a step of a K-fold cross-validation methodology.

This is why a more reliable approach is generally used: **K fold cross-validation**. A step of cross validation is represented in Figure 2.1. In K fold cross-validation, K models are built during K different steps. Training and validation sets are merged and then divided into K partitions. At the i^{th} step, with $i \in [1, K]$, a model is built using all the partitions except partition i. This partition i is used to test the error of the model. The process is repeated K times, for each partition. The final validation error will then be the averaged error obtained over the K cross validation steps. When K is equal to the number of observations (i.e, one model is built per observation), this process can also be referred to as **leave-one-out** cross-validation.

Several families of algorithms exist to tackle the challenges of supervised machine learning. The family of (ensembles of) decision trees methods aims to create decision trees dividing the m-dimensional space defined by the features into subspaces sharing similar outputs. Overfitting and bias are then controlled by pruning the tree, by carefully choosing the decisions in the tree, and by using ensemble of trees. Support Vector Machines (SVMs) try to find the position for a (non-)linear border between the classes over the feature space giving the best bias-variance trade-off by maximizing its distance to the observations that are the closest, where this distance is called the classification margin. Other algorithms try to optimize the weights of a (non-)linear sum over the different feature values in order to minimize the difference between the sum and the outputs. The bias variance trade-off is controlled by the minimization of the number of weights and their relative importance. The k Nearest Neighbor algorithm simply tries to find the most similar observations in the dataset, and control the trade-off by considering the number of observations taken into consideration at prediction. In the next section, we briefly describe the two main families of machine learning methods in more details.

2.2 Machine Learning methods

Machine learning is an active research topic since the middle of the 20th century, and since then many supervised machine learning methods were developed along the years. In this work, we mainly focus on one family of these methods: the ensembles of decision trees. Indeed, as explained in Chapter 1, these methods are known for their good performances

in image processing tasks. After a description of this family of methods, we also give a description of the most popular family of machine learning methods today: the multi-layer perceptrons.

2.2.1 Decision tree

The decisions trees are a widely used technique, not only in the context of supervised machine learning. As shown in Figure 2.2 (Left), the principle behind a decision tree is to follow a tree of rules with binary answers in order to come up with a final decision at one of the leaves of the tree (terminal node). Given a defined feature space and a set of rules based on inequality tests, a tree also corresponds to a partitioning of the feature space (see also Figure 2.2).

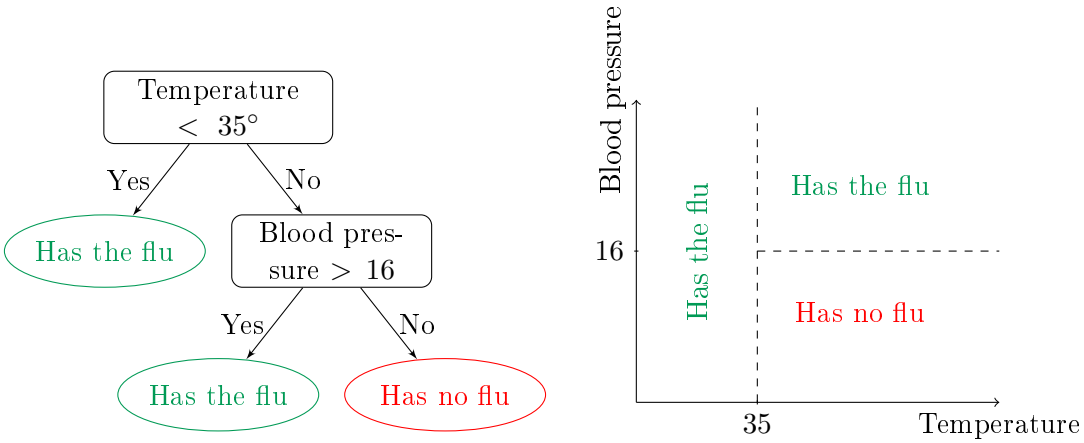


Figure 2.2: Left:Toy example of a decision tree. Right: Spatial representation of the decision tree.

In supervised machine learning, these decision trees can be built by using the learning set. By choosing rules with binary answers, consisting of inequality tests in the feature space of the observations (e.g Tension< 16), the different leaves of the trees will correspond to a partitioning of the learning set. By using supervised machine learning, these rules can be set in order to regroup the observations sharing similar outputs in the same partitions. More formally, if we assume that $X(i, j)$ is the numerical value of the j^{th} feature of the i^{th} observation, and y_i the output of this observation, we can define a generic function for training a decision tree, presented in Algorithm 1.

Given the node decision rules, the output of a new observation is computed by propagating it in the tree using its features, from the root node of the tree to one of the leaf nodes. The observation will then be assigned to the label of the corresponding leaf node. More formally, given a tree T and a new observation characterized by a vector of its feature values x, we can define another function for predicting an observation from a trained decision tree. This is presented in Algorithm 2.

In Algorithm 1, it remains to define the function `find_split`, which tries to find the best

Algorithm 1 `train_tree_from_data(X,y)`

if $\mathrm{Var}(y) = 0$ **then**
  label $\leftarrow y_0$
else
  $(k, v) \leftarrow$ `find_split(X,y)`
  $X_l \leftarrow \{X(i) | X(i,k) < v\}$
  $y_l \leftarrow \{y(i) | X(i,k) < v\}$
  $X_r \leftarrow \{X(i) | X(i,k) \geq v\}$
  $y_r \leftarrow \{y(i) | X(i,k) \geq v\}$
  $T_l \leftarrow$ `train_tree_from_data`(X_l, y_l)
  $T_r \leftarrow$ `train_tree_from_data`(X_r, y_r)
end if

Algorithm 2 `predict_output_with_tree(T,x)`

if T is a leaf **then**
  return T.`label`
else
  if $x(T.k) < T.v$ **then**
    return `predict_output_with_tree`$(T.T_l, x)$
  else
    return `predict_output_with_tree`$(T.T_r, x)$
  end if
end if

rule to use to partition the learning sample examples that have reached a given node. This best rule is often obtained by the computation of a quality score for each possible split at this node and by the selection among them of the split that maximizes this quality score. Typical quality scores measure the reduction of output *impurity* brought by the split, where output impurity is often measured by Shannon's entropy [13] or Gini index [72] in classification and by output variance [13] in the case of regression. More precisely, for a problem with c classes, Shannon's entropy and Gini index are computed respectively as follows for a given vector y of output classes:

$$\mathrm{Entropy}(y) = -\sum_{i=1}^{c} f_i \log_2 f_i \tag{2.2}$$

$$\mathrm{Gini}(y) = 1 - \sum_{i=1}^{c} f_i^2 \tag{2.3}$$

$$\tag{2.4}$$

where f_i is the proportion of examples of the ith class in y, i.e., $f_i = (\frac{\#\{j | y_j = i\}}{\#y})$, with $\#y$ the size of y. In the case of regression, output variance is simply:

$$\mathrm{Var}(y) = \frac{1}{\#y} \sum_{i=1}^{\#y} ||y_i - \bar{y}||_2^2. \tag{2.5}$$

Denoting by $Imp(y)$ any one of these three impurity measures, the quality score $Q(y, s)$ of a split s that separates the output vector y into two new output vectors y_l and y_r is computed as follows:

$$Q(y,s) \quad = \quad Imp(y) - \frac{\#y_l}{\#y} Imp(y_l) - \frac{\#y_r}{\#y} Imp(y_r). \tag{2.6}$$

The score is thus the difference between the output impurity at the node to split and the average of the impurities at the left and right successors of this node, weighted by the proportion of examples that they contain. For the three aforementioned impurity measures, the resulting score is always positive and it is maximized when the output is constant in the two successor nodes created by the split, leading to a zero impurity after the split.

As described, Algorithm 1 only stops expanding a tree branch as soon as the output is constant for all examples in the leaf at the end of this branch. We say in this case that the tree is fully grown. In practice however, some pruning needs to be applied to avoid overfitting. Indeed, increasing too much the number of nodes reduces the learning sample error but might lead to an increase of variance and thus an increase of the generalization error [37]. Pruning reduces the size of a tree, with respect to a fully grown tree, by replacing some of its subtrees by leaf nodes. It can be carried out either a priori or a posteriori. In the first case, pruning is called pre-pruning and it merely consists in extending the condition in Line 1 of Algorithm 1 for stopping the expansion of a given node. For example, one can stop splitting a node either when its depth has exceeded some threshold, when the number of examples it contains is too small, or when the output impurity is below some threshold. When pruning is carried out a posteriori, it is called post-pruning. In post-pruning, a fully grown tree is first built using Algorithm 1 and then irrelevant subtrees within this tree are highlighted on the basis of a separate validation set and they are replaced by leaf nodes. As a result of pruning, some leaf nodes potentially contain examples whose outputs might be different. The label of these leaf nodes can then be computed as the average of those outputs (regression) or the most represented class (classification). These labelings respectively minimizes the mean squared error in regression and the error rate in classification as estimated on the training set.

Even when pruned, decision trees still suffer from a high variance, which makes them often not competitive with other supervised learning methods such as Support Vector Machines or Multi-Layer Perceptrons. One way to get rid of this variance is to use ensemble of decision trees instead of single trees. These methods are described in the next section.

2.2.2 Ensembles of decision trees

The principle behind ensemble methods in machine learning is to exploit the predictions of several different models built from the same training set in order to take a decision. Majority vote or averaging the outputs of these different models are classical approaches to combine them at prediction.

Ensembles of decision trees consist in combining multiple decision trees built from the same training set. In the previous section, we described a generic approach for building a

supervised decision tree, but its construction is deterministic: at each node, every possible splits are evaluated, and the best one is used for splitting the node. Using this approach in the context of ensembles of decision trees would therefore be useless, because every tree model would be the same, and have the exact same output for any observation. For this reason, several approaches, described in the following, have been proposed to randomize the selection of the splits and thereby generate different trees from the same training set. Because of the introduction of this randomization, a single randomized tree will be typically less good than a standard deterministic decision tree, both in terms of bias and variance. However, averaging the predictions of several randomized trees is expected to bring superior performances mainly because of a drastic reduction of variance with respect to standard deterministic decision trees. Randomization indeed makes the tree less dependent on the specific training set used, while averaging gets rid of the variance brought by the randomization.

One of the most popular algorithm to train an ensemble of decision trees is Breiman's Random Forests algorithm [12]. The selection procedure for a given node of the tree is detailed in Algorithm 3. Let us assume that at a given node, we have a set of N observations $x_i, i \in \{1, \dots, N\}$, characterized by their m features, and y_i represents the corresponding output. The algorithm first selects a random subset S of k features with replacement among the original m features, with k a user-defined parameter. Then, for each of the k selected features, the optimal split $v(i)$ is computed using the function `find_best_split`. The best split among the k is then finally selected for splitting the node (function `find_best_split_pair`). The best splits are found using the same quality scores used for building standard decision trees (see Section 2.2.1).

Algorithm 3 `find_split(X,y,k)`

1: $S \leftarrow$ set of k integers randomly drawn (without replacement) in $\{1, \dots, m\}$
2: **for** $i \in S$ **do**
3: $\quad v(i) \leftarrow$ `find_best_split(X,y,i)`
4: **end for**
5: **return** `find_best_split_pair(X,y,S,v)`

Another specificity of the Random Forest algorithm is that each tree of the ensemble is grown from a bootstrap sample of the original training set, as in *Bagging* [11]. A bootstrap sample is obtained by sampling N examples with replacement from the original training of size N. Each bootstrap sample thus potentially contains several copies of some training examples, while other training examples are missing. Bootstrap sampling introduces some further randomization of the trees that slightly increases bias (because of a reduction of the effective training set size) but leads to a stronger reduction of variance.

The Extremely Randomized Trees algorithm [38] uses an even more randomized approach for selecting a split at a given node during training. It modifies Algorithm 2.2.2 by replacing the selection of the best split for each of the k selected features (Line 3) by the selection instead of a random split. For a numerical feature, a random split is obtained by drawing a cut point uniformily at random between the mininum and the maximum values of the features in the current node. This modification with respect to Random Forest brings a

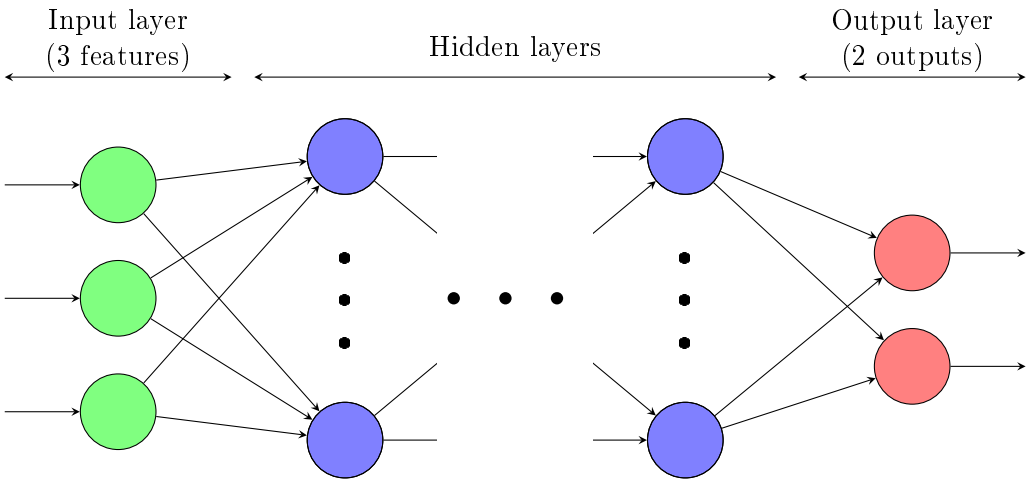


Figure 2.3: Visual representation of a neural network.

significant speed-up when training the model (since the optimization of the split is not needed anymore). It also leads to a stronger reduction of variance that makes the use of bootstrap sampling unnecessary [38]. Due to its ability for handling a large number of features, this algorithm is very popular for supervised machine learning tasks in image processing [64, 23] and it will also be the basis of our landmark detection solutions later in this thesis. The method shares the same hyper-parameters as Random Forest: k the number of features randomly selected at each node, T the number of trees in the ensemble, and n_{min}, the minimum number of examples for splitting a node. T has to be fixed to the highest possible value taking into account only computational considerations, n_{min} is usually set to 2 in the context of classification, and a common default value of k is $\sqrt{m}$, with m the total number of features [38].

2.2.3 Multi-Layer Perceptrons

The algorithm of multi-layer perceptrons [78] is among the most popular supervised machine learning algorithms of the last years. While it was introduced at the end of the eighties, its interest has raised in the recent years due to the advent of the more general domain of deep learning.

The principle of a multi-layer perceptron is to represent the relationship between the input observations and their outputs by using interconnected mathematical structures called neurons. As represented in Figure 2.3, neurons are organized into layers. At the exception of the first (input) layer, each neuron of a layer receives as inputs the outputs of the neurons of the previous layer. Except for the last (output) layer, the output of each neuron is connected to the inputs of the neurons of the next layer. More complex network structures, even with recurrent connections, are possible but they will not be considered here.

A neuron takes as input the output values $v_1, ..., v_J$ of the J neurons from the previous

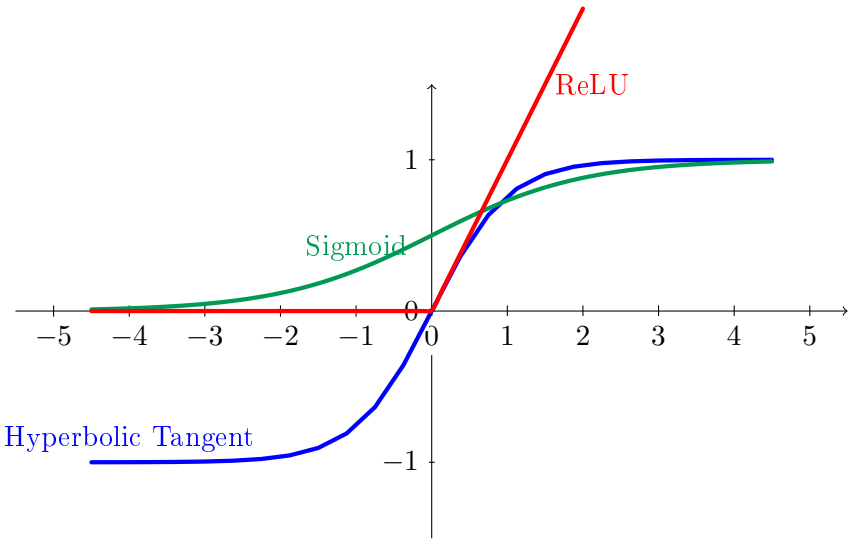


Figure 2.4: Typical activation functions used in a multilayer perceptron

layer, and outputs a single value computed as a weighted sum of its inputs sent through a non-linear activation function f, i.e., $f(\sum_{i=1}^{K} w_i v_i)$. Training the network consists in optimizing the weights so as to minimize a loss function computed on the training set. There are different functions commonly used as activation functions (see Figure 2.4): the ReLU (ramp identity), the hyperbolic tangent or the sigmoid function. These functions can potentially vary between networks, layers, and neurons. The use of non-linear activation functions makes the function computed by the network non linear and it allows multi-layer perceptrons to approximate every possible function, with a sufficient number of layers and neurons per layers [24].

We can distinguish three different types of layers: the first layer of a network is always the input layer. Each neuron of this layer corresponds to a specific input feature of an observation. There is no weight or activation function in this layer. Given an observation, the output of these neurons is the value of the feature corresponding to the neuron. The second type of layer is the hidden layer. There can be several hidden layers in a neural network. They are assigned weight values as well as an activation function, and their output corresponds to the non-linearized weighted sum described in the previous paragraph. The third and last type of layer is the output layer: it takes as input the outputs of the last hidden layer, and outputs the predicted output of the observation assigned at the input layer. In this layer, there are as many neurons as there are dimensions in the output space. The neurons in that layer also have weights and activation functions.

If we consider the global multi-layer perceptron as a function denoted $F(x; w) \in C$ where x is the input features of an observation and w is the weight vector of all the neurons of every layer, the goal of the training phase is to find the weights w minimizing the average over the training set of a loss function comparing the predictions made through F on each training observation $x_i \in R^m$ and its corresponding real output value $y_i \in C$. This

minimization problem can be formulated as follows:

$$\min_{w} \frac{1}{N} \sum_{i=1}^{N} \text{loss}(y_i, F(w, x_i)) \tag{2.7}$$

In regression, a typical loss function is the squared error:

$$\text{loss}_{SE}(y, \hat{y}) \quad = \quad ||y - \hat{y}||_2^2, \tag{2.8}$$

where the output vectors can be multi-dimensional. In classification, a typical loss function is the cross entropy, which is written as:

$$\text{loss}_{CE}(y, \hat{y}) \quad = \quad -\sum_{j=1}^{c} y_j \log(\hat{y}_j) \tag{2.9}$$

where c is the number of classes, $y \in \mathbb{R}^c$ is the indicator vector of the true class (with $y_j = 1$ if j is the true class, 0 otherwise), and $\hat{y}_j$ is the probability estimated by the neural network that the example belongs to class j [1]. Algorithms used to solve problem (2.7) are generally based on gradient descent methodologies, benefiting from the fact that derivatives of the average loss can be computed efficiently using the backpropagation algorithm [56].

While multi-layer perceptrons have been used since the end of the eighties, they have regained important interest recently. New methodological developments and computing infrastructures (based on GPUs) have indeed been proposed that allow to significantly speed-up their training and also to train neural networks with increasing numbers of layers that reached impressive performance on very complex tasks at the heart of artificial intelligence. These advances have lead to the definition of a whole new domain called "deep learning" [39]. The success of these methods is also linked to the fact that the typical size of the databases available to train supervised learning models have very significantly increased, which is very crucial to be able to train very deep neural network architectures.

Some of these network topologies, such as the Convolutional Neural Networks [56] have proven to be extremely efficient in imaging tasks involving supervised machine learning [53, 76, 79]. Although they are not tested in this thesis, landmark detection methods could probably benefit from the latest advances in deep neural networks. This will be discussed in more details in the conclusion chapter of this thesis.

2.3 Images in supervised machine learning

Numerical images, in two or three dimensions, measure the intensity of electromagnetic waves coming from, or being reflected by different objects. Given the measured wavelengths, images can supply different types of informations (X-Ray imaging, echography,

[1] To ensure that the c network outputs correspond to proper probabilities, they are usually sent through a so called Softmax layer.

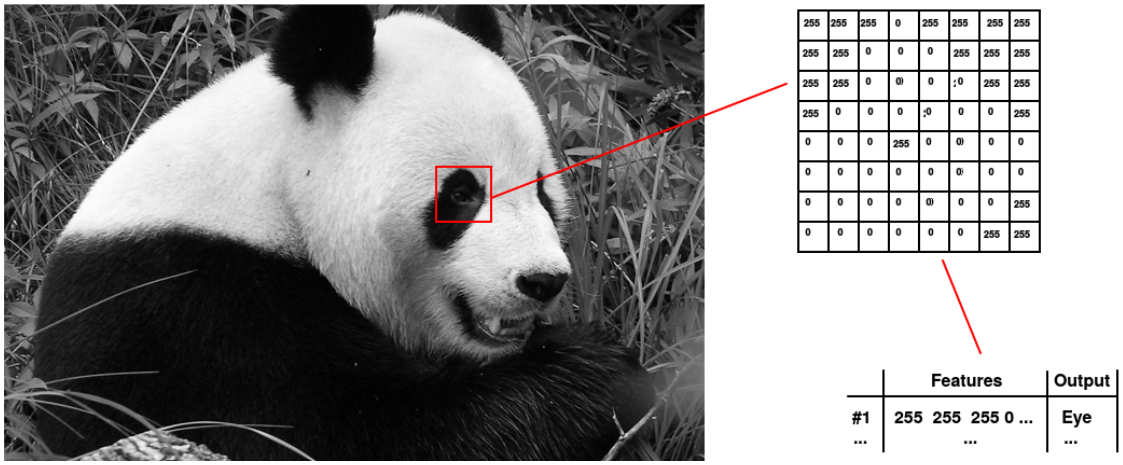


Figure 2.5: Description of an image using pixel values.

and photography are some examples). With numerical imaging, this intensity measurement is typically discretized in square pixels averaging the intensity over a given square resolution.

An image recording the intensity at a given wavelength can therefore be represented as a matrix of pixel intensity values. A classic RGB image, where the intensity at three different wavelengths is recorded, is represented by three matrices, one for each wavelength. A 3D image records the averaged intensity over cubic volumes, and their values are represented as cubes, also called voxels. Hence, 3D images correspond to three dimensional matrices.

Given that supervised machine learning tools use numerical features to describe an observation, these intensity values can be exploited as features for image related observations. An example is given in Figure 2.5, where the location in the image can for example be represented by their surrounding pixel values.

There are two main ways to describe an image from these values:

- Transform the raw intensity values into new values that will help the supervised machine learning algorithm to understand the data. This could help the algorithm to become less sensitive to potential deformations such as rotations, scaling, or illumination changes. A large number of such transformations (visual image descriptors) have been proposed in the literature. They help the algorithms to become more robust to deformations and/or highlight some of the image properties. For example, instead of using the raw intensity values, we could use a histogram of those values, an image descriptor insensitive to rotations. Gradient-based information (difference with intensity values of surrounding pixels) is another descriptor that can help to understand the variation of pixel intensity values around edges. More complex descriptors such as Local Binary Pattern or Speeded-Up Robust Features are classic examples commonly used in automated image processing [80, 2, 62].

- Directly use the raw intensity values as they are. The idea behind this approach is that if a sufficient amount of data is fed to the right supervised learning algorithm, the algorithm will be able to decide which are the most relevant relations between the intensity values to use. With the increase in size of both image datasets and computing capabilities, this approach has become increasingly popular in the recent years, especially with Random Forests and Convolutional Neural Networks, two types of methods able to take advantadge of these raw intensity values, given a sufficient amount of training data [64, 9, 53, 79].

In this thesis, we will explore these two alternatives by comparing raw intensity values with standard image descriptors from the literature, both fed as inputs to Random Forests models.

2.4 Landmark detection methods

In the literature, three different approaches have been proposed to tackle the problem of landmark detection.

The first approach, and also the most widely studied prior to the development of our algorithm, focuses on landmark detection for segmentation purposes (face or biomedical segmentation). The works coming from this area are mostly improvements or extensions of two important algorithms developed by Tim Cootes and his colleagues: the Active Shape Model (ASM) [22] and the Active Appearance Model (AAM) [21].

From an initial setup of landmark positions, the ASM method relies on the repetition of two iterative steps until convergence:

1. Update the landmark positions to close locations having better visual correspondence with the landmarks.
2. Update the landmark positions by fitting a model of the global landmark shape, learned from a dataset of manually annotated images.

This algorithm having been developed for segmentation purposes, the visual correspondence used by the initial algorithm in step 1 was based on the edginess of the pixel location. In segmentation tasks, landmarks are indeed generally placed at edge locations, denoting the extent of a given shape or object.

One drawback of the ASM algorithm is that it does not consider the texture of the shape object. The AAM algorithm aims at addressing this issue by taking this visual information into account. It uses an optimization process that tries to minimize the least-squares difference between the current shape model and its projection on the target image, taking into account the intensity values of the shape as well as the landmark positions. However, this algorithm needs a good initialization in order for the optimization procedure to work properly. As the algorithm only focuses on the shape texture, landmarks locations are less likely to converge towards edge positions, which makes it less efficient than the ASM method for segmentation problems.

Several papers have proposed to improve these two algorithms to address their own specific segmentation tasks. For example, Rogers *et al.* [75] proposed a shape model which is more adapted to the complexities of medical image segmentation problems. Duta and Sonka [29] derive ways to exploit prior knowledge about the pixel intensities and landmark positions to improve the initialization of the landmark position vector. Van Ginneken *et al.* [84] proposed to improve the search for better visual positions by using a kNN landmark classifier, and tested different image descriptors. This work was later pursued by Sukno *et al.* [81], who speeded up the classification by using a neural network and new image descriptors. Some authors have also applied Active Shape Models for the detection of cephalometric landmarks. For example, Yue *et al.* [95] proposed to divide cephalometric images into different subregions (using prior knowledge) on which they applied different Active Shape Models, one per region. Kafieh *et al.* [45] also use an Active Shape Model, where the initial landmark positions are found using a neural network classifier taking image sub-windows as inputs. More recently, Lindner and Cootes [59] proposed to build heat maps using a visual regression model, with the latter then exploited by the Active Shape Model.

The second approach tackles the problem of landmark detection with different ways to modelize and use the shape of the landmarks: Donner *et al.* [28] use a Markov Random Field in order to select the most likely combination of landmark candidates found by a refined classification model (this algorithm is described in more details in Section 2.5.1). Ibragimov *et al.* [43] propose a framework in which landmark positions are found by using a game-theoretic solution.

Lastly, **the third approach** proposes to perform the detection of the landmarks only on the basis of a visual model, without using shape models for correcting landmark positions. Along this line, Stern *et al.* [80] obtained promising results for detecting landmarks on Zebrafish images by using a random forest model to classify image sub-windows described by a combination of image descriptors. Kaur and Singh [48] used a two steps approach that uses Zernike moments in order to find the landmark positions on cephalometric images.

2.5 Selected Methods

The landmark detection approach that will be developed in this thesis is inspired by preliminary work of our group on this topic presented by Stern *et al.* in [80], which is representative of the third family of techniques presented in the previous section. Commonalities and differences with this latter method will be highlighted in Chapters 3 and 4. In this section, we describe in details two methods from the literature that we consider to be the most competitive for the purpose of anatomical landmark detection in biomedical applications. These two methods, like ours, exploit visual models based on Random Forests. The method by Lindner and Cootes [59] is one of the most recent and promising approaches using ASM for landmark detection. The method by Donner *et al.* [28] relies on a different shape model for correcting landmark positions based on Markov random field. Note that these two methods were actually developed in parallel and independently of our own

method. Because no implementations of these two methods are available publicly, we reimplemented them ourselves to carry out our comparative analyzes in Chapters 4 and 6. The purpose of our detailed description below is also to provide our own interpretation of the different steps of these methods, as all details are not always provided in the original publications. Minor differences might thus appear between our implementation and the original methods.

2.5.1 Donner *et al.*, 2013: Global localization of 3D anatomical structures by pre-filtered Hough Forests and discrete optimization

This first state of the art landmark detection approach was developed by Donner *et al.* in order to detect landmarks in both two-dimensional images and three-dimensional volumes [28]. Although it can be applied for both cases, our description focuses on two-dimensional images. This approach is divided into three different steps, each using a different machine learning model. These three steps of the algorithm are described below in turns. In Chapter 4, we will refer to this method as the **DMBL** method, where DMBL simply stands for the first letters of the last names of the authors of paper [28].

Step 1: Pixel classification

Description. Given a new image for which we want to predict the landmark positions, the first step of the algorithm consists in classifying each of its pixels using a classification model. Each pixel of the image is assigned one probability per landmark. This probability corresponds to the likelihood computed by the machine learning classification model that the position of the pixel corresponds to the position of the landmark.

Supervised machine learning model. The model used is a multi-class classification model: there is one class per landmark, and an additional class for the pixels that do not correspond to any landmarks. Random Forests are used as the machine learning algorithm. Classification Random Forests can output probabilities instead of a class: given that each tree of the Random Forest is casting a vote, dividing the total number of trees agreeing that an observation belongs to a given class by the total number of trees in the forest gives an estimate of the probability that the observation belongs to the given class.

Pixel descriptor. In this algorithm, the pixel of an image I located at position (x, y) is described by N features $F_i, i \in [1, N]$, computed using the difference between the pixel value $I(x, y)$ and pixel values located at random offsets $v_i \in \mathbb{R}^2$. These offsets are common to all the images and pixels extracted, at training and prediction. They are chosen using a gaussian function $\mathbb{N}(0, \Sigma)$. The process is showed in Equation 2.10.

$$F_i(I, x, y) = I(x, y) - I(x + v_i(0), y + v_i(1)), v_i \sim \mathbb{N}(0, \Sigma), \Sigma = \begin{pmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{pmatrix} \tag{2.10}$$

In this step, σ (the variance of the gaussian) and N (the number of features) are parameters that need to be tuned during the validation procedure.

Training. For a given training image I, all the pixels (x_j, y_j) within a radius R_1 from the landmark position $(x_{I,l}, y_{I,l})$ are extracted (the training algorithm computes their features), and assigned to the class l. The process is repeated for each landmark. Additionally, sampled pixel locations that were not yet extracted, and do not belong to any of the landmark classes, are extracted and assigned to the *not-landmark* class. The process is then repeated for each training image, and a Random Forest model is trained using this training dataset.

Output prediction. At prediction, each pixel (x, y) of the image is classified using the Random Forest model. In order to speed up the classification process, the images can be resized using a parameter δ. This resizing also affects the size of the images during the training process. For a predicted image, the output of this step will consist in one *probability image or vote map* V_l for each landmark l, where $V_l(x, y)$ corresponds to the probability of position (x, y) corresponding to the landmark position as computed by the model.

Step 2: Candidate selection

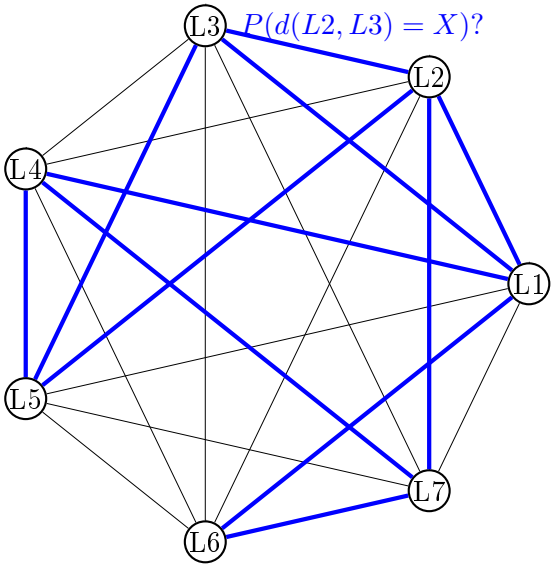
Description. This second step takes as input the vote maps of the first step. The goal of this second step is to refine the vote maps in order to select a given number of potential candidate positions for each of the landmarks. The idea is to build one regression model per landmark l. The model for landmark l learns the offset between a pixel position (x, y) and the position (x_l, y_l) of the landmark in the image, ie., $(x - x_l, y - y_l)$.

Supervised machine learning model. This is a multi-output regression problem (2 outputs). Regression Random Forests are used for the machine learning model.

Pixel descriptor. The same pixel descriptor as used during the first step of the algorithm (see Equation 2.10) is also used during this step.

Training. At training, L models are built, one for each of the L landmarks. For a landmark l and an image I, all the pixels within a radius R_2 from the landmark positions $(x_{I,l}, y_{I,l})$ are extracted, and their offset to the landmark position assigned as the regression output. This process is repeated for each of the training images.

Output prediction. At prediction, for a new image I with vote map $V_{I,l}$ for landmark l, the pixel offsets $(\Delta x_j, \Delta y_j)$ are computed using the trained model for each pixel position (x_j, y_j) such that $V_{I,l}(x_j, y_j) \geq \beta \max_{x',y'} V_{I,l}(x', y')$. The vote map is then updated using



#	Position	Pr.
1	(112,21)	0.12
2	(114,18)	0.11
...	...	...
Nc	...	...

Figure 2.6: Representation of the Markov random field for landmark detection, using 7 landmarks and two edges (the selected edges are in blue).

the offsets that were computed: $V_{I,l}(x_j - \Delta x_j, y_j - \Delta y_j) = V_{I,l}(x_j - \Delta x_j, y_j - \Delta y_j) + V_{I,l}(x_j, y_j)$, and the process can be repeated S times. Then, the vote map is reduced to its N_c best landmark positions.

The output of this step is thus a refined vote map, composed of at most N_c landmark positions per landmark.

Step 3: Finding the best combination of landmark candidates

Description. This third step takes as input a refined vote map, with at most N_c landmark positions per landmark. For a total number of L landmarks, there is therefore a possibility to select N_c^L different combinations of landmark positions. This third and last step will try to find the best combination of candidates using the combinatorial optimization procedure of a Markov Random Field.

Markov Random Fields. An example of a given Markov Random Field is represented in Figure 2.6. The idea is to find a combination of landmarks that maximizes both the landmark probabilities obtained during the second step of the algorithm, and probabilities concerning distance relationships between the landmarks. These distances probability functions are modeled using Gaussians, based on statistics of landmark distances (average and standard deviation) estimated in the training set. In order to reduce the number of edges to consider, the algorithm proposes to limit the consideration of distances to T edges per landmark. The selection is done by looking at the variations of the geometric distances between the landmarks. Differential entropy is used to estimate the reliability of

using a landmark position in order to predict the position of the other landmarks. For a given landmark, the T most predictive landmark pairs (edges) are then used in the Markov Random Field.

Candidate selection. Once the Markov Random Field modelization is performed using the training set, the landmark candidates for an image (obtained at step 2) can be integrated to the model. A belief propagation algorithm specific to Markov Random Fields [94] will then be able to find the optimal combination of candidates according to this model. The output of this third step will then be the best combination of landmark candidates that were found at step 2. This combination will be the final detected positions of the landmarks on the new image.

2.5.2 Lindner and Cootes, 2015: Fully automatic cephalometric evaluation using Random Forest regression-voting

The approach presented in this section uses a two steps methodology [59]. First, a pixel offset regression approach is used in order to build a vote map of the landmark positions. Second, the landmark detection algorithm tries to fit a shape of the landmark structure to the image by using the vote maps and a PCA model of the landmark structure. In Chapter 4, this method will be refered to as the **LC** method, again as a reference to the first letter of the last names of the authors of paper [59] (Lindner and Cootes).

Step 1: Pixel offset regression

Description. The idea of this first step is also to build a vote map for a new image. The main difference is that instead of a classification approach, a pixel offset regression approach is used. At prediction, pixels are extracted, and each one of them votes for the landmark location by using the offset predicted by a regression model. One model per landmark is created at training.

Supervised machine learning algorithms. Given that the output is an offset, this a multi-output regression problem. Random Regression Forests are used in order to build the model. One Random Forest model is used per landmark.

Training. At training, for landmark l, the pixels are extracted at positions (x_j, y_j) within a window $[-d_{\max}, d_{\max}]$ from the landmark position on each of the training images. Their regression output corresponds to the offset with respect to the real landmark position $(x_j - x_l, y_j - y_l)$.

Pixel descriptor. The pixels are described using a given number of Haar-Like features within a window of size w surrounding the pixel position. As presented in Figure 2.7,

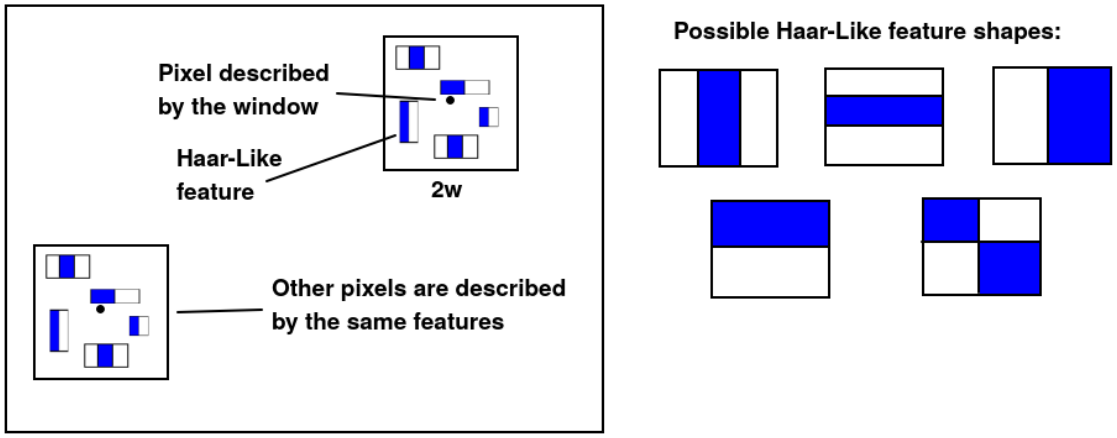


Figure 2.7: Haar-Like features used to represent a pixel. In order to compute the value of a haar-like feature, the sum of the pixels in the blue zones are subtracted to the sum of the pixel values in the white zones.

Haar-Like features correspond to the difference between sums of pixels within square or rectangular areas. They are all extracted at fixed random location inside a window of size $2w \times 2w$ around the landmark position.

Output prediction. At prediction, a mean shape is first computed to estimate the landmark position. In this case, this means that the position of the landmark will be estimated as the average position of the landmark in the training set. Then, each pixel in the window $[-d_{\max}, d_{\max}]$ around this position will be extracted, and the tree model will be used to build a vote map. For each pixel (x, y) extracted, each tree of the random forest outputs its offset (x_v, y_v) to the landmark position, and adds one vote to the position $(x-x_v, y-y_v)$ as being the position of the landmark. For a forest of T trees, $T \times (2d_{\max}+1)^2$ votes are therefore casted.

The process is repeated for each landmark, hence this first step will output a vote map for each of these landmarks.

Step 2: Shape fitting using PCA reduction

Description. The idea of this second step is to optimize the shape structure while considering the results obtained with the vote maps V_l of each of the landmarks l as well as the shape structure modeled from the training images using principal component analysis (PCA).

Algorithm. The landmark positions $(x(l), y(l))$ are initialized with the locations of the pixel positions which obtained the highest number of votes at step 1. Then, an iterative loop

is created. The first step of this loop corrects the landmark positions vector by changing the landmark positions to the closest positions making the structure an acceptable shape according to the shape model. The second step changes these new landmark positions to the location obtaining the highest number of votes within a radius r that is iteratively reduced by a factor β. The process is repeated until the radius has reached a minimal acceptable size (1 for example). This procedure is presented in Algorithm 4.

Algorithm 4 Shape-fitting description of the LC landmark detection algorithm

1: **for** l in $[1, L]$ **do**
2: $(x(l), y(l)) \leftarrow \arg\max_{x,y} V_l(x, y)$
3: **end for**
4: **while** $r > r_{\min}$ **do**
5: $(x, y) \leftarrow$ `correct_shape`(x, y)
6: **for** l in $[1, L]$ **do**
7: $(x(l), y(l)) \leftarrow \arg\max_{x,y} V_l(x, y)$ s.t $(x - x(l))^2 + (y - y(l))^2 \leq r^2$
8: **end for**
9: $r = r \times \beta$
10: **end while**

Shape correction algorithm. The shape correction step (line 5 of Algorithm 4) is performed using a PCA-based shape model. Let us denote by $x_i \in \mathbb{R}^{2L}$ the vector of landmark positions of the ith image of the training set, with $i = 1, \ldots, N$. The idea of PCA is to approximate each such vector as follows [37]:

$$x_i \approx \mu_x + Pb_i, \tag{2.11}$$

where $\mu_x \in \mathbb{R}^{2L}$ is a location vector, P is a $2L \times t$ projection matrix whose t columns are constrained to be orthogonal unit vectors, and $b_i \in \mathbb{R}^t$ is a t-dimensional representation of the ith image, with $t \leq 2L$ a user-defined parameter. Fitting the parameters of this model so as to minimize the following (least square) reconstruction error:

$$\sum_{i=1}^{N} ||x_i - \mu_x - Pb_i||^2, \tag{2.12}$$

leads to the following solution:

$$\mu_X = \frac{1}{N} \sum_{i=1}^{N} X_i \tag{2.13}$$

$$b_i = P^T(x_i - \mu_x), \tag{2.14}$$

$$\tag{2.15}$$

and P has its columns set as the t eigenvectors of highest eigenvalues of the covariance matrix of $\mathbf{X}$, the $N \times 2L$ matrix compiling the landmark positions of the N images in the training set.

Given μ_x and P as estimated from the training set, the shape correction algorithm will take as input a landmark position vector $x' \in \mathbb{R}^{2L}$ (concatenating the current x and y coordinates of the landmarks) and it will output a new vector x'' of corrected coordinates computed as follows:

$$x'' = \mu_x + P(P^T(x' - \mu_x)). \tag{2.16}$$

2.6 Conclusion

The aim of this chapter was to give a brief overview about the different topics discussed throughout this manuscript. In Section 2.1, we presented supervised learning and introduced some definitions and concepts commonly used in this domain. Section 2.2 introduced some of the most typical machine learning algorithms used today and Section 2.3 briefly discussed their use in image processing. For more details, we encourage the interested reader to refer to popular general textbooks on the subject of Machine Learning, such as [37] or [68]. Reference [23] is also an interesting read about the application of Random Forest based methods in image processing. In Section 2.4, we then presented existing landmark detection methods. Finally, we detailed in Section 2.5 two recent state-of-the-art landmark detection algorithms that we will compare empirically with our methods in the following chapters.

Chapter 3

A Landmark Detection Method for Cephalometric Studies

In this chapter, we present a first anatomical landmark detection method. This method is dedicated to the analysis of cephalometric images, and was presented during the Cephalometric X-Ray Landmark Detection Challenge at the 2014 ISBI Conference in Beijing. With this first method, we introduce two core concepts that we will continue to use in the following chapters: we describe the pixels by using multi-resolution features, and estimate the approximate region of the landmarks using a gaussian function. This chapter is divided into five sections. In Section 3.1 , we motivate the need for an automated landmark detection method in cephalometry. Then, in Section 3.2, we describe and analyze the image dataset. This analysis will be used in Section 3.3, where we describe our method and justify our design choices by using this analysis. In Section 3.4, we first analyze our cross-validation results. Then we introduce the other methods presented during the challenge, and compare our results with theirs. Finally, in Section 3.5, we draw the conclusions from the experiments that were made. Note that this method was the first step towards the development of a more generic approach for the detection of landmarks in the context of morphometric studies with different types of images. This extension will be presented in the next chapter.

This chapter is based on the work published in

> *R. Vandaele, R. Marée, S. Jodogne, and P. Geurts.* Automatic cephalometric x-ray landmark detection challenge 2014: A tree-based algorithm. *Proceedings of the ISBI International Symposium on Biomedical Imaging, Automatic Cephalometric X-Ray Landmark Detection Challenge, 2014.*
>
> *C.-W. Wang, C.-T. Huang, M.-C. Hsieh, C.-H. Li, S.-W. Chang, W.-C. Li, R. Vandaele, R. Maree, S. Jodogne, P. Geurts, C. Chen, G. Zheng, C.-W. Chu, H. Mirzaalian, G. Hamarneh, T. Vrtovec, and B. Ibragimov* Evaluation and comparison of anatomical landmark detection methods for cephalometric x-ray images: A grand challenge. *IEEE Transactions on Medical Imaging, 34(9):1890:1900, 2015.*

No.	Anatomical Landmarks
1	sella turcica
2	nasion
3	orbitale
4	porion
5	subspinale
6	supramentale
7	pogonion
8	menton
9	gnathion
10	gonion
11	lower incisal incision
12	upper incisal incision
13	upper lip
14	lower lip
15	subnasale
16	soft tissue pogonion
17	posterior nasal spine
18	anterior nasal spine
19	articulate

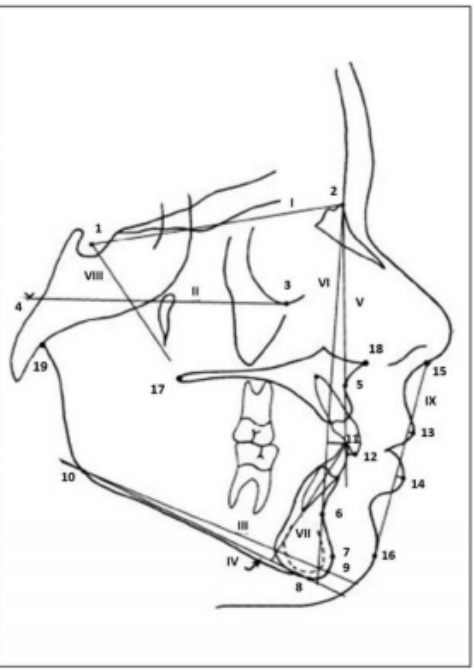


Figure 3.1: Representation of the 19 landmarks used in recent cephalometric studies.

3.1 Background

This chapter will focus on the development of a landmark detection method allowing to automatically detect landmark in the context of cephalometric studies. Cephalometric studies consist in the analysis of cephalograms, which are radiographs of the craniofacial area that can be acquired in 2D or 3D. During this analysis, the radiographs are used in order to compute measurements (distances and angles) between the different regions of the skull. Once these measurements are computed, they will allow orthodontists and stomatologists to diagnose and treat possible deformities. In another context, these measurements also allow to easily evaluate different parts of a patient's anatomy such as the shape and the size of the jaw, the mandible or even some internal soft tissues of the skull. This is why surgeons also use these studies to plan some of their surgeries such as the treatment of obstructive sleep apnea [91].

In the context of a cephalometric analysis, the measurements between the regions of the skull are computed by considering the distances and the angles between anatomical landmarks distributed on the whole area of the skull. In 1982, Rakosi [73] defined a set of 90 *standard* landmarks which have been used by orthodontists for clinical research to perform their cephalometric analysis. Among these, a smaller set of 19 landmarks was then commonly adopted in recent studies and for clinical practice [30, 95, 45, 48, 89]. The Figure 3.1 shows the repartition of these 19 landmarks on the skull.

These landmarks thus need to be annotated on each of the cephalograms that are studied before the measurements can be obtained. In practice, medical experts annotate these

images manually. This process is complicated: indeed, for some of the landmarks that are not easily recognizable, medical experts will need to extract their position by computing relative distances and angles from close shapes and/or structures. These structures and shapes will thus need to be manually traced before the location of the landmark can be annotated. In [30], it was evaluated that an expert took about 20 minutes in order to annotate a single cephalogram. Another problem with manual annotation comes from the intra and inter observer error [47]: expert bias and mistakes has become such a real problem that several clinical papers focused on the problem of the manual identification of these landmarks [26, 63, 19].

The automatic detection of these landmarks can be the solution to facilitate these issues. Some studies already focused on investigating methods for the automatic localization and identification of cephalometric landmarks. In 2006, Yue et al. [95] built a modified active shape model to detect 12 anatomical landmarks, achieving a 71% success rate of landmark detection within 2.0mm and 88% within 4.0mm. In 2009, Kafieh et al. [45] combined neural networks with modified active shape models and developed a technique with 93% landmark detection accuracy for 16 bony structures within 5.0mm.

However, it is considered that the automatic detection of anatomical landmarks in cephalometric images is difficult and poorly explored due to the complexity and the variability of these images. The problems arising with the anatomical landmark detection methods presented in recent studies are the following:

1. The type of the landmarks that are considered differ from one study to another. Some landmarks could be more easily identified than others, and some methods could be biased into accurately detecting one type of landmark while inaccurately detecting other types of landmarks.

2. The number of landmarks could have an influence on the performances of a given method: we can assume that the larger the number of landmarks, the more stable the models using the global landmark structure will be.

3. The number of training images will also have an influence on the performances: by using more training images, the models are likely to perform better because the algorithm will have more visual representations of the landmarks to learn from.

4. The variability in the dataset can also have a direct influence on the results: age and sex are for example factors that can have a direct influence on the patient's morphology. By only focusing on one subgroup, a method could thus be biased.

5. The way the landmarks were annotated can also have a direct influence on the results: it can be suspected that if the landmarks are inaccurately annotated, the landmark detection methods will have more difficulties to detect those points.

6. Given the study, we noticed that the performance metrics differ from one study to another: some consider the percentage of landmarks detected within a given radius while others only report the MRE. This complicates the comparison between methods, but we can also assume that these methods were thus optimized in order to best perform on their given criterion.

	Intra-Observer Variability		Inter-Observer Variability
	Expert 1	Expert 2	Expert 1 vs Expert 2
MRE (mm)	1.73 ± 1.35	0.90 ± 0.89	1.38 ± 1.55

Table 3.1: Averaged intra and inter observer annotation error on the landmarks of the cephalometric dataset.

Because of these problems, and the fact that it stays difficult to share sensitive patient related data, it is complicated to choose an algorithm given its stated performances in a paper: these could indeed be biased by the type of dataset that was used during the study.

The 2014 ISBI Cephalometric challenge was, to our knowledge, the first attempt to provide a common cephalometric dataset that researchers could use in order to evaluate their algorithm. In this chapter, we propose a novel landmark detection algorithm that was tested on this dataset, and provide valuable comparisons with state of the art methods presented during the challenge.

3.2 Dataset and performance criterion

The dataset was supplied by the National Taïwan University for the 2014 ISBI Cephalometric challenge [91]. Each image of the dataset is a two dimensional gray scale image of size 2400×1935 pixels, with 0.1mm^2 per pixel resolution. The cephalometric radiographs were collected from patients aged from 6 to 60 years old. Additional images were provided during two international challenges [91, 90]. The dataset was divided into three parts described below.

1. A training dataset consisting of 100 training images. Each image was supplied with the positions of the 19 landmarks presented in Figure 3.1 that were manually annotated.

2. A first test set *off-site* consisting of 100 images. For this dataset, only the images were supplied, the ground truth annotations were only known to the organizers in order to ensure the validity of the predictions.

3. A second *on-site* test set consisting of 100 images. For this dataset only the images were supplied to researchers. Ground truth annotations were also only known to the organizers.

In order to ensure the quality of the annotations, each landmark position was annotated by two medical experts. The final position of the landmark was chosen as the mean distance between the two annotations. The average distance between the corresponding landmarks annotated by the two experts is given in Table 3.1 as a comparison for the error of the landmark detection algorithms.

A sample image with the corresponding landmark is given in Figure 3.2.

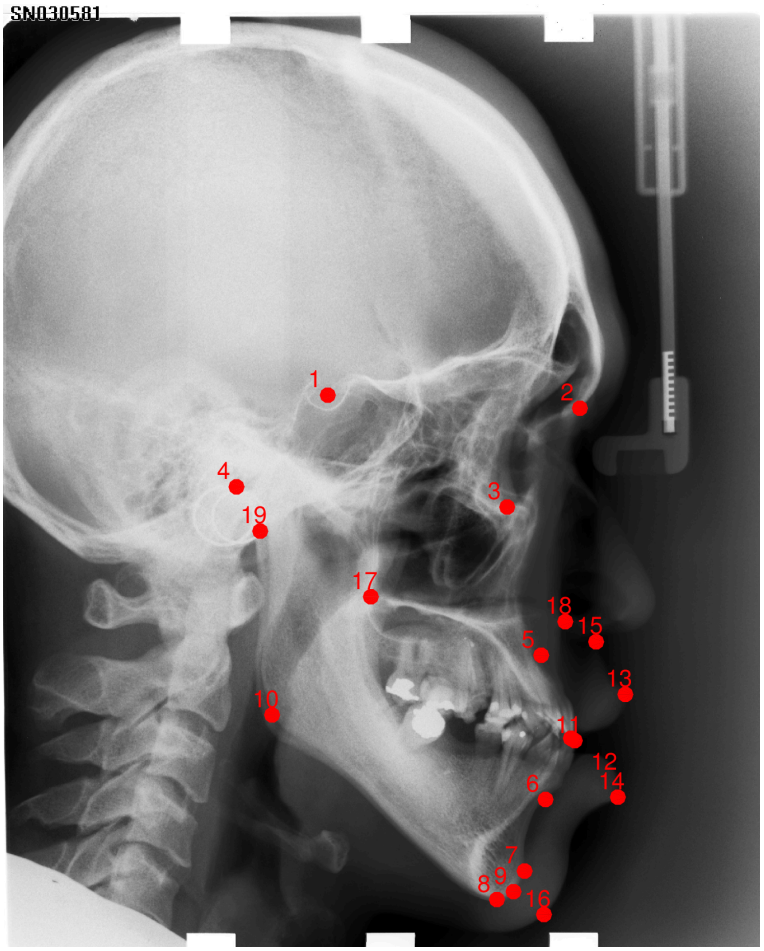


Figure 3.2: Sample image of the cephalometric dataset with the 19 corresponding landmarks.

During the challenge, 5 performance criteria were considered: 4 of them were the percentage of landmarks detected within a radius of $2, 2.5, 3$ and 4mm. The last one was the average euclidean distance between the predicted positions of the landmark, and their real positions (MRE). The idea behind considering these multiple criteria is that we have seen that landmark detection papers use different performance criteria, that could be related to specific area(s) of interest. By using these multiple criteria, we are thus offering an initial comparison with these papers. It can be noticed that the computed intra and inter observer variability presented in Table 3.1 is almost reaching the threshold given by the first and smallest error criterion (2mm).

LDM	$\sigma(X)$	$\sigma(Y)$	LDM	$\sigma(X)$	$\sigma(Y)$
1	45.12	44.2	**11**	76.16	76.47
2	63.66	65.68	**12**	74.25	74.92
3	49.85	58.28	**13**	66.79	79.63
4	38.44	38.3	**14**	81.31	82.32
5	68.25	67.4	**15**	62.87	77.04
6	85.64	77.99	**16**	98.84	92.92
7	96.51	89.05	**17**	47.45	49.15
8	96.22	91.56	**18**	60.72	74.34
9	97.01	90.94	**19**	38.49	43.93
10	61.04	62.63			

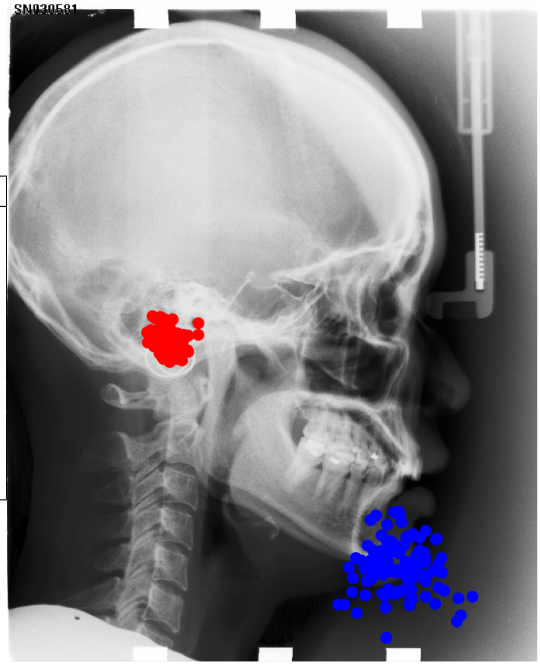


Figure 3.3: On the left, the table summarizes the standard deviation of each landmark, on both axes (in pixel units). On the right, an image on which the 100 positions of two landmarks in our training dataset have been superposed. In red, landmark 4, the landmark with the smallest deviation. In blue, landmark 16, the landmark with the highest deviation.

3.3 Methodology

3.3.1 Dataset analysis

Before detailing our method, we present the initial analysis that led us to the design choices made in our algorithm.

Landmark Positions. A first important observation we made on this dataset was the variance in the positions of the landmarks. The standard deviation of the landmark positions in the training dataset are given in Figure 3.3. When compared to the total size of the images, this table shows that the variance in position of the landmarks stays relatively small: the horizontal variance is about 2 to 5% of the total width of the image, and the same goes for the vertical variance, that is about 2 to 4% of the total height of the image.

This shows that the landmarks are located in very specific positions in the images. Depending on the model, this could suggest that full scans (with our without grid spacing) of images for finding a landmark, as it is for example suggested in [80] could thus spend considerable amounts of processing time extracting information where a given landmark has statistically no chance of being found.

Landmark Appearance. In the analysis of this dataset, it is also important to discuss on the visual appearance of the landmarks. Indeed, as in every landmark detection application, their visual appearance will play a major role in the choice of a suitable landmark detection method.

In Figure 3.4, we show the visual representations of the 19 landmarks at different different sizes of pixel windows.

In order to perform this analysis, we visualized the main differences between the landmark representations at different window sizes. For every landmark l and every window size w, we computed the corresponding mean image $\bar{I_{l,w}}$ in the dataset. If we consider that $(x_{i,l}, y_{i,l})$ represents the position of the landmark l in image I_i, then the mean image was computed as following:

$$\bar{I_{l,w}} = \frac{1}{N}\sum_{i=1}^{N} I_i(x_{i,l} - w : x_{i,l} + w, y_{i,l} - w : y_{i,l} + w), \tag{3.1}$$

where N represents the number of images in the dataset. For every landmark window l, w of every image i, we then computed its difference $\epsilon_{l,w}(i)$ to the corresponding mean image using RMS error:

$$\epsilon_{l,w}(i) = \sqrt{\sum_{x_w=-w}^{w}\sum_{y_w=-w}^{w}(I_i(x_{i,l} + x_w, y_{i,l} + y_w) - \bar{I_{l,w}}(x_w + w, y_w + w))^2} \tag{3.2}$$

Figure 3.4 shows some of the results we obtained. For each window size, we show the landmark window with the smallest (first row), the median (second row) and the highest (third row) difference for each landmark. From this figure, we can draw multiple conclusions:

- With the largest pixel windows (256 and 512), the landmarks appearances seems to slightly vary: even if the orientation of the images stays more or less identical, the morphology of a human being will vary. However, global rotations between the images and the global landmark structure stays small.
- With the smallest pixel windows, we can observe more stability in the landmark windows. However, it seems more likely that these pixel windows will be confused with other places in the image, even landmarks. For example, at the smaller resolutions, there is only few differences between landmark 1 and 17, or between landmarks 2, 14 and 16. And this without taking into account other places in the image that could lead to additional confusion.
- By looking at the smallest windows, we can also observe that some of the landmarks do not correspond to corners or intersections, but specific places in the images. Landmark 1 for example, seems to be defined at the center of an oblong shape.
- By using our mean squares criterion, we initially expected to observe high difference in luminosity, but looking at the windows, it clearly is not a problem in our dataset.

What we can learn from these observations is that specific window sizes will provide different kinds of visual informations about the window appearance: while the smaller windows will give an accurate description of the surroundings of the landmark, they be could easily confused with other structures, landmarks or other structures in the image. On the other hand, larger pixel windows have a more varying content and are thus more difficult to recognize accurately, but because these windows contain a larger amount of informations, they will also be harder to confuse with other structures in the image. This is why, depending on the landmark and the error criterion that will be considered, we suspect that multiple resolutions will be necessary in order to correctly detect the landmarks. Additionally, we suspect that because some landmarks do not correspond to borders or intersections, classic pixel descriptors such as SURF, SIFT, LBP, or even gradient information will be less useful than usual. This phenomenon could probably be even more increased due to the fact that problems of rotations and luminosity stays relatively small in our dataset.

Conclusion. Given current landmark detection applications that focus on face recognition, this landmark detection problem for cephalometric images seems to be quite different: we have bigger images, with a smaller number of landmarks a smaller dataset. Because the landmarks are not always located at corners or intersections, landmark detection techniques based on edges and corners detection will likely be less efficient than usual.

3.3.2 Presentation of the method

Given the analysis, we chose to improve a supervised learning approach based on the work of Stern et al. [80]. Stern obtained promising results for detecting landmarks on several datasets of zebrafish images, sharing the same traits than our cephalometric dataset.

This method exploits the manually annotated images to train models able to predict landmark positions in new, unseen images.

In particular, a separate classification model is trained for each landmark to predict whether a given image pixel corresponds to the position of the landmark or not. This model is trained from a learning sample composed of pixel descriptors extracted either in the close neighborhood of the landmark or at other randomly chosen positions within the training images. Each pixel in the training sample is described by a vector of visual features extracted from a window centered at its position.

In order to find the landmark position on a new image, all of the image pixels are classified. The final landmark position corresponds to the median position of the pixels classified as having the highest probability to be the landmark.

Our algorithm, described in the following sections extends this work in order to speed-up the classification procedure, and allow the parametrization of some design choices.

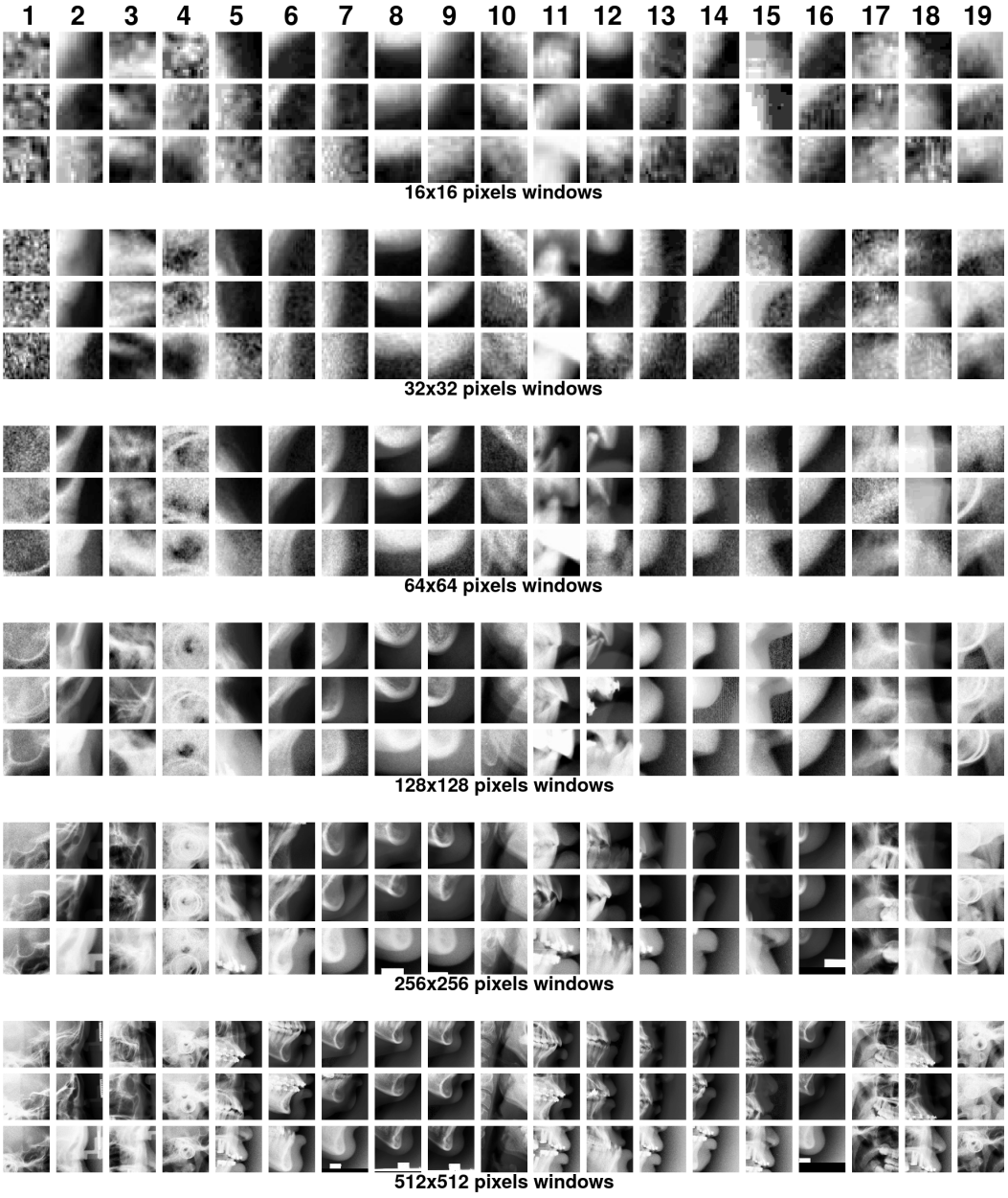


Figure 3.4: Appearance of the landmarks using pixel windows of different sizes. The landmarks are located at the center of the windows.

Extraction and description of pixels

Each observation in the training sample corresponds to a pixel at position (x, y) in one of the training images and is labeled into one class among $0, 1$ and described by several input features. We describe below successively the output class associated to each pixel, the input features used to describe them, and the pixel sampling mechanism.

Output classes. In principle, only one position in each image corresponds to the landmark, which means that if N training images are available, only N positive examples would be available to train our pixel classification model. To extend the set of positive examples, we consider as positive examples all pixels that are at a distance at most R from the landmark, where R is a method parameter. More precisely, if the landmark is at position (x_l, y_l) in an image, then the output class of a pixel at position (x, y) in the same image will be 1 if $(x - xl)^2 + (y - yl)^2 \leq R^2$, 0 otherwise.

Multi-resolution input features. In the previous section, we showed that the landmark's visual appearance could drastically change from one window size to another, and that different window sizes could help the algorithm to correctly detect the landmark's positions. This is why, prior to extracting the pixel descriptors, the image is resized to six different resolutions given in Equation 3.3.

$$I_r = \frac{2400}{2^r} \times \frac{1935}{2^r} \forall r \in \{0..5\} \tag{3.3}$$

A pixel located at position (x, y) will then be described by one window F_r of size $(2W+1)^2$ for each resolution r, described in Equation 3.4. These descriptors will then be concatenated into a single descriptor. In order to describe these multi-resolution windows, we will use its raw pixel values. As we have seen in Chapter 2, it was indeed shown in [64] that raw pixel values were able to perform correctly for most classification tasks when used in combination with Random Forests. This choice differs from Stern [80], who concatenated several single-resolution pixel descriptors (LBP, Sobel Gradient, RGB and HSV values). We made this choice in order to speed-up the extraction procedure at both training and prediction. The intuition being that, given our dataset, it was more important to provide informations about the landmark pixel location than enforce strong robustness against deformations.

$$F_r = I_r(\lfloor \frac{y + t_y}{2^r} \rfloor - W : \lfloor \frac{y + t_y}{2^r} \rfloor + W + 1, \lfloor \frac{x + t_x}{2^r} \rfloor - W : \lfloor \frac{x + t_x}{2^r} \rfloor + W + 1) \forall r \in \{0..5\} \tag{3.4}$$

Pixels of a subwindow extending beyond the image limit will be set to zero. In total, each pixel will thus be described by an input feature vector of size $6 \times (2W + 1)^2$.

Landmark	Avg. distance (mm)	STD	Landmark	Avg. distance (mm)	STD
(1)	5.47	3.16	(11)	9.73	4.69
(2)	8.00	4.46	(12)	9.50	4.61
(3)	6.79	3.58	(13)	9.29	4.69
(4)	4.91	2.33	(14)	10.29	5.32
(5)	8.66	4.15	(15)	8.89	4.48
(6)	10.37	5.18	(16)	12.10	6.17
(7)	11.78	5.84	(17)	6.06	3.17
(8)	11.86	6.01	(18)	8.55	4.39
(9)	11.90	5.97	(19)	5.32	2.43
(10)	7.73	4.12			

Table 3.2: Average distance of the landmarks to the average of their positions on the training set.

Pixel sampling scheme. Naively sampling pixels uniformly from the training images will give a very unbalanced classification problem. Indeed, each image contains 2400×1935 pixels among which only a small number belongs to the positive class. For example, for a radius $R = 2$mm, only 0.027% of the pixels (i.e., 1256) correspond to positive examples.

In opposition to Stern's method where all the positive pixels are extracted with two times the number of randomly sampled negative pixels, we wanted to have more control over the composition of our training dataset. With our method, we randomly select N pixels in each training image, where $P\%$ of these N pixels are selected among positive pixels and $100 - P\%$ are selected among negative pixels.

In addition, we constrained the image area in which the negative pixels are selected by taking into account the fact that a landmark is located in very close positions from one image to another. To confirm that, Table 3.2 reports the average distance of each landmark to its average position over all training images. These numbers show that each landmark is located in a very specific region of the image of radius of size between $5 - 15$mm. At the prediction stage (see below), we will use this information to constrain the search for a landmark to a given subregion of the image around the average landmark position in the training images. Therefore, it is enough to put in the training sample only pixels that belongs to this region. Negative examples in each image will be selected uniformly at random at a distance of at most 40mm around the landmark.

Classification model training

To train the pixel classifier, we will use the Extremely randomized tree algorithm [38]. As we explained in chapter 2, this method builds an ensemble of T fully developed decision trees grown each from the original training sample (i.e., without bootstrapping). At each node, the best split is selected among k features chosen at random, where k can take its value between 1 and m, with m the total number of features. For each of the k (continuous)

features, a discretization threshold is selected at random within the range of variation of that feature in the subset of observations in the current tree node. The score of each pair of feature and threshold is computed and the best pair among the k is chosen to split the node. As a score measure, we use the Gini index reduction.

Landmark prediction

Let us denote by $\mu_l \in \mathbb{R}^2$ and $\Sigma_l \in \mathbb{R}^{2\times 2}$ respectively the average and the covariance matrix of the landmark positions across the training images and let us denote by σ_{x_l} and σ_{y_l} the standard deviation of its x and y positions respectively (i.e., the diagonal elements of Σ_l), also estimated from the training data. To make prediction of the the landmark position with our tree-based pixel classifier, we proceed as follows:

- We randomly draw $16\sigma_{x_l}\sigma y_l$ pixel positions from the multivariate normal distribution:

$$\mathbb{N}(\mu_l, \Sigma_l) \tag{3.5}$$

- Each of the resulting pixels is classified by the tree ensemble and the final predicted landmark position is taken as the median position among the pixels that are predicted as being the landmark with the highest confidence by the tree-based model (i.e, which receives the highest number of votes for the positive class from the T trees in the ensemble).

This subsampling scheme also differs from Stern's work. It allows us to improve predictive performance by reducing the probability of generating spurious landmark predictions at irrelevant positions in the images. It also considerably speedups the algorithm with respect to a full scan of all image pixels.

3.4 Results Analysis

3.4.1 Cross-Validation Results

Protocol

In this section, we describe the protocol we adopted to generate all 19 landmark detection models.

Parameter tuning. First, parameters were set to some default value or optimized using ten-fold cross-validation and then a model was retrained, for each landmark and error criterion, using the optimal set of parameters. The main method parameters and their values that were tested are presented in Table 3.3.

Name	Description	Values tested
W	The size of the windows	8
R	The distance to the interest point to decide on the training pixel output class (mm)	$\{0.2, 0.5, 0.7, 1, 1.2, 1.5, 1.7, 2, 2.5, 3\}$
P	The proportion of non-landmark pixels in the dataset (%)	$\{10, 20, 30, 33.33, 40, 50, 60, 70, 80, 90\}$
t_x	The translation of the subwindow (horizontal)	$\pm\{0.8, 1.6, 3.2, 6.4, 12.8$ $25.6\}$
t_y	The translation of the subwindow (vertical)	$\pm\{0.8, 1.6, 3.2, 6.4, 12.8$ $25.6\}$
N	The number of pixels per image randomly sampled to train each classification model	500
k	The number of features selected at each node in the Extremely Randomized Trees algorithm	$\sqrt{6 \times (2W+1)^2}$
T	The number of trees	500

Table 3.3: Description of the method's parameters and values tested at cross-validation.

During parameter tuning, T was fixed to a default value of 500 and we used the suggested default value of k, which is the square root of the number of input features [38]. N was fixed to 500 and W to 8 in all our experiments. All other parameters were tuned by 10-fold cross-validation independently for each landmark and each error criterion relevant for the challenge.

The parameter tuning was done in several steps:

Step 1) The optimal values of t_x and t_y were jointly tested for translations using $R = 1$mm and $P = 33\%$.

Step 2) R was then optimized using $P = 33\%$ and the optimal values of t_x and t_y determined at the previous stage.

Step 3) Finally, P was optimized with the other parameters set at their optimal values.

In total, this represents about 2000 cross-validation jobs for each criterion.

Final model training. Separate models were then retrained using all 100 training images for each landmark and error criterion using the optimal values of the parameters determined during the cross-validation. All non-optimized parameters were set similarly as during the cross-validation except the number of trees T, which was raised to 5000. Landmark predictions were then generated on the test image using the approach described in 2.3 (for each landmark and error criterion).

Software. We use the implementation of the Extremely Randomized Trees in scikit-learn [70] and our own python code for pixel and feature computation. Visual interpretation of

Landmark	$\leq$ 2mm	$\leq$ 2.5mm	$\leq$ 3mm	$\leq$ 4mm	Eucl. Dist.
sella (1)	87	90	93	96	1.4 ± 1.2
nasion (2)	80	86	86	91	1.8 ± 2.0
orbitale (3)	61	72	81	87	2.1 ± 1.7
porion (4)	76	86	92	96	1.6 ± 2.1
subspinale (5)	45	57	72	83	2.9 ± 2.5
supramentale (6)	68	80	86	95	1.9 ± 1.6
pogonion (7)	90	95	95	97	1.2 ± 1.4
menton (8)	95	97	98	99	0.9 ± 0.8
gnathion (9)	95	97	99	99	$1. \pm 1.2$
gonion (10)	36	46	55	69	3.8 ± 3.1
lower incisal incision (11)	83	87	93	95	1.4 ± 2.3
upper incisal incision (12)	87	89	92	94	1.6 ± 4.7
upper lip (13)	84	88	91	95	1.8 ± 2.8
lower lip (14)	84	90	94	96	2.4 ± 5.1
point pm or mn (15)	88	94	94	98	1.2 ± 1.2
soft tissue pogonion (16)	64	74	81	88	1.9 ± 1.8
posterior nasal spine (17)	84	89	94	98	$1.5 \pm 1.$
anterior nasal spine (18)	63	72	78	88	2.1 ± 1.9
articulate (19)	62	69	74	81	2.2 ± 2.3
Mean	75.37	82	86.74	91.84	1.83 ± 1.81

Table 3.4: Results on all landmarks **without** translation, in terms of detection rates at various ranges of accuracy and mean euclidean distance to the landmark.

the results was done using Cytomine [64], a generic web platform for the visualization and annotation of large-scale bioimages.

Cross-Validation Analysis

Tables 3.4 and 3.5 report the best cross-validation performances after optimization for each criterion, respectively without and with translation. In this latter case, we also report in the table the values of t_x and t_y that give optimal performance for the 2.5mm detection rate. Note that values in these tables are optimal values over different parameter settings. They are therefore most probably optimistically biased and only provided here for information purpose. A more realistic assessment of our method performances will be done on the challenge test data. There is a clear improvement for some landmarks by using translations. The sella point for example, is more correctly detected. We notice however that two particular points are not correctly detected, even at higher acceptance criterion: the supramentale and the gonion. Given the good results obtained on other landmarks and other inconclusive tests we have made on these two points, our conclusion is that either the dataset is not able to capture the high variability of the surrounding of these landmarks or there were some errors during the manual annotation process.

Figure 3.5 shows the position of the gonion on different images. It seems that the local

Landmark	$\leq$ 2mm	$\leq$ 2.5mm	$\leq$ 3mm	$\leq$ 4mm	**Eucl. Dist.**	t_x	t_y
sella (1)	95	96	96	97	1.21 ± 1.92	3.2	1.6
nasion (2)	78	83	86	90	1.86 ± 2.06	0	0
orbitale (3)	63	75	83	92	2.06 ± 1.50	0.8	−6.4
porion (4)	77	86	92	97	1.53 ± 1.22	0	0
subspinale (5)	54	63	71	83	2.78 ± 2.20	0	1.6
supramentale (6)	71	78	86	95	1.84 ± 1.56	−1.6	−0.8
pogonion (7)	89	94	97	99	1.21 ± 1.30	0	0
menton (8)	94	97	98	100	0.94 ± 0.80	0.8	−0.8
gnathion (9)	97	99	99	100	0.91 ± 0.69	−3.2	−0.8
gonion (10)	38	48	56	66	3.76 ± 2.85	1.6	1.6
lower incisal incision (11)	89	92	95	97	1.44 ± 2.35	−1.6	1.6
upper incisal incision (12)	88	92	95	97	1.29 ± 3.27	3.2	−6.4
upper lip (13)	84	89	93	95	1.56 ± 2.08	0	−0.8
lower lip (14)	87	93	96	99	1.45 ± 2.36	−0.8	−0.8
point pm or mn (15)	88	92	95	98	1.19 ± 1.07	1.6	1.6
soft tissue pogonion (16)	67	75	83	91	1.94 ± 1.80	−0.8	−1.6
posterior nasal spine (17)	83	90	95	98	1.38 ± 1.06	0.8	0.8
anterior nasal spine (18)	67	78	84	91	2.01 ± 1.56	−3.2	0
articulate (19)	65	74	79	86	2.28 ± 2.06	1.6	3.2
Mean	77.58	83.89	88.37	93.21	1.72 ± 1.77		

Table 3.5: Results on all landmarks **with** translation, in terms of detection rates at various ranges of accuracy and mean euclidean distance to the landmark.

position of the landmark does not fit the same structure on each of the images.

3.4.2 Challenge results and comparisons

As we explained in Section 3.2, the test set was divided into two different parts: an offline dataset (Test1) made of 100 images, and an online dataset (Test2) made of 100 other images. For both of these datasets, the landmark annotations were not available. A summary of the results we obtained during the challenge is presented in Table 3.6. A landmark per landmark MRE comparison is given in Figure 3.6.

Comparison with the cross-validation results

We can notice that the accuracy of our method has greatly decreased between the results obtained during the cross validation and the results obtained on the test set. We can also notice a slight difference in accuracy between the two test sets, which seems to show that the second test set is more difficult to handle for our method than the first one. By analyzing the difference in detection error, we can notice that the biggest differences in terms of detection error happen on the error rates with the smallest radius while the biggest radius

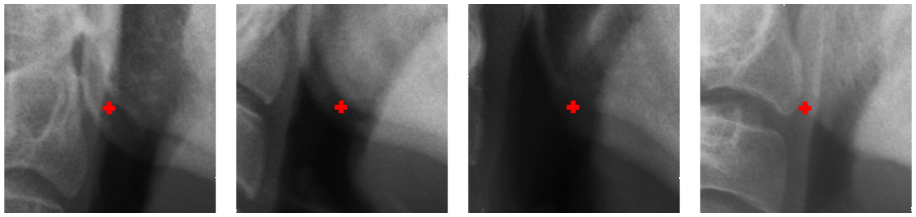


Figure 3.5: Gonion surroundings for training set images 10,20,30 and 40. In red, the position of the gonion landmark.

Method	2mm	2.5mm	3mm	4mm	MRE
Cross-validation results					
Vandaele et al.	77.58	83.89	88.37	93.21	1.72
Test 1 dataset					
Ibragimov et al. [43]	72.74	78.79	82.68	87.68	1.819
Chu et al. [20]	39.74	51.79	62.11	77.79	2.921
Chen and Zheng [18]	43.89	54.58	64.21	78.42	2.805
Mirzaalian and Hamarneh [67]	58.21	67.32	72.84	80.68	2.605
Vandaele et al. [87]	70.26	77.95	83.47	88.53	1.951
Test 2 dataset					
Ibragimov et al. [43]	70.21	76.37	81.58	88.16	1.919
Chu et al. [20]	44.11	57	68.05	83.84	2.679
Chen and Zheng [18]	42.89	53.89	65.32	78.53	2.847
Mirzaalian and Hamarneh [67]	62.32	70.42	75.68	84.05	2.353
Vandaele et al. [87]	66.74	74.32	80.26	87.84	2.198

Table 3.6: Results of the challenge on the first and second test dataset.[91]

seems less impacted by this difference. It thus seems that the landmarks are predicted in the correct region of the image, but our method still encounters some difficulties to accurately locate the position of the landmark inside this region. On the positive side, this suggests that our method estimating the region in which the landmark with a gaussian function works, and that the largest pixel windows allow us to locate this region. On the negative side, by looking at the results, we can notice that our method has clearly more difficulties to accurately locate the landmark and its close surroundings. Given the significant difference between the cross-validation and the test sets on these criterions, this could mean that we overfitted some of the method's parameters which mostly influenced the accurate localization of the landmark during the cross validation. These could be the translation parameters t_x and t_y: as we consider only *small* translations from 0 to 32 pixels, we can notice that the content of the small pixel windows will always change with a higher percentage than the biggest pixel windows.

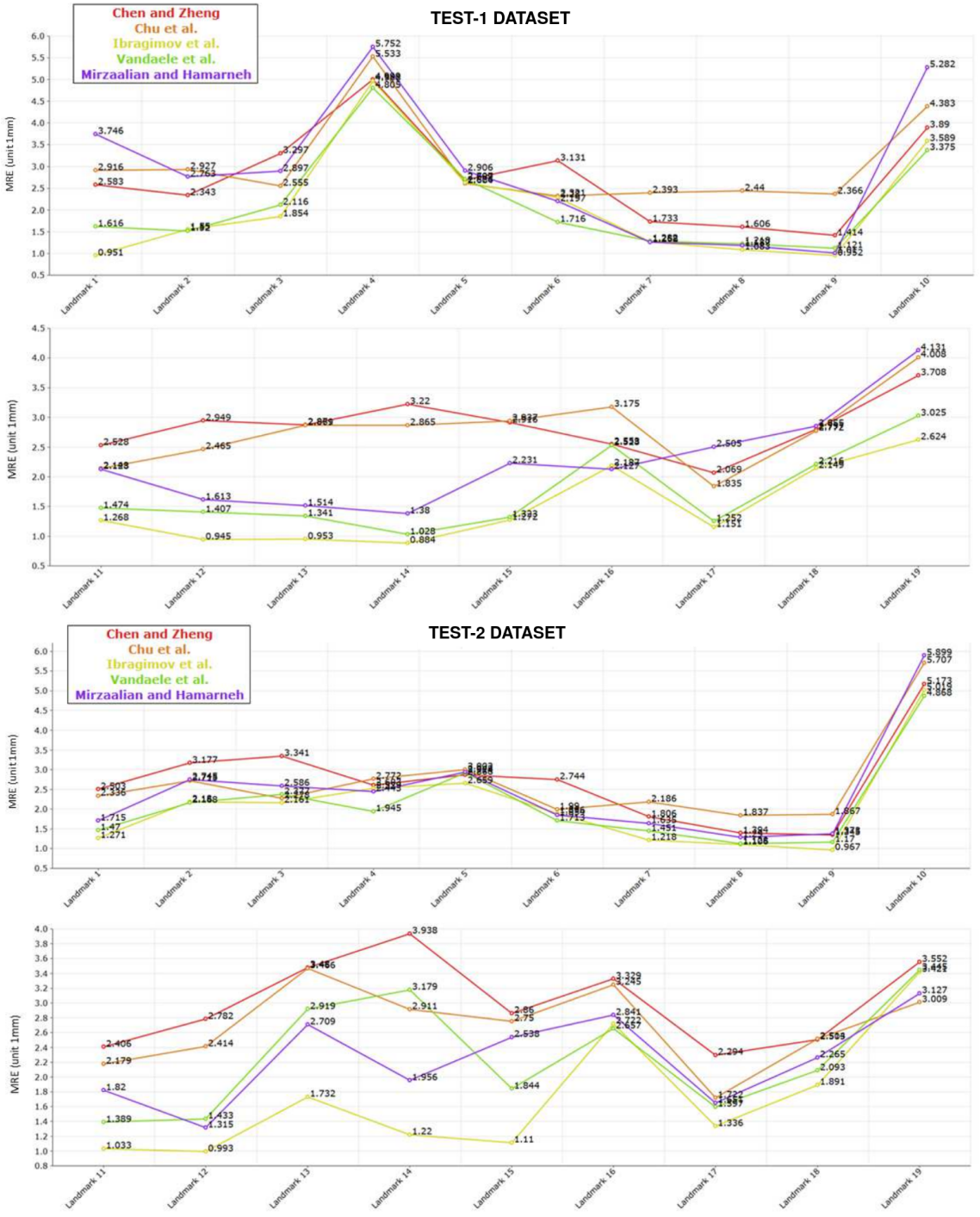


Figure 3.6: Reported MRE prediction error of the 19 landmarks for every algorithm. This figure was presented in [91].

Comparison with other methods and observations

During the challenge, we were ranked second. The results, presented in [91], are shown in Table 3.6. 12 methods were initially presented, but only the 5 best methods remained for being tested on the second test dataset.

The first method was proposed in [43]. In this method, the authors propose a two step approach: in the first step, they use one random forest pixel classifier per landmark to evaluate the likelihood of the pixel to correspond to the landmark. The pixels will be described by using Haar-Like features coming from their close neighborhood. These models will allow them to create *appearance likelihood maps* for each of the landmarks in a new image. In a second step, they model the relations between each pair of landmarks by two gaussians: one based on the average distance between the landmarks, and another based on the average angle between these landmarks. By considering only the likeliest landmark positions according to their appearance likelihood maps, they are thus able to score each landmark position combination by considering both their appearance and their relationships in distances and angles. By considering the landmarks as players, the landmark positions as player strategies, and the score as a game output, they model this problem as a game theory problem, and are thus able to find the optimal solution of their optimization problem using game-theory based algorithms. In the context of this challenge, 40 additional landmarks were manually added to the dataset in order to ensure the stability of the global landmark structure.

The third method (our method was randed second) was proposed in [67]. In this method, the landmark detection problem is formulated as an optimization problem taking into account both the landmark visual appearance and the relationships between pairs of landmarks. In order to learn the visual appearance of a landmark, they use one random forest pixel classifier per landmark. They describe their pixels by several descriptors: local binary patterns on RGB and HSV modalities [69], x,y coordinates and Frangi et al. descriptors [36]. The spatial relationships between the pairs of landmarks is expressed as the manahalobis distance between the average vector separating the pair of landmarks in the training dataset and this vector computed at prediction. The computation of the landmarks optimal position is done through the pictorial structure algorithm [34] that will link this problem to a minimum spanning tree optimization problem.

The fourth method was proposed in [20]. In this method, one random forest pixel regressor per landmark is trained. These regressor will learn the offset between the position of a pixel and the position of its corresponding landmark. Each pixel will be described by a Histogram of oriented Gradient (HoG) [25]. At prediction, pixels will be extracted from the image, and each tree of each forest will vote for the position of its corresponding landmark through a small gaussian distribution which should generate accurate *vote maps* for each landmark. These vote maps will then be used to generate a vector y_0 representing an initial guess of the landmarks positions in the image. This vector y_0 will then be updated

through an iterative process that will take into account the global landmark structure: at each step, they try to find the optimal representation of the landmarks position y by trying to explain y through a sparse linear combination of shapes of the training dataset, where the sparsity is controlled with a parameter λ.

The fifth method was proposed in [18]. In this, method, the authors also propose to divide the detection in a two step approach: in the first step, they propose to produce probability vote maps for each landmark. Their idea is to train random forest regressors that will predict the offset between square patches in the image and the landmark position. The correction of their landmark position vector will then be similar to the previous algorithm.

Comparison with the algorithms. We can notice a significant gap between the three best methods, and the two last, that are barely able to reach a 50% detection rate with the 2.5mm error detection rate. Notice that the two last accepted methods were very similar. On the other hand, except on two performance criterions of the first dataset, we are always slightly outperformed by Ibragimov et al. except on two criteria for the first dataset, and we always outperform the third method (Mirzaalian and Hamarneh). By looking at the MRE, it is interesting to notice that all the methods follow approximately the same trend: if one landmark is more difficult to detect for a given method, it is also more difficult to detect for others. Another interesting point is that we obtain the best performances on the landmark 10 on both the Test-1 and the Test-2 datasets while this landmark was described as a landmark for which location was only defined by the locations of the other landmarks.

From this perspective, we can conclude several things. Although we were the only team not using a method correcting the landmarks positions after training, we still obtained the second best results during the challenge, and the only algorithm that outperformed us had to use manually added landmarks to the dataset in order to ensure the consistency of the dataset. This method still obtains results worse than ours for landmark 10, a landmark which location can only be accurately retrieved using the location of the other landmarks. As we will try to show in the following chapters, we suggest it is due to the particular nature of the dataset where we have a limited number of images. The images being of considerable size, the small number of landmarks does not allow to completely refine the landmark positions by using the relationships between these landmarks, except by adding a significant number of landmarks in the dataset. In a second time, those results also seem to confirm our hypothesis that multi-resolution features are useful during the detection of the landmarks: our simple raw pixel value multi-resolution feature descriptor obtains results similar to the other algorithms without any type of post-processing correction. It is however complicated to certify its usefulness from these results, because the other pixel descriptors that were tested during this challenge were tested in different contexts: the post-processing. Finally, we could observe that every other method during this method was performing full image scans in order to find the landmarks. Using our approach to approximate the region of each landmark could thus speed-up the building of their vote

	Landmark 4		Landmark 10		Landmark 19	
	Expert 1	Expert 2	Expert 1	Expert 2	Expert 1	Expert 2
Test 1	4.5 ± 2.4	1.8 ± 2	2.3 ± 1.8	0.9 ± 0.8	3 ± 3	0.8 ± 1.1
Test 2	2.5 ± 1.4	1.7 ± 1.8	2.6 ± 1.6	0.1 ± 0.9	2.2 ± 2.4	0.8 ± 1.6

Table 3.7: Intra observer variability of manual marking of landmark 4, 10 and 19. This Table was first presented in [91].

maps without affecting their results.

In terms of detection speed, although we did not implemented the other algorithms that were proposed and no such study was performed during the challenge, it seems that our algorithm is the fastest among the ones proposed: we do not have any post-processing step requiring a possibly heavy optimization procedure, and our algorithm simply uses raw pixel values as pixel descriptors. Moreover, instead of building a complete vote map for each of the landmarks, we limit the search of the landmark inside a region of the image.

Comparison with intra and inter observer variability. Intra and inter observer variability in the annotations could have an important impact on the results, not just at training, but also at prediction: if the landmarks were poorly annotated, then we can expect to obtain poor-quality results. Averaged intra and inter observer variability is reported in Table 3.1, and intra observer variability at prediction was reported for 3 landmarks, shown in Table 3.7. We can notice that the increased intra-observer variability of landmark 4 in Test 1 corresponds to a peak in detection error of every method in Figure 3.6. We can also notice that the detection error of landmark 19 is higher with Test 2, which corresponds to the increase in annotation variability of the first expert. Landmark 10 being a specific landmark (its position depends on the location of the other landmarks), the difference in annotation error is less significant. From these three landmarks, it is difficult to give a final conclusion but it seems that intra observer variability has had an influence on two of the landmarks where we could observe significant differences.

When compared to the intra and inter observer variability in the training set, we can conclude that there is still place for improvement for our landmark detection method, that do not reach the intra nor inter observer variability that was observed. This is however also the case for the first method of the challenge.

3.5 Conclusion

We showed that it was possible to accurately detect some of the landmarks using a combination of Extremely Randomized Trees and simple multi-resolution features. We think that given the small size of the dataset and the variance of the landmarks between the images, these results are promising in comparison to existing algorithms. However, for some landmarks, our results are still significantly worse than human annotations, which

means that our algorithm can only be considered in the context of manual assistance. Still, this algorithm is competitive with state-of-the-art methods.

The main advantage of our approach with respect to existing works is its simplicity and efficiency: it considerably reduces the number of pixels to extract at prediction while using simple features that can be easily extracted. High level features such as Zernike moments or Haar-Like can more accurately describe an image or a window, but they are slower to compute, and this could be detrimental in some applications.

The results of the algorithm presented above were thus considered satisfying: by making a detailed analysis about the dataset, we were able to build an efficient algorithm regarding state of the art methods:

- Our multi-resolution approach allows us to consider several resolutions at the same time and still use simple pixel descriptors. This approach seemed to have brought advantage when compared to other method using more sophisticated pixel descriptors.
- We are able to accurately detect the landmark by only extracting pixel descriptors for a small amount of locations. This allows us to significantly speed up the method when compared to classical full-scans of the images.
- We showed the advantage of using post-processing landmark position refinement was not always improving the results, especially during this challenge.

However, it seems that our method can still be improved in several ways:

- By comparing cross-validation and test results, it seems clear to us that some of our parameters were overfitted during cross-validation. An improvement to our method would be to prevent this overfitting.
- Some choices were made without real validation: the number of pixel positions extracted at prediction and the choice of a suitable pixel descriptor for example.
- Another way to improve our method could be to make our method more robust to deformations: this could be done through the use of a more robust pixel descriptor, but also by modifying the way pixels are extracted at training from the images.

Chapter 4

Landmark detection methods for 2D morphometrics studies

In this chapter, we extend and thoroughly analyse the anatomical landmark detection method that we have presented in the previous chapter. In an attempt to make this method more generic, we study the impact of the complete and extended set of its parameters using a cross-validation approach on three different datasets used in morphometric studies: the cephalometric dataset that was presented in the previous chapter along with a dataset of zebrafish larvae images and a dataset of drosophilia wings images. Given the diversity provided by these datasets, we are also able to extract generic conclusions on how to set the parameters of the method according to the types of landmarks and images of the dataset. This chapter is divided into five parts: in Section 4.1, we motivate the need for a generic landmark detection method. Then, we present our three datasets in Section 4.2. We then present our method and its differences with the one presented in the previous chapters in Section 4.3. In Section 4.4, we show our cross-validation results and compare our method with two state-of-the-art anatomical landmark detection methods that were reimplemented. Finally, we extract conclusions from our comparisons in Section 4.5. The algorithms implemented in this chapter were integrated to the Cytomine open-source software, and the datasets made available to foster further research.

This chapter is based on the work published in

> *R. Vandaele, J. Aceto, M. Muller, F. Péronnet, V. Debat, C.-W. Wang, C.-T. Huang, S. Jodogne, P. Martinive, P. Geurts, and R. Marée.* Landmark detection in 2d bioimages for geometric morphometrics: a multi-resolution tree-based approach. *Scientific Reports , 8(1):538, 1 2018.*

4.1 Background

We have seen in the previous chapter that cephalometric studies consisted in the study of skull radiographs images where distances, shapes and angles were measured in order

to help clinician and other experts to draw conclusions about a patient's anatomy and plan their treatment or their surgery in consequence. Cephalometry is one of the specific application of geometric morphometric studies, that is a generic term for the analysis of distances, angles and shapes for all kinds of objects and images. Especially, it has become the dominant set of methods used to quantify sizes and shapes of biological objects [51]. It involves the analysis of configurations of landmarks (ie. discrete anatomical loci) among individuals and has been applied to a huge diversity of models and research questions. For example, it has already been used to study the morphology of neanderthal's fossils humeri [77], extract digging rules from the shapes of dinosaur's skeletons [32], study the ancestry of butterflies through the measurements made on their wings [17] or even study the variations of flower shapes [83]. In this chapter, we will focus on three application of morphometric studies: cephalometry, and two others: the first one is the study of zebrafish larvae images. The zebrafish larvae are often used as models in pharmacology to study the impact of drugs on their development (growing of their cartilage, bones, outgrowths), and in order to measure these impacts, morphometric measurements must be taken [1]. The second is the study of drosophilia wings. Drosophilia wings are also a model used in the context of developmental biological studies. By studying geometric morphometric measurements, experts are able to study the process of evolution (gene transmission,...).

Typically, the detection process is identical as in cephalometry: landmark positioning is first performed manually in individual two-dimensional images. Then, landmarks configurations are compared using, e.g., Procrustes superimposition [8] and various multivariate statistics can be applied to characterize landmark configuration variations - and thus shape changes - in large populations. As such studies could typically involve hundreds or even thousands of individuals and tens of landmarks, the need to manually position the landmarks prior to such analysis is a very limiting factor. There is therefore a strong need for (semi-)automated landmark detection methods in biology.

In computer vision, we saw that the problem of landmark localization has been extensively studied in faces [15, 93]. Methods for face analysis can however not be easily transposed to biological images, because of their very different and variable nature. The small size of ground-truth datasets typically available in biology also requires to design more data-efficient approaches. In biology, the landmark structure is also very different than in face images. Indeed, the number of landmarks of interest is typically small (~ 20 landmarks) and the images typically large ($\sim 1500 \times 1500$), which makes the landmarks more spaced apart than in face images. In the biomedical field, the problem of automatic landmark positioning has been mainly addressed in cephalometry. Several successful landmark detection algorithms such as ours have been proposed in this domain that are based on pixel classification or regression using machine learning techniques possibly followed by global landmark structure refinement [43, 28, 59]. Because these approaches have been proposed in the literature to tackle this specific cephalometric application none of them was systematically evaluated on a broader range of biomedical applications.

In this chapter, we study variants of the cephalometric landmark detection method that was proposed in the previous chapter. One of the goal of this chapter is to improve our algorithm with the observations that we made in the previous chapter, while the other will be to create a more generic landmark detection method that makes no specific

assumption about the types of images to analyze and landmarks to detect. This method is thus still based on the extraction of multi-resolution features and the use of generic tree-based ensemble machine learning methods, namely (Extremely) Randomized Forests (see [12],[38]).

The contributions made with this chapter are the following:

- We propose a novel generic learning-based approach for landmark detection.
- We thoroughly study the effect of its parameters on three diverse bioimage datasets (for human cephalometric radiographs, zebrafish skeletogenesis, and Drosophila wing developmental studies). From this analysis, we derive guidelines for choosing these parameters on new problems.
- We compare our method with several landmark detection algorithms [28, 59, 43] from the literature, both on the same three datasets and on two cephalometric challenges. These comparisons show that our approach yields competitive results in terms of accuracy, with lighter models and lower prediction times.
- We provide an open-source implementation of these algorithms through the Cytomine platform [66] that further implements proofreading tools to combine automatic detection and manual refinements.
- As an important side contribution, we provide an easy access to the datasets used in this study with the hope that the landmark detection problem will gain more interest in bioimage informatics and machine learning research.

4.2 Materials

We tested our method on the three datasets summarized below. An illustration of the landmarks is given in Figure 4.1 with one image per dataset and their corresponding landmarks.

The standard deviation of the landmarks is given in Table 4.1. We can observe that the landmarks of all the datasets follow the same trend that was observed in Chapter 3: the position of the landmarks vary only in small areas inside the images.

- **CEPHA**, a dataset of 100 lateral human cephalometric radiographs. This dataset has been previously described in [91, 90] and in the previous chapter. Some of the landmarks corresponds to visual edges (7, 8, 13) while some others corresponds to morphological locations with less local visual information (1, 4, 19). In this chapter, given that we only know the landmark positions for the 100 first images, only these images will be used.
- **DROSO**, a dataset of 138 colored images of Drosophila wings. Image resolution is 1440 by 900 pixels. Fifteen morphometric landmarks were manually acquired on 138 images as described in [27]. Note that on this dataset most landmarks are located at highly informative locations such as edges and intersections.

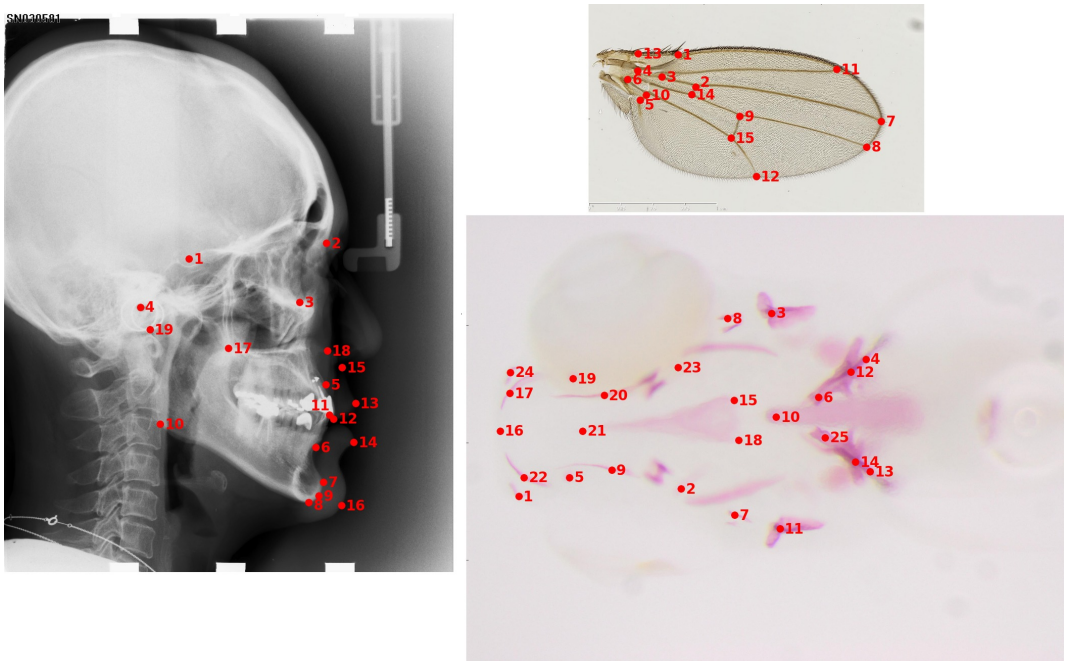


Figure 4.1: Sample image and corresponding landmarks for each dataset. CEPHA (left) with 19 landmarks, DROSO (top right) with 15 landmarks and ZEBRA (bottom right) with 25 landmarks.

- **ZEBRA**, a dataset of 113 ventral views of head skeleton of zebrafish larvae. The image resolution is 2576 by 1932 pixels. 25 landmarks were manually marked and reviewed by the same expert. Most landmarks of this dataset corresponds to locations with little visual information (1, 2, 5, 15, 18,...).

4.3 Methods

4.3.1 Algorithm description

The principle of this method is similar to the one presented in Chapter 3: we tackle the problem of landmark detection with a supervised learning approach: we exploit manual image annotations (ie. (x, y) positions of each landmark in the training images) to train recognition models for each landmark. These models are then used to predict each landmark position in new, unseen, images. The first difference comes with the fact that instead of just considering a classification approach, we also consider and compare two different learning methods for landmark detection: one based on pixel classification as we did in the previous chapter, and the other based on pixel distance regression. With the first one, the classification model is thus trained for each landmark separately to predict for each image pixel whether it corresponds to the position of the landmark or not. In the second method, a regression model is trained, also for each landmark separately, to predict for each image

	DROSO		CEPHA		ZEBRA	
#LDM	$\sigma(X)$	$\sigma(Y)$	$\sigma(X)$	$\sigma(Y)$	$\sigma(X)$	$\sigma(Y)$
1	57.8	62.8	45.1	44.2	85.8	70.7
2	43.3	48.9	63.7	65.7	105.6	78.4
3	47.7	73.1	49.9	58.3	95.9	80.1
4	50.1	92.3	38.4	38.3	108.2	84.2
5	43.1	89.0	68.2	67.4	88.2	75.0
6	47.6	100.7	85.6	78.0	107.9	71.9
7	51.1	119.5	96.5	89.1	92.4	69.3
8	53.7	110.3	96.2	91.6	94.0	79.3
9	41.1	28.8	97.0	90.9	91.7	64.4
10	42.5	84.6	61.0	62.6	104.5	67.3
11	63.6	83.2	76.2	76.5	96.6	70.8
12	67.3	34.6	74.2	74.9	106.3	81.1
13	58.9	92.9	66.8	79.6	111.4	85.1
14	41.0	50.8	81.3	82.3	106.3	80.2
15	46.4	29.9	62.9	77.0	99.8	66.4
16			98.8	92.9	84.7	104.1
17			47.4	49.2	87.2	69.3
18			60.7	74.3	97.3	64.9
19			38.5	43.9	90.0	68.3
20					90.6	67.9
21					100.8	64.0
22					86.0	69.7
23					106.5	94.2
24					85.7	68.8
25					107.4	71.1

Table 4.1: Standard deviation of each landmark, on both axes (in pixel units)

pixel its distance to the landmark. Regardless of the method (classification or regression), the models are trained from a learning sample composed of pixels extracted either in the close neighborhood of the landmark or at other randomly chosen positions within the training images. In this work, each pixel in the training sample is also described by a vector of visual features at different resolutions, the difference with our previous work being that several window descriptors will be considered. It allows to rely on local repeatable patterns and to disambiguate locally similar patterns using wider contexts.

The different steps of the algorithm for a single landmark are explained in the following subsections. The whole procedure is repeated for each landmark separately.

Extraction and description of pixels

Each observation in the training sample corresponds to a pixel at some position (x, y) in one of the training images. Each pixel is labeled by a discrete or a numerical output,

depending on the chosen method (classification or regression), and it is described by several input features. We list below successively the output associated to each pixel respectively in the classification and in the regression method, the input features used to describe them, and the pixel sampling mechanism.

Classification output. In principle, only one position in each image corresponds to the landmark, which means that if N training images are available, only N positive examples will be available to train our pixel classification model. To extend the set of positive examples, we consider as positive examples all pixels that are at a pixel distance at most R from the landmark, where R is a method parameter. More precisely, if the landmark is at position (x_l, y_l) in an image, then the output class of a pixel at position (x, y) in the same image will be 1 if $(x - x_l)^2 + (y - y_l)^2 \leq R^2$, 0 otherwise.

Regression output. With the regression method, the output associated to each pixel is the euclidean distance between this pixel and the landmark position in the training image. If the landmark is at position (x_l, y_l) in an image, then the output value of a pixel at position (x, y) in the same image will be its euclidean distance to the landmark d_l described in Equation 4.1:

$$\sqrt{(x - x_l)^2 + (y - y_l)^2} \tag{4.1}$$

Multi-resolution input features. Similarly to the previous chapter and in contrast to [80, 28, 59] where single-resolution features are extracted, in this work, we capture the context of the landmark at different scales and distances. A pixel at location (x, y) will be described by D multi-resolution square windows of resized height and width $2W + 1$ centered at its position (x, y), where W is a method parameter.

To this goal, images are downsized to D different resolutions prior to the windows extraction and the D resulting feature vectors are concatenated. For our images of size $m \times n$ pixels, these resolutions are described in Equation 4.2

$$\frac{m}{2^i} \times \frac{n}{2^i} \forall i \in [\![0..D]\!] \tag{4.2}$$

Out of image pixel values are set to zero. The influence of the chosen resolutions is shown in Section *Results*. An example of these windows is shown in Figure 4.2. We considered five ways of describing the multi-resolution windows:

- RAW: the raw pixel values of each resized windows are concatenated into a single vector. This will give a pixel descriptor of $D \times (2W + 1)^2$ features.
- SUB: the differences between the raw pixel values of the resized windows and the raw value of the pixel located at the (x, y) position. The pixel descriptor will thus also be of size $D \times (2W + 1)^2$.
- SURF: each window is described by the extended SURF descriptor [6], a descriptor previously proven to show robustness against rotations, scaling and noise. The

extended SURF descriptor consisting of 128 features, a pixel will be described by $128 \times D$ features.

- GAUSSIAN SUB: the differences between the raw pixel values of N_g pixels and the raw pixel value of the pixel located at the (x, y) position on each of the D resolutions, where N_g is a method parameter. The N_g pixels are chosen according to offsets from the (x, y) position. These offsets are chosen randomly according to the gaussian distribution $\mathbb{N}(0, \sigma)$, where σ is a user-defined parameter. In total, each pixel is represented by $D \times N_g$ features. Note that the window size W has no impact on this descriptor. Its role is taken by the parameter σ measuring the spread of the gaussian distribution.
- HAAR-LIKE: N_h Haar-Like features [88] of random size and position are randomly extracted inside each of the D windows, leading to $N_h \times D$ features.

Gaussian sub features were proposed in [28] and Haar-Like features were used in [58] to detect landmarks, in both cases however without multiple resolutions.

Pixel sampling scheme at training. Training a model on all pixels from all training images will be practically unfeasible in most cases and we will thus have to construct our training set by sampling the pixels. Uniformly sampling pixels from the training images will give however a very unbalanced learning problem for both classification and regression methods. For example, with a radius $R = 20$ pixels, only 1256 observations correspond, in classification, to positive examples, and in regression, to pixels within a distance < 20 to the landmark. This is very small compared to the whole size of the images (e.g., about 4 millions pixels for our images). To generate a more balanced training sample, we select $\pi\frac{R}{s}^2$ pixels within a radius R to the landmark in each training image, where s is a user-defined spacing parameter allowing to control the number of pixels extracted. $P\pi\frac{R}{s}^2$ additional pixels are then randomly selected outside this radius, where P a user-defined parameter.

In practice, one can expect in many medical and biological applications that the same landmark will be located in close positions from one image to another (see Figure 4.3 for an illustration on one of our datasets). At prediction stage, this information can be used to constrain the search for the landmark position to pixels that are not too far from the average position of the landmark in the training images. When this constraint is exploited at prediction stage, it is natural to avoid putting in the training sample pixels that are too far away from the landmark position. For this reason, we propose to select the $P\pi\frac{R}{s}^2$ pixels outside the radius R uniformly at random within a radius $R_{max} > R$ centered at the landmark position (see Figure 4.2 for an illustration).

This subsampling contrasts with [80] where pixels were sampled in the whole image during the training and the prediction phase.

Robustness to rotations. To improve robustness to rotations, and also to artificially increase the representativeness of the training data, we propose to expand our training

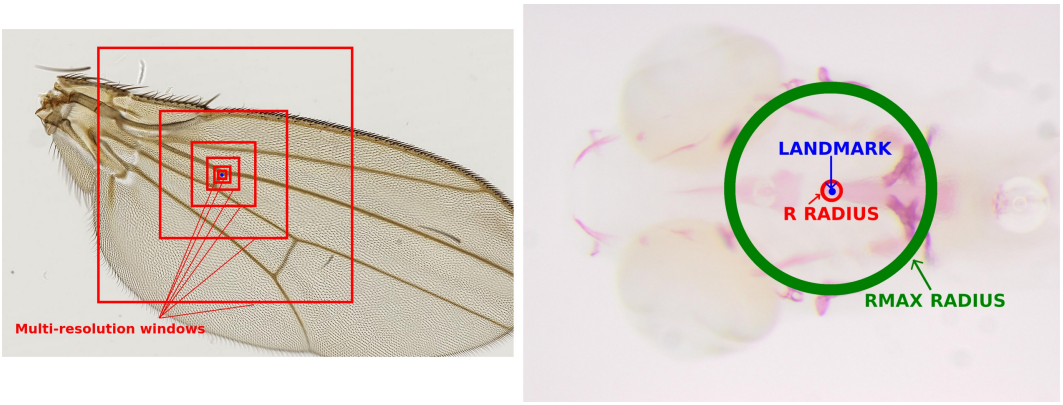


Figure 4.2: On the left, illustration of multi-resolution features representing one pixel (on the DROSO dataset, with $D = 6$ windows). The corresponding described pixel is located at the center of the windows (in blue). On the right, illustration of R and $R_{\max}$ radius (on the ZEBRA dataset). Observations in the R radius are considered as landmarks (positive) for the classification approach. At training, $P\pi R^2$ non-landmark observations are extracted in the $]R, R_{\max}]$ radius.

set by adding artificially rotated versions of the training images. More precisely, to the original training set, we add N_r new versions of each training image, each obtained by rotating this image by an angle randomly selected between $[-\alpha, \alpha]$, where N_r and α are two method parameters. With this operation, the total size of the training set will thus be multiplied by $N_r + 1$.

In the experiments, we will show that the problem of robustness to rotations is not important on our three datasets because the deformations stay small. On other datasets with bigger rotations, we would suggest to initially use a 2D registration algorithm such as [74] in order to roughly align the images between each other before analysis.

Classification and regression model training

To train the pixel classifier or regressor, we will also use the Extremely Randomized Trees algorithm [38] presented in Chapter 2. Note that with this particular algorithm and at the difference of the algorithm presented in the previous chapter, both the classification and the regression variants of the algorithm will be tested.

Landmark prediction

Let us denote by $\mu_l \in \mathbb{R}^2$ and $\Sigma_l \in \mathbb{R}^{2\times 2}$ the average and the covariance matrix of the landmark positions across respectively the training images. To make prediction of the landmark position with our tree-based pixel classifier or regressor, we proceed as follows:

1. We randomly draw N_p pixel positions from a multivariate normal distribution $\mathcal{N}(\mu_l, \Sigma_l)$.

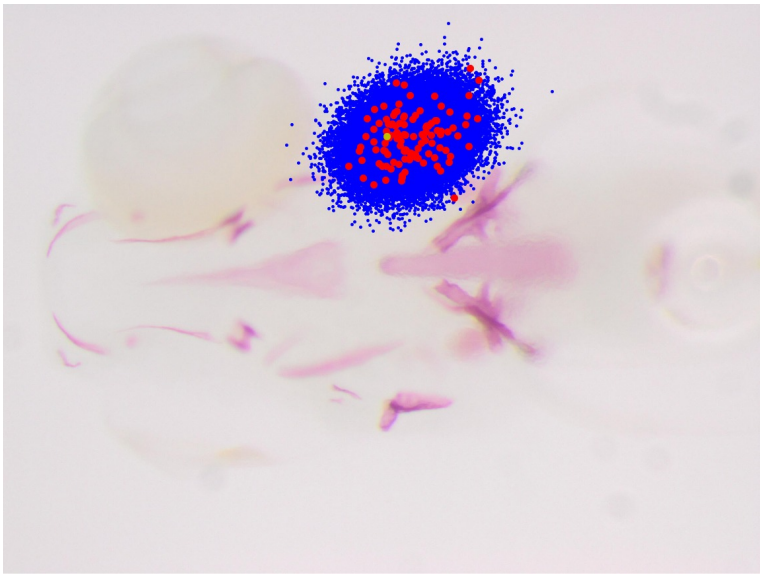


Figure 4.3: In red, the position of landmark 8 as observed in all the images of the ZEBRA dataset, overlaid on an image. In blue, the position of the corresponding 30.000 examples extracted during prediction according to our sampling strategy. In blue, the real landmark position.

2. We apply the classification or regression model on each of the resulting pixels.
3. The predictions for these pixels are then aggregated as follows to obtain the final predicted landmark position:
    - Classification: the final position is taken as the median position among the pixels that are predicted as being the landmark with the highest confidence by the tree-based model (i.e, which receives the highest number of votes for the positive class from the T trees in the ensemble).
    - Regression: the final position is taken as the median position among the pixels that are predicted as being the closest pixels to the real landmark position (i.e, for which the predicted distance to the landmark position is the smallest).

The subsampling scheme of the first step is illustrated in Figure 4.3. Such subsampling allows to improve predictive performance by reducing the probability of generating spurious landmark predictions at irrelevant positions in the images. It also considerably speeds up the algorithm with respect to an exhaustive scan of all image pixels as it was performed in [80].

Summary of the algorithm

Training. An illustration of the different steps for training a single landmark classifier is given in Figure 4.4. For each of the training images, landmark pixels are extracted within

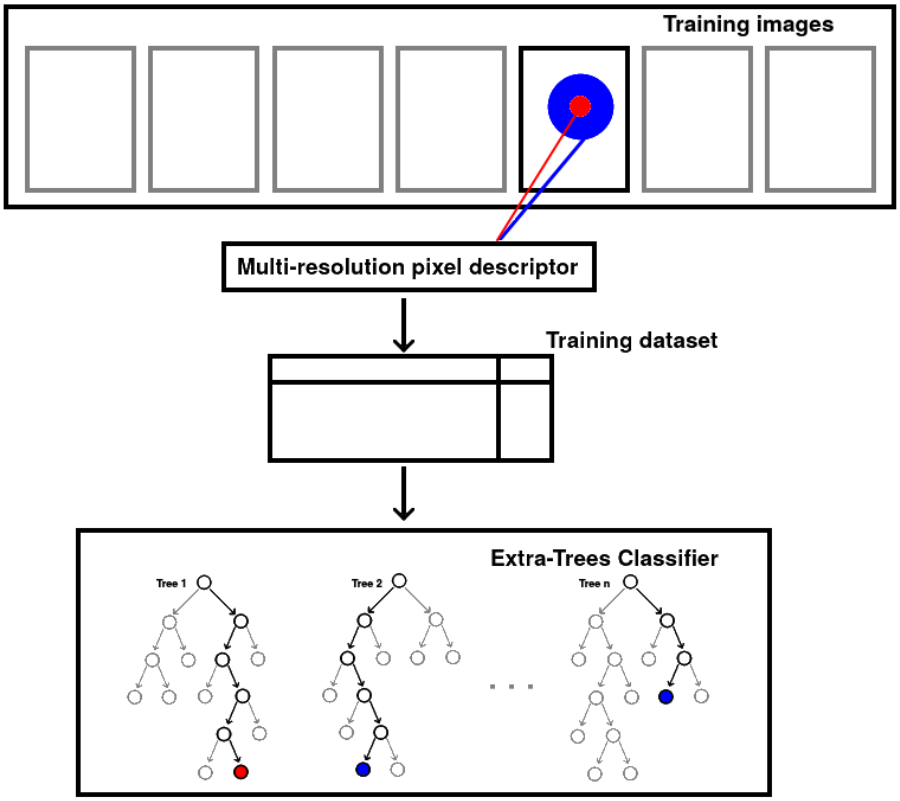


Figure 4.4: Summary of the training phase.

a range R of the true landmark location. Non-landmark pixels are extracted within the range $]R, R_{\max}]$ from the landmark position. These pixels are described using one among five different multi-resolution pixel descriptors (RAW, SUB, GAUSSIAN SUB, HAAR-LIKE, SURF). The dataset obtained is then used to train an extra-trees classifier. In the regression setting, the same procedure is applied, except that the output is numerical and set to the euclidean distance between the pixel and the landmark position.

Prediction. A summary of the prediction phase for a single landmark is illustrated in Figure 4.5. For a new image, pixel locations are extracted from a gaussian distribution trained from the landmark positions in the training dataset. These pixels are described by the same multi-resolution pixel descriptors that were used at training. The classifier or the regressor is then used to score each of these pixel locations. The median of the pixel locations with the highest (classification) or lowest (regression) score is considered as the final predicted landmark location.

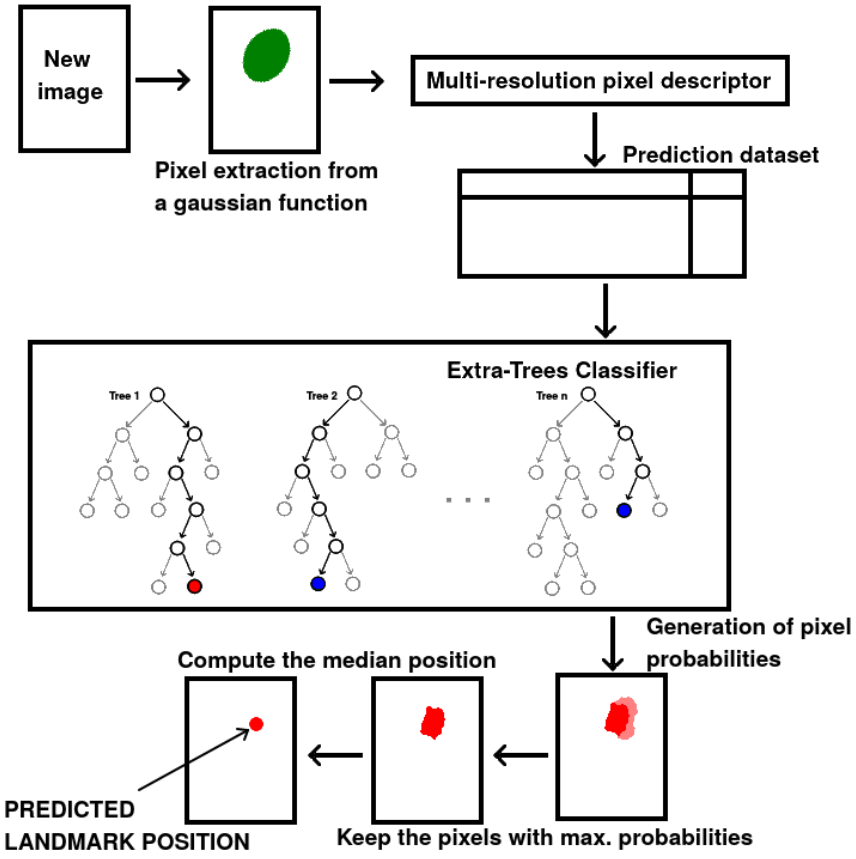


Figure 4.5: Summary of the prediction phase.

Implementation Details

Algorithms were implemented in Python using the scikit-learn library [70] for its efficient implementation of the Extremely Randomized Trees algorithm and the OpenCV library [10]. All algorithms were further integrated into the open-source Cytomine web software [66] using its software template mechanisms. End-users interested to interact with a Cytomine web-server through its web interfaces can read the Cytomine user guide [65] and documentation [1], where complete instructions for using Cytomine and our landmark detection algorithms are given.

Most of the results presented in this chapter were obtained on several computer clusters. About 20000 cluster jobs were needed for the complete cross validation of our algorithm, which represents about 10000 hours of computing time.

Note that on a regular computer (8×2.8Ghz, 8Go RAM), our implementation takes approximately 1 minute to build a model for a single landmark, and approximately 1 second to detect a single landmark on an image with $N_p = 50000$. A typical model

[1] `http://doc.cytomine.be`

Parameter	Description	Default Value
W	The size of the multi-resolution window	8
R	The distance to the landmark position determining the training pixel output class	15 (CEPHA) 9 (DROSO) 20 (ZEBRA)
s	The spacing between the landmarks extracted inside the R radius	2
$R_{\max}$	the maximal distance to the interest point to extract non-landmark observations	600 (CEPHA) 300 (DROSO) 1000 (ZEBRA)
P	The ratio of negative versus positive examples sampled during training	1 (CEPHA) 2 (DROSO) 1 (ZEBRA)
N_p	The number of pixels randomly extracted during prediction	30000
N_r	The number of rotated versions of each training images that are introduced in the dataset	3
α	The maximal rotation angle (in degree)	30
D	The number of resolutions introduced in the feature representation of each window	5
T	The number of trees	50
F	The feature type used to describe the windows	RAW

Table 4.2: Description and default values of our method's parameters at validation

$(R = 15, P = 2, T = 100, D = 5)$ with ± 100 images takes 2 gigabytes of RAM. Reducing the values of these parameters can lead to significant speed-ups, but it can also lead to a significant decrease of accuracy (see section *Results*).

4.4 Results and Discussion

In this section, we will first study the behavior of our parameters through 10-fold cross validation. The goal of these experiments is to evaluate the influence of the method parameters and to use our three datasets in order to extract guidelines for their initial setting in future applications. We will then compare our results with existing algorithms. Finally, we will discuss the results and extract some guidelines for landmark detection on bioimage datasets.

4.4.1 Influence of the method parameters

Experimental protocol

We used 10-fold cross validation to perform our experiments. On each dataset, when one parameter was tested, all the others were fixed to default values given in Table 4.2.

In our experiments, we consider only the euclidean distance of the predicted landmark

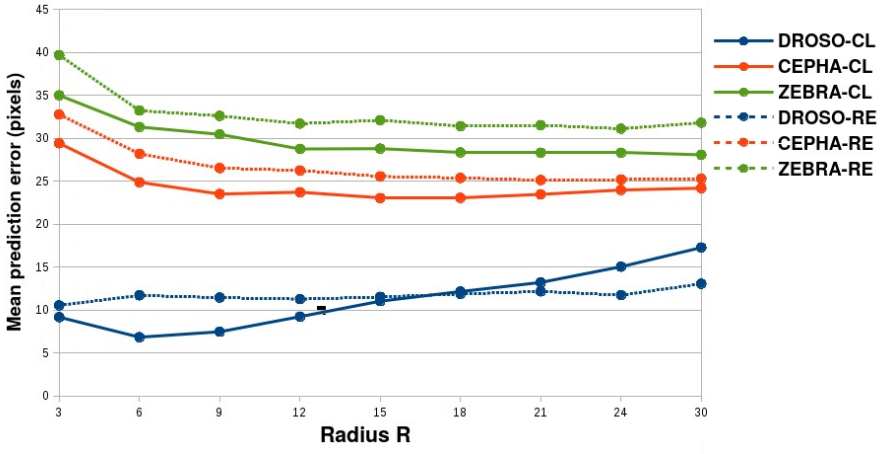


Figure 4.6: Influence of the parameter R of our algorithm.

position to the real landmark position as an error criterion. Given the fact that we have images of different sizes, and, as we will see, very different performances between the image datasets, this criterion was the easiest to use and optimize. Given the comparisons made in the previous chapter, where the performance order was the same for each criterion, we also suspect that we can assume the trends will follow for each of the error criterions that were considered. Moreover, the euclidean distance has the advantage to be applicable to any other dataset and to be easily interpretable.

Results

Before discussing our findings, it is interesting to note that landmarks from the DROSO dataset are always detected with a significantly higher accuracy than landmarks from the two other datasets. This result is not surprising when looking at the sample images in Figure 4.1. Indeed, landmarks from the DROSO dataset are clearly located at borders and intersections and are thus already easier to detect by human experts.

The influence of the **radius R** is presented in Figure 4.6. In **classification**, on CEPHA and ZEBRA, the higher R the better. On DROSO, where the landmarks are easy to detect, increasing R too much leads to a loss of accuracy. We explain this phenomenon by the fact that DROSO landmarks are intersections and edges, thus making the landmark position a highly informative position. Increasing R can thus only increase the confusion with close pixels. On the three datasets, a too small R has a negative impact on accuracy. This is probably due to a reduction in the number of the training examples. For CEPHA and ZEBRA, increasing R improves the accuracy. For these datasets, this increase allows to consider close well-defined structures and implicitly take into account the uncertainty on the exact location of the landmark (due to variation with the manual annotations by human experts). The radius R has less impact in **regression** than in classification although the main trends are similar. Note that the regression approach is expected to be less impacted by R because of its continuous outputs while this directly affects the binary output in

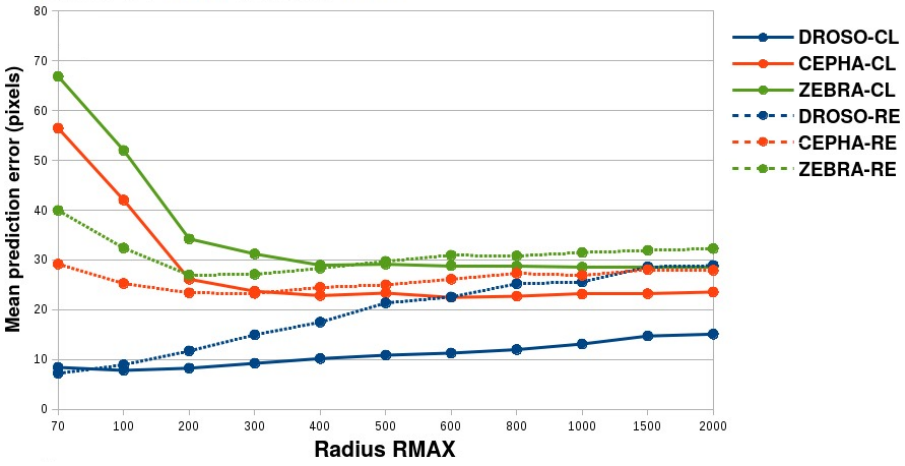


Figure 4.7: Influence of the parameter $R_{\max}$ of our algorithm.

classification.

The influence of the **radius** $R_{\max}$ is presented in Figure 4.7. In **classification**, we noticed quite similar effects on the three datasets: a small $R_{\max}$ can lead to confusion at prediction time with pixels far from the landmark, while a large $R_{\max}$ increases the probability of confusion with close pixels, because there will be proportionally less pixels close to the landmark in the training set. On ZEBRA and CEPHA, large values of $R_{\max}$ work better, suggesting that there is more confusion with remote pixels. On DROSO, smaller values are preferable, suggesting there is more confusion with closer pixels. This parameter thus needs careful tuning although $R_{\max} \in [200, 300]$ seems to bring results close to the optimum. The main trends are similar in **regression**, however in this case the error clearly increases on all three datasets when R_{max} is increased too much. This might be explained by the fact that increasing R_{max} directly increases the range of the output values (i.e., distances) considered during model training. This could have a negative impact on the prediction error for small distances that are the only predicted distances used to determine the final position of the landmark.

The influence of the **ratio** P is presented in Figure 4.8. In **classification**, increasing the ratio P of negative versus positive examples significantly improves the results on the DROSO dataset, while, even if it is positive, the impact is more subtle on CEPHA and ZEBRA. Actually, looking at the same curves for individual landmarks (results not shown), we notice that increasing P has a negative impact for some landmarks on these two datasets. On DROSO, because landmarks are easier to detect, increasing P will decrease the risk of confusion with pixels outside the R radius. On CEPHA and ZEBRA, giving more weights to pixels outside the radius R increases the chance not to detect as positive pixels inside the radius, which has a negative impact on accuracy. The trends are very similar in **regression**: on DROSO, the larger P, the better, while on CEPHA and ZEBRA, the optimum value is landmark dependent.

The influence of the **number of trees** T is presented in Figure 4.9. For both **classification**

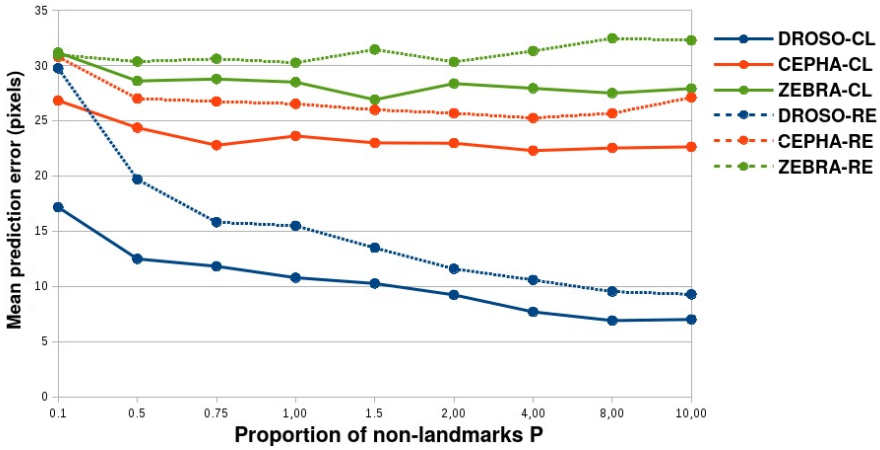


Figure 4.8: Influence of the parameter P of our algorithm.

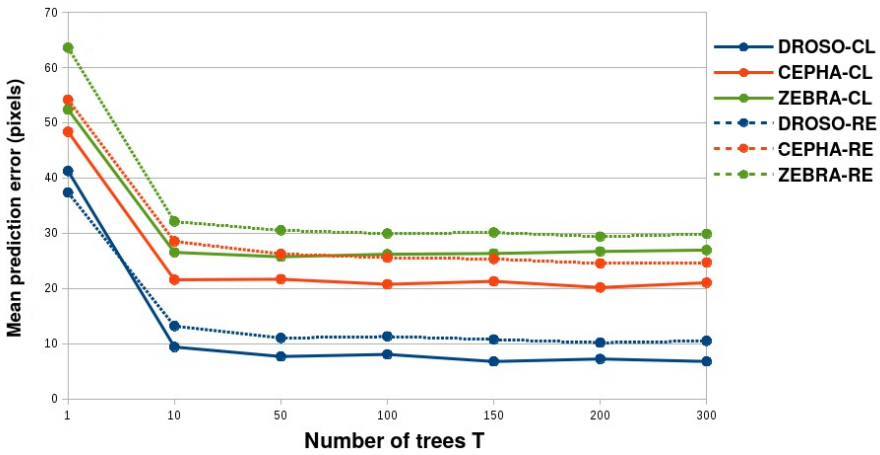


Figure 4.9: Influence of the parameter T of our algorithm.

and **regression**, the impact of increasing the number of trees is always positive as expected. We can observe however that increasing the number of trees beyond 50 does not bring improvement.

The influences of the **number of rotations** N_r and the **maximum value of the angle** α are presented in Figure 4.10. The rotations do not seem to have an impact on the (mean) error. This suggests that our pixel descriptors are robust enough to orientation changes in our three datasets. Additional experiments about the robustness of our pixel descriptors to rotations is supplied in the Results section of this chapter.

The influence of the **number N_p of pixels tested at prediction** is presented in Figure 4.11. On our three datasets, we observe that N_p should be at least 10.000 to reach convergence. This number is equivalent to a complete search in a window of size 100×100 (i.e respectively $0.77, 0.21$ and 0.2 of the full images on the DROSO, CEPHA and ZEBRA datasets). Increasing N_p beyond 10.000 does not significantly improve the results.

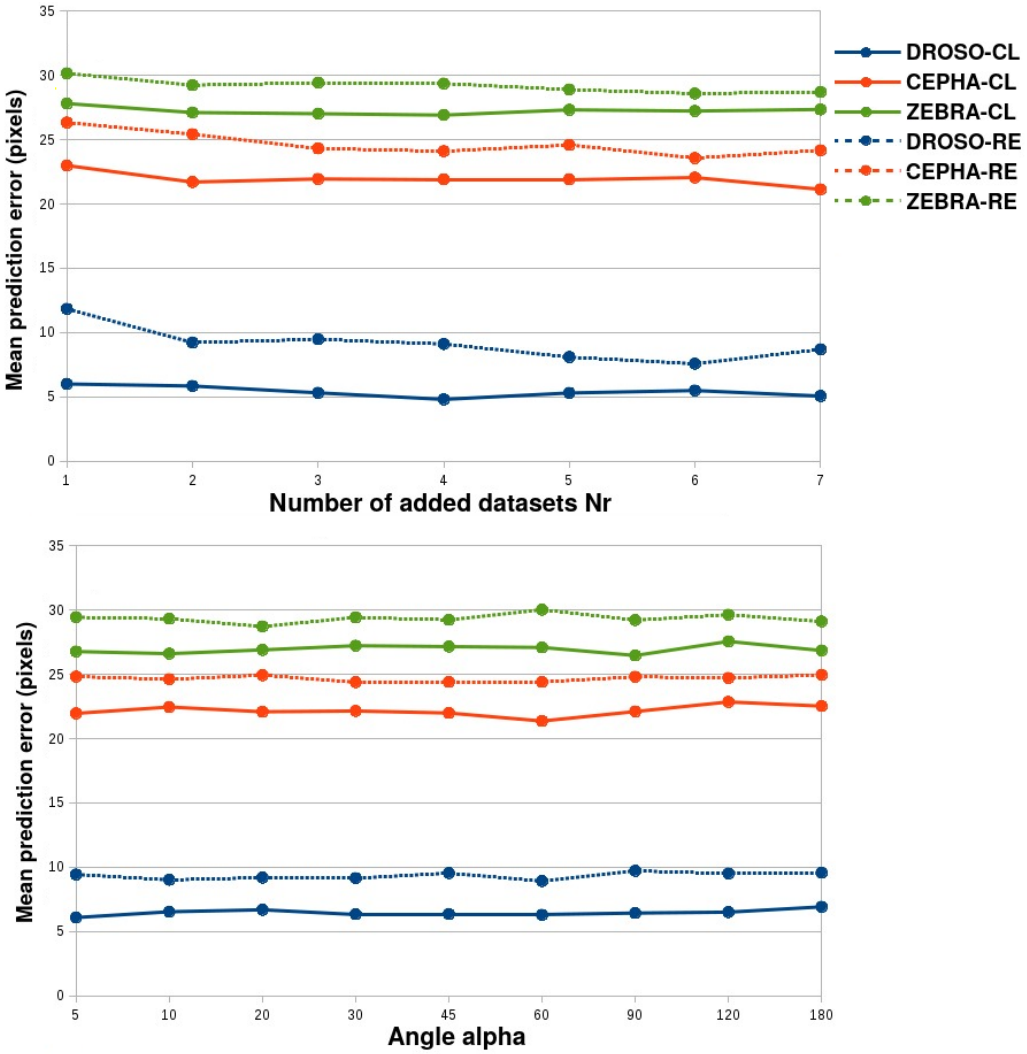


Figure 4.10: Influence of the parameters N_r and α of our algorithm.

The influence of the **window size** W is presented in Figures 4.12 and 4.13 respectively for RAW and HAAR-LIKE features. For this parameter, we made a distinction between RAW and HAAR-LIKE features, because the size of W does not influence the size of the feature vector when using the latter type of features. Also note that when using GAUSSIAN SUB features, the algorithm is not influenced by this parameter. When using RAW features, increasing W up to 8 seems to significantly improve the results for the classification and regression approaches. Then, the marginal improvement decreases and, for DROSO, becomes negative. While small windows do not contain enough information, larger ones could create an overfitting problem due to the quadratic increase in the number of features. Because this quadratic increase will also affect the computational cost of our problem, we did not consider higher values. When using HAAR-LIKE features, the size of W still needs to be kept small, higher resolution information being provided by the use of

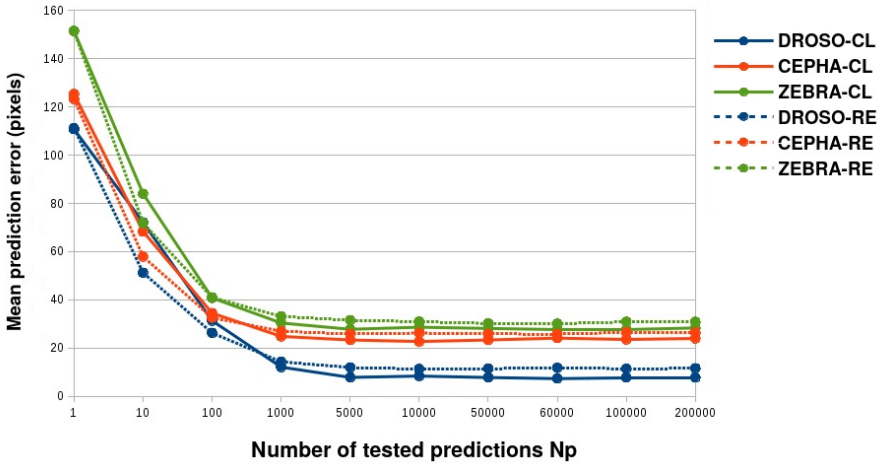


Figure 4.11: Influence of the parameter N_p of our algorithm.

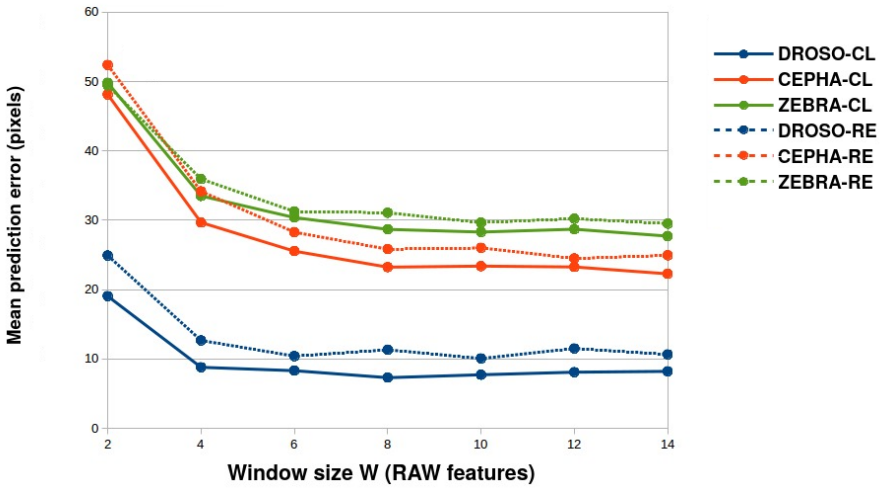


Figure 4.12: Influence of the parameter W of our algorithm.

multi-resolution windows. Optimal W values are in this case included between 15 and 20. This experiment was performed using the same number of descriptors per pixels as when considering RAW feature descriptors (1536).

The influence of the **window descriptor features F** is analysed in Figure 4.14. In this figure, we also added single resolution features **SR** in order to analyze the influence of our multi-resolution approach. The best resolution was validated among the 6 resolutions used by our pixel descriptor. The σ of GAUSSIAN SUB (SR) was validated among 6 values (10, 25, 50, 100, 200, 400). The W of HAAR-LIKE (SR) was validated among 6 values (8, 20, 50, 100, 200, 400) and the W parameters used with other features was also validated among 6 values (8, 15, 20, 25, 30, 40). The total number of features for HAAR-LIKE (SR) and GAUSSIAN SUB (SR) was set to be the same as the number of descriptors for RAW

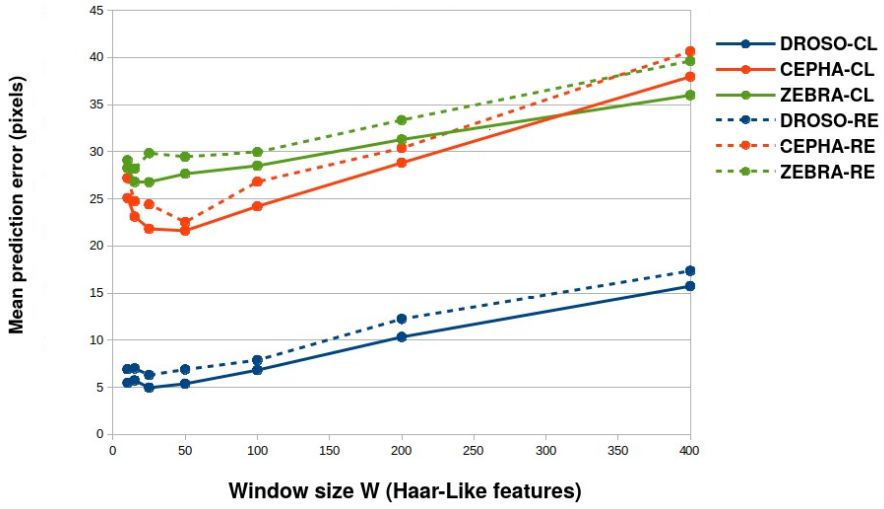


Figure 4.13: Influence of the parameter W with Haar-Like features of our algorithm.

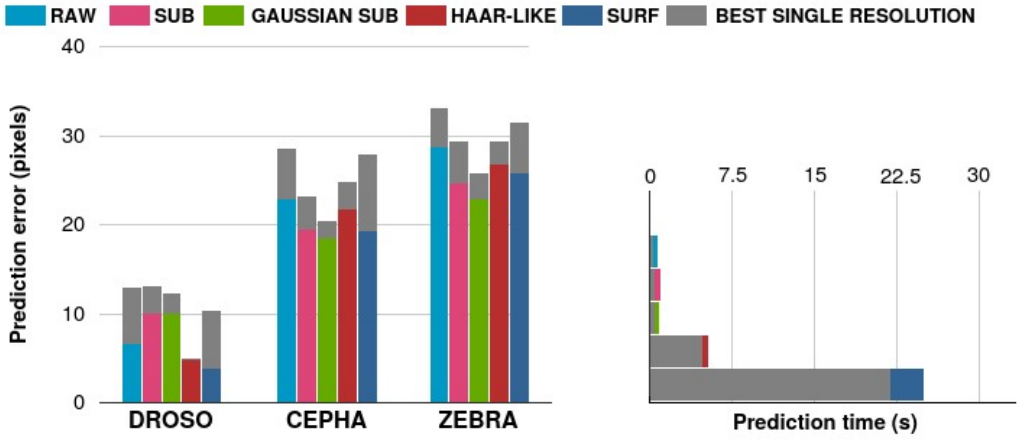


Figure 4.14: Influence of the use of the window descriptor features using classification. Above: the mean error using each window descriptor on each dataset. Below: the time needed for extracting 10.000 observations using each window descriptor. Grey bars represent the results obtained by using only the best resolution found during a 10-cross validation process.

and SUB (1536 descriptors).

From Figure 4.14, we can conclude that using multi-resolution features always improves the performances while only slightly increasing the prediction time (it takes about 1 second for the resizings of a 2576×1932 image in our python implementation when $D = 5$).

Subtracting the value of the central pixel (SUB, GAUSSIAN SUB) instead of using raw pixel (RAW) values has a positive impact on both CEPHA and ZEBRA, while it clearly has a negative impact on DROSO. This is most probably due to the particular nature of

the landmarks on the DROSO dataset. Landmarks in this dataset mostly correspond to borders and intersections that are rendered more difficult to detect when centering the pixel values. The SURF descriptor always performs well when using multi-resolution windows, but is only significantly better on the DROSO dataset. The GAUSSIAN SUB descriptor obtains the worst results on the DROSO dataset, but the best on the CEPHA and ZEBRA datasets. This suggests this feature descriptor is only efficient for landmarks located in areas with low visual information. The opposite goes for the HAAR-LIKE pixel descriptor, which is the most efficient on the DROSO dataset.

On the DROSO dataset, multi-resolution Haar-Like features seems to be the best option to use. Given the good performances of the single resolution Haar-Like algorithm, its use could also be advised. We explain the small difference in terms of performances on the DROSO dataset by the fact that most landmarks are located on corners or intersections, which makes them easier to detect, even on a single resolution. For the same reason, gaussian subtraction seems to have difficulties on this dataset: gaussian offsets are less useful when using local information.

On the CEPHA dataset, gaussian subtraction seems to be the best option, closely followed by SUB, which is window pixel subtraction. In this dataset, landmarks are not always defined at intersections or edges, and thus the use of pixel-wise context information becomes more interesting. We also noticed that SUB could obtain very accurate detection of well-defined landmarks while GAUSSIAN SUB had more difficulties to reach this accuracy.

On the ZEBRA dataset, multi-resolution subtraction GAUSSIAN SUB and window subtraction SUB are also the best feature descriptors to use. This shows these multi-resolution feature descriptors are the most adapted to detect landmarks that are not presented at corners or intersections.

From the results, we can also conclude that while RAW and SUB do not obtain the best results on any of the datasets, these basic descriptors always obtain good performances. This is interesting when considering GAUSSIAN SUB and HAAR-LIKE: while they obtain the best performances on a given dataset, they obtain the worse for another. Given their small prediction time, we thus think RAW and SUB can be used to reach good performances on any dataset.

The influence of the **number of resolutions** D is presented in Figure 4.15. In both **classification** and **regression** and for all datasets, increasing D first leads to a strong reduction of the error. At some point however, the error starts increasing. In average over all landmarks, the best performance is obtained by using $D = 6$ on our datasets.

4.4.2 Comparison with other algorithms

In this section, we compare the results obtained by our method with several landmark detection algorithms. Because there is no available implementation of these algorithms, these comparisons are divided in two parts. First, we compare our method with our own implementation of the algorithms presented in [28] and [59] on our three datasets, using

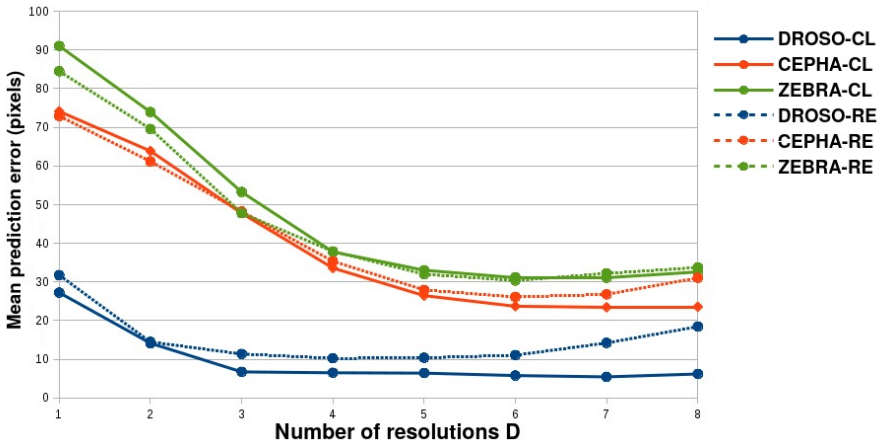


Figure 4.15: Influence of the parameter D of our algorithm.

only the original number of landmarks. Second, we present the results obtained by all algorithms on two cephalometry datasets used during international landmark detection challenges [91, 90].

Comparisons on our datasets

We compared our algorithm with our own implementations of the algorithms presented in [59] and [28]. These methods are described in Section 2.5, page 22. A quick summary is proposed in the following paragraphs.

The method presented in Lindner and Cootes's paper[59] (that we call **LC**) is divided in two phases: first, individual landmark offset regressors are trained. These landmark offset regressors are then used to build vote maps giving the likeliest position of the landmarks. In a second step, these vote maps are combined with a PCA based model of the landmark shapes in order to build an active shape model for which an optimal configuration can be found through an iterative process. In a related study, they further improved the method performances on cephalometric data by considering a larger dataset and evaluating the accuracy of two different experts annotating the images [60].

In Donner et al.'s paper [28] (that we call **DMBL**), a three step approach is proposed: first, a random forest classifier is trained to classify pixels. For N landmarks, the classifier associates to each pixel one class among $N + 1$ classes: either the type of a landmark or the background. This first step will be used to build a probability map for each of the landmarks. These probability maps will be refined into a small number of candidates for each landmark by using landmark offset regressors. The final position of the landmarks will be chosen among the candidates of each landmark using a Markov Random Field based on the distances between the landmark positions.

We estimated the results obtained with these two methods and ours on each dataset and landmark. We divided our datasets into a training and a test set. For each dataset, the

Algo	Tested Values	DMBL	Tested Values	LC	Tested Values
R	$6, 9, 15, 20$	T (phase 1)	50	PCA reduc	$1, 25\%$, $50\%, 100\%$
$R_{\max}$	$100, 300, 500$	F (phase 1)	32	$d_{\max}$	$50, 100$, $200, 500$
P	$1, 3$	R (phase 1)	$3, 5$	N_s	1000
N_p	10.000	σ	$10, 50, 100, 200$	W	$100, 200, 400$
N_r	$1, 3$	δ	$0.25, 0.5$	n	1600
α	$30°$	P	$1, 0.5N$	T	50
T	50	R	$10, 20, 50, 100$	step	$4, 8, 16$
D	6	N_s	1000	$R_{\max}$	$100, 300$, $500, 1000$
F	RAW,SUB GAUSSIAN SUB, HAAR-LIKE	T	50	$R_{\min}$	$1, 2, 10, 20$
		F	32	α	$0.1, 0.5, 0.9$
		Filter Size	$3, 10$		
		β	$0.2, 0.5$		
		#Iterations	$1, 3, 5$		
		#Candidates	$1, 5, 10$		
		#Edges	$0.1N, 0.5N, N$		

Table 4.3: Parameters tested during cross validation on the three datasets for our method, DMBL [28] and LC [59].

methods were tuned on the training set using 10-fold cross validation. The models were then built using the complete training set and then evaluated on the test set. We chose to use half of the dataset images as learning set, and the other half as test set. For parameter exploration during CV, we used a grid search where some common parameters were fixed for a fair comparison: the number of trees, used in all methods, was fixed to 50. The number of descriptors for a pixel was also roughly fixed: our algorithm used 1536 features while LC and DMBL were set to use 1600. The tested values are presented in Table 4.3. The algorithms were all implemented in Python.

The results are presented in Figure 4.16. We obtained the best averaged error on each of the three datasets. On the DROSO dataset, that seems to be the easiest for all the algorithms, DMBL performs clearly worse than our algorithm and LC. We explain this by the feature engineering choice made in DMBL, whose pixel descriptors focus less on local appearance. On DROSO, our algorithm has only a small advantage compared to LC. However, our algorithm obtains the best performances for 10 landmarks, LC 5, and DMBL 0. On the CEPHA dataset, LC seems to obtain worse results than our method and DMBL. From our observations, we suspect that our algorithm performs better due to unreliable landmark candidates for DMBL and the correction phase of LC, that imposes too much constraints on the possible shapes described by the landmarks. Our algorithm obtained the best performances for 12 landmarks, LC 1, and DMBL 6. On the ZEBRA

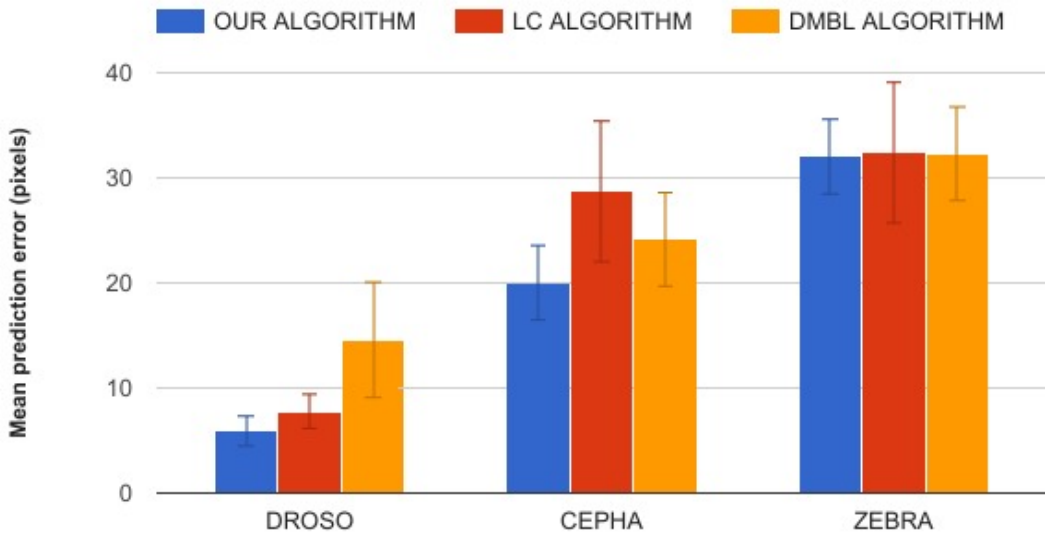


Figure 4.16: Comparison of our algorithm with [59] and [28] on our three datasets. Error bars corresponds to 95% C.I.

dataset, the results of the three algorithms are really close, but our algorithm obtains the best mean error. Our algorithm obtained the best performances for 12 landmarks, LC 6, and DMBL 7. We also suspect that the landmark structures are not defined well enough when using small training image datasets (69 images for DROSO, 50 for CEPHA and 57 for ZEBRA), thus making the refinement steps of LC and DMBL less useful.

Comparisons on the ISBI cephalometry challenges

We considered a comparison with the preliminary version of our algorithm that was presented in the context of the 2014 ISBI challenge [87, 91]. The goal of this challenge was to predict the position of each of the 19 cephalometric landmarks presented above as accurately as possible. Here we focus on the comparison of our algorithm with the best results obtained during these two challenges. For the 2014 challenge, we compared our algorithm with its preliminary version that ranked second at the challenge, and with the challenge best performer [43], a method proposed by Ibragimov et al. (called **ILPV** in the rest of the chapter). ILPV combines random forests with game-theoretic tools that take into account relations between landmark positions. To improve their performance at the challenge, ILPV furthermore manually created new landmarks that they incorporated in their training phase. The main difference between our preliminary and proposed method are the parametrization of the sampling at training and prediction, the use of rotations and the possibility to use SUB, HAAR-LIKE and GAUSSIAN SUB descriptors. We also compared our method with the 2015 ISBI challenge two best performers: ILPV that ranked second, and LC that ranked first. In the context of this challenge, they also

used 10 additional manually annotated landmarks in order to ensure the consistency of their active shape model optimization problem. Note that LC is one of the state-of-the-art algorithms we tested in the previous section (but without any addition of manual landmark annotations).

The landmark by landmark comparison between the three methods for both challenges is presented in Figure 4.17. This comparison was performed according to the challenge rules: we used the initial training sets (different for each challenge) composed of 100 images for the 2014 challenge and 150 images for the 2015 challenge. We tuned the parameters by 10-fold internal CV on the training set, trained the final models with the optimal parameters found on the whole training set, then compared the predictions of the different methods on the test set composed of 100 new images for both challenges. For parameter tuning, we used the same grid search as in the previous section. For the first challenge, the global mean accuracy of the preliminary version of our algorithm is 21.84 pixels. ILPV obtains 18.989 and the method proposed in this chapter 17.55. For most of the landmarks (18 out of 19), we observed an improvement between the previous and the novel variants of our algorithm. This shows that in some cases, using multi-resolution raw pixel values can help to obtain a better accuracy. On 11 out of 19 landmarks, our method works also better than ILPV, while it is outperformed on 8 landmarks. Given ILPV requires some additional manual annotation effort to reach this performance, we believe that our results are very competitive. For the second challenge, the global mean accuracy of the 1st method (LC) is 16.74 pixels and the second (ILPV) 18.46 pixels while we obtained a mean accuracy of 17.79 pixels. We obtained the best results for 3 landmarks, the second best results for 10 other landmarks. As we show in Figure 4.17, it is clear that the differences are also small between the different methods. In comparison to these two other methods using additional landmarks to refine the landmarks shape, we think our algorithm, without additional annotation and refinement, is competitive with the state of the art.

Comparison of speed and memory consumption

Table 4.4 compares the four algorithms in terms of training and prediction speed as well as memory occupation. In terms of **memory occupation**, at **training**, our algorithm will build N classification or regression mono-output models. These models will be built by extracting around N different datasets of pixels. LC will build N bi-output regression models as well as a PCA model. In our experiments, we extracted approximately the same number of pixels for this algorithm and ours. In DMBL, 1 classification model will be built for phase 1 and N bi-outputs regression models. In our experiments, each of these models were built using approximately the same number of pixels as in our algorithm. In addition, a Markov Random Field will be built. We did not reimplemented ILPV, but given its model features, we can reasonably estimate that a model would ask for approximately the same number of pixels than our algorithm. This means that, at training time, DMBL extracts the largest amount of pixels while the three other algorithms use approximately the same amount. At **prediction**, we showed we could already reach our best performances by extracting around 10.000 pixels per landmark, while the number of pixels to extract will depend on image size for LC, DMBL and ILPV. DMBL extracts height $\times$ width $\times \delta^2$ pixels,

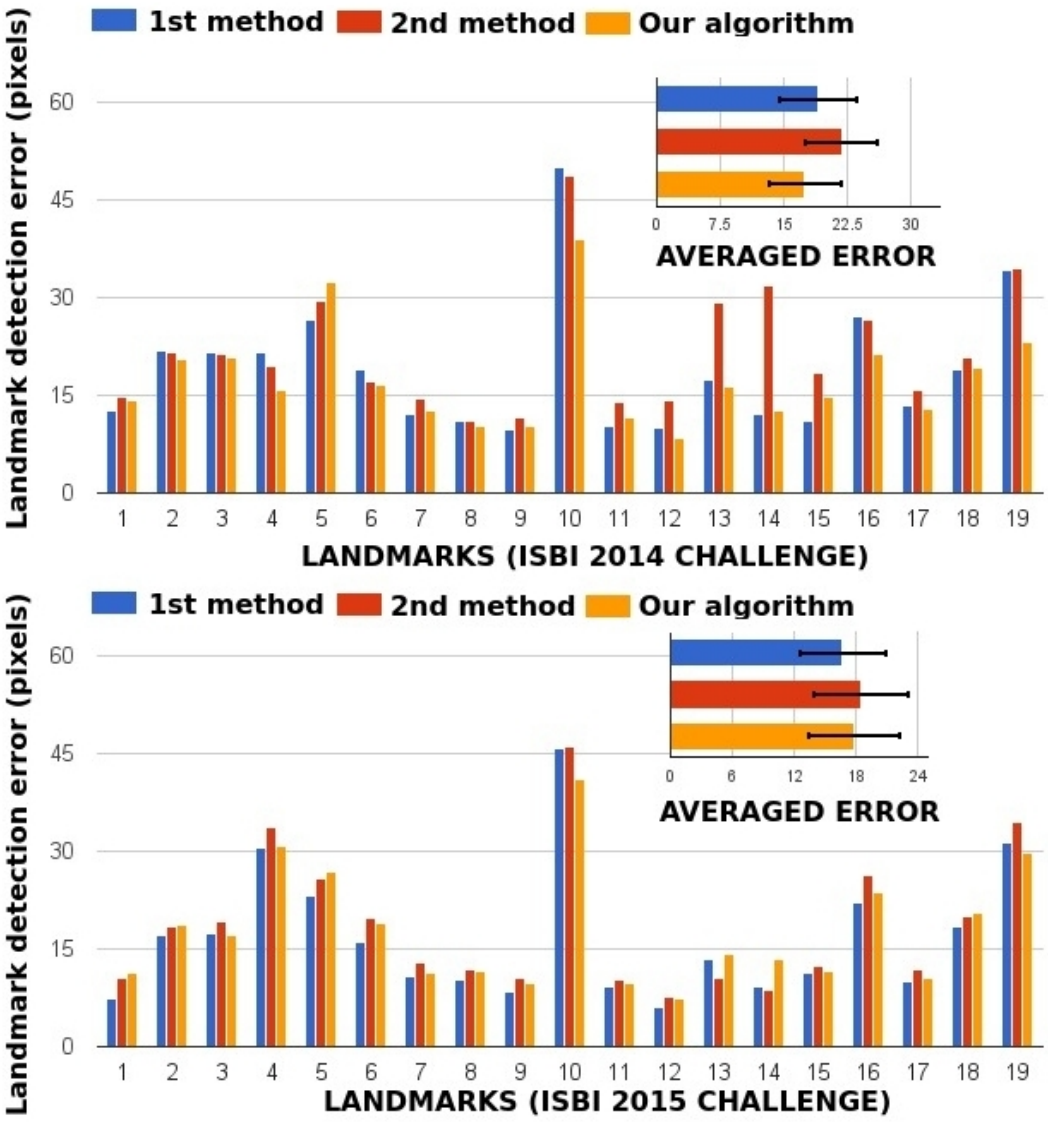


Figure 4.17: Comparison with 2014 and 2015 ISBI Cephalometric X-Ray Challenge best results. Error bars corresponds to 95% C.I. For 2014, 1st method is ILPV. 2nd is an initial version of our algorithm. For 2015, 1st method is LC and 2nd ILPV.

where the optimal δ we found was 0.5. LC extracts $\frac{\text{height}\times\text{width}}{\text{step}^2}$, and we found an optimal step of 4. In the cephalometric challenge, ILPV extracts $\frac{\text{height}\times\text{width}}{\text{step}^2}$, where step$=3$ for the cephalometric challenge. In addition, DMBL will need to keep probability maps of size $N \times \text{height} \times \text{width} \times \delta^2$, LC's vote maps of maximal size $N \times \frac{\text{height}\times\text{width}}{\text{step}^2}$ and ILPV probability maps of size $N \times \frac{\text{height}\times\text{width}}{\text{step}^2}$. Given the size of our images and the variance of the landmark positions, our algorithm uses a significantly smaller amount of memory than other methods on our three datasets. Note however that with smaller images or more variable landmark positions, the memory requirement and time consumption of our method

might exceed those of LC and ILPV. In terms of **speed**, at **training**, DMBL will need to extract slightly more pixels than the other algorithms, and build one additional forest. LC, DMBL and ILPV are building additional models based on the landmarks coordinates. This is why our algorithm can be considered as faster than the other algorithms at training. However, if parallelization capacities are available, all the models can be built at the same time, and the same goes for the pixel extraction. In this context, the bottleneck will be the difficulty of the pixel extraction, which will make LC, ILPV that both use single resolution Haar-Like features and/or our algorithm if we use SURF features the slowest algorithm. At **prediction**, given that we extract less pixels and do not use an additional refinement step, our algorithm should be the fastest as long as we are not using SURF features. Another bottleneck comes from the refinement step(s) because these steps can not be parallelized between each other. This is still playing in favor of our algorithm which has no refinement step.

4.4.3 Robustness analysis

In this section, we will analyze the influence of the deformations in the images on the accuracy of our method.

We define the deformation of an image i, d_i as the euclidean distance between its landmarks and the mean shape (the mean position of the landmarks). This deformation is computed once the shapes have been centered in order to reduce the impact of translation:

$$\bar{x}_{i,l} = x_{i,l} - \frac{1}{L}\sum_{j=1}^{L} x_{i,j}, \bar{y}_{i,l} = y_{i,l} - \frac{1}{L}\sum_{j=1}^{L} y_{i,j} \tag{4.3}$$

$$d_i = \sqrt{\sum_{l=1}^{L}(\bar{x}_{i,l} - \frac{1}{N}\sum_{k=1}^{N}\bar{x}_{k,l})^2 + (\bar{y}_{i,l} - \frac{1}{N}\sum_{k=1}^{N}\bar{y}_{k,l})^2} \tag{4.4}$$

Where L is the number of landmarks and N the number of images. In order to keep the deformations comparable between the datasets, image heights and widths were set to 1, and the number of landmarks was fixed to $L = 10$. These L landmarks were selected randomly.

The deformation distribution of both approaches is given in Figure 4.18. From this figure, we can conclude that the deformations in the DROSO dataset are more important than in CEPHA and in ZEBRA.

Figure 4.19 shows the influence of the importance of the deformation on the error when the distance to the mean shape criterion is used. As it could be expected, RAW, SUB and GAUSSIAN features have more difficulties to handle large deformations than HAAR and SURF features. Donner's algorithm (DMBL) also seems to encounter difficulties with bigger deformations. Haar-Like features seems to be the less impacted by the deformations along with Lindner's algorithm (LC).

	Our algorithm	Lindner et al. [60]
TRAINING	**PIXEL EXTRACTION** N datasets - 1 dataset ± 100.000 pixels Extraction can be parallelized **MODEL BUILDING** N single output classification OR regression forests **ADDITIONAL OPERATIONS** Each training image must be resized D-1 time(s) If HAAR-LIKE features are used, integral images need to be built	**PIXEL EXTRACTION** N datasets - 1 dataset ± 100.000 pixels Extraction can be parallelized **MODEL BUILDING** N double-output regression forests 1 PCA model (using landmark coordinates) **ADDITIONAL OPERATIONS** Integral images need to be built
LANDMARK DETECTION (for 1 image)	**PIXEL EXTRACTION** N_p pixels - N_p << height × width (± 10-30.000) Extraction can be parallelized **ADDITIONAL OPERATIONS** Building of multi-resolution images Integral image (if HAAR-LIKE is used)	**PIXEL EXTRACTION** $\frac{\text{height} \times \text{width}}{\text{step}^2}$ pixels Extraction can be parallelized **ADDITIONAL OPERATIONS** N vote maps are built Iterative active shape model optimization Integral image needs to be built
PIXEL DESCRIPTOR	RAW, SUB, SURF, HAAR-LIKE GAUSSIAN SUB	HAAR-LIKE
LEARNING MODEL	Extremely randomized trees (classification OR regression)	Random forests (regression) PCA model

	Donner et al. [28]	Ibragimov et al. [43]
TRAINING	**PIXEL EXTRACTION** N+1 datasets (1 phase 1 + N phase 2) - 1 dataset ± 100.000 pixels Extraction can be parallelized **MODEL BUILDING** 1 classification forest (N+1 classes) N double output regression forests	**PIXEL EXTRACTION** N datasets - 1 dataset ± 100.000 pixels Extraction can be parallelized **MODEL BUILDING** N classification forests (2 classes) 1 gaussian model (fron coordinates) **ADDITIONAL OPERATIONS** Integral images need to be built
LANDMARK DETECTION (for 1 image)	**PIXEL EXTRACTION** Phase 1: height × width × delta^2 pixels (1-3M) Phase 2: #Candidates× N×#iterations pixels (max.) (500-1000) Extraction can be parallelized (per phase) **ADDITIONAL OPERATIONS** N probability maps are built (size height×width×delta^2) MRF optimization	**PIXEL EXTRACTION** height×width pixels Extraction can be parallelized **ADDITIONAL OPERATIONS** N probability maps are built (size height×width) GTF optimization Integral image needs to be built
PIXEL DESCRIPTOR	GAUSSIAN SUB	HAAR-LIKE
LEARNING MODEL	Extremely Randomized Trees (classification and regression) Markov Random Field	Random forests (classification) Gaussian model

Table 4.4: Comparison of the time and memory consumption of the algorithms. N is the number of landmarks.

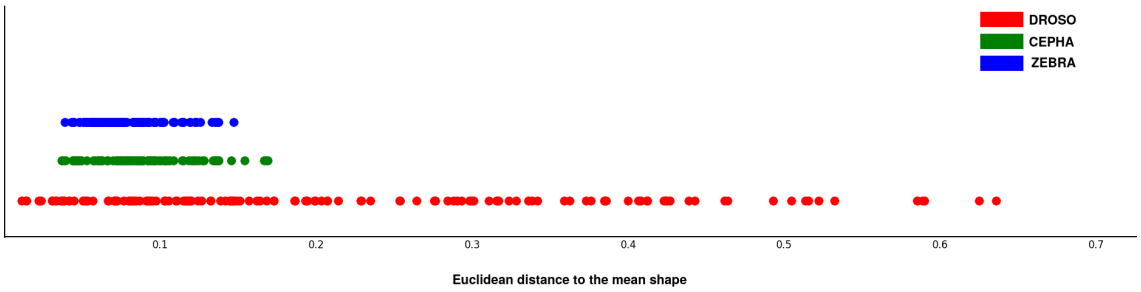


Figure 4.18: Distribution of the deformation in the images

4.4.4 Guidelines

In this section, we provide guidelines to end-users to choose the best method and parameters for their applications.

Method choice

We showed that we could obtain good performances with our algorithm with respect to other landmark detection methods: while offering slightly better performances, our algorithm also creates lighter models and needs to extract less pixels during the prediction phase, thus offering a speed-up during prediction.

However, for datasets with a significantly higher number of landmarks ($> 40 - 50$), we expect the algorithms of DMBL, LC and ILPV to bring better results than our own algorithm because they better exploit the global landmark structure for the localization of the landmarks. In this context, we would advise to use LC's algorithm when landmarks are visually well defined locally, and DMBL otherwise.

Furthermore, our experiments showed that these algorithms could be improved by using some of our algorithm's specificities:

- Different types of pixel descriptors should be considered: our experiments showed that the performances of a given pixel descriptor can vary from one dataset to another given the type of landmarks. Moreover, even if it slightly increases computation time, multi-resolution feature extraction improves the performances. In the context of a new application, we think that the choice of the pixel descriptor should always be empirically assessed, if not per landmark, at least per dataset.

- In most biomedical applications, the deformation between the images will remain small. We showed that this information can be used to reduce the number of pixels to extract at prediction time, and thus speed up the prediction process.

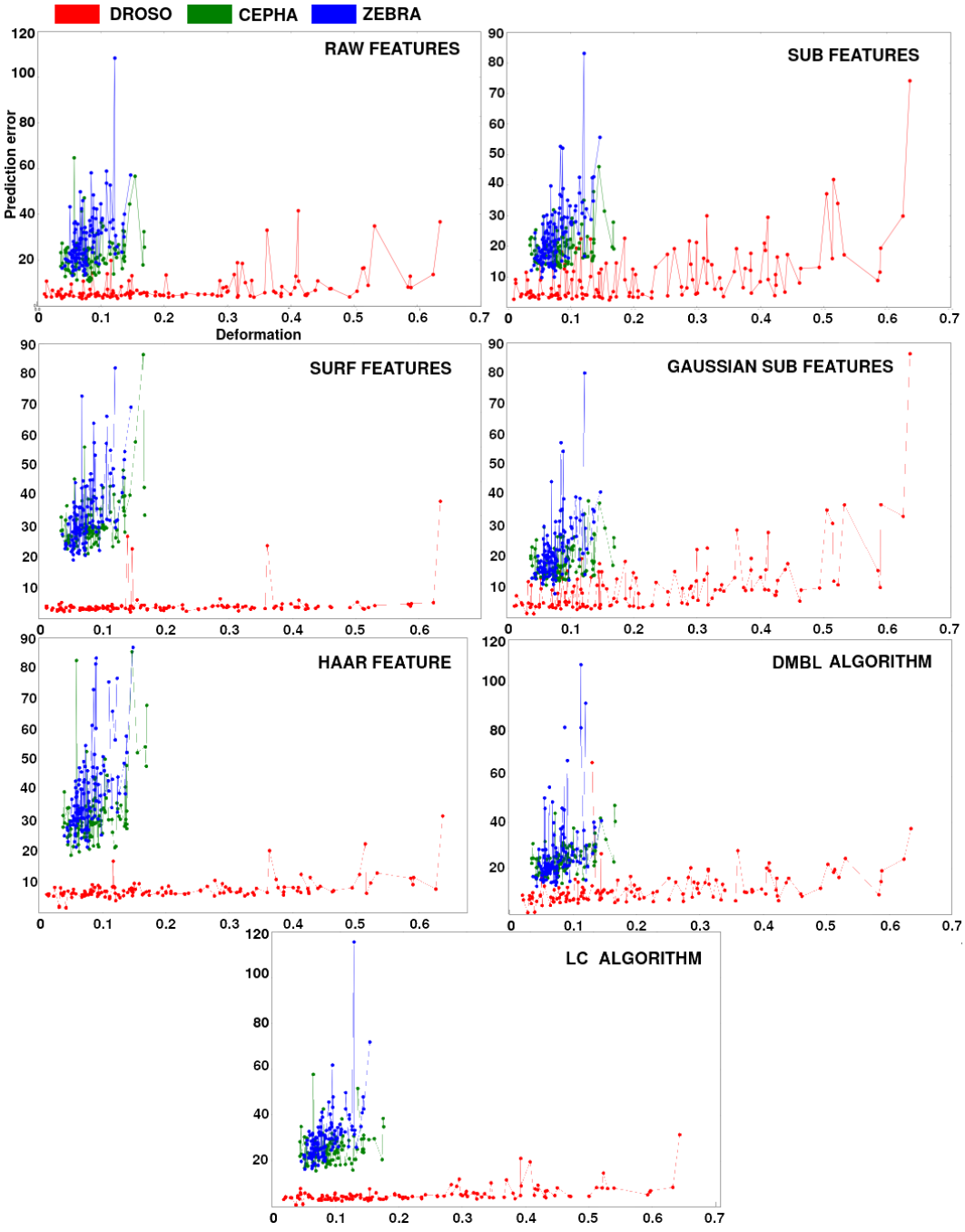


Figure 4.19: Prediction error according to the mean-based deformation criterion.

Parameter setup

Using our method, we recommend to use the classification approach as we showed it has a better accuracy in almost all settings compared to the regression approach. In addition, it is typically less demanding in terms of computing time and memory, mainly because it leads to significantly smaller trees.

According to our experiments, only R, $R_{\max}$, and F have to be tuned to optimize performance. The optimal value of these parameters are expected to be related to the appearance of the landmarks and they might thus be tuned for each landmark individually. On landmarks well defined at borders or intersections, small R values can be used ($R = 6$ for example). Given R and $R_{\max}$ are in pixel units, the range of variation of these parameters should also be adapted to the image resolution : at minimum, R must encompass the landmark possible locations, and $R_{\max}$ the nearest visible structures. We advise to use HAAR-LIKE features in the context of landmarks located at borders or intersections, and GAUSSIAN SUB features otherwise. If low computing time is a key factor, RAW and SUB features could also be assessed. Because it requires more computation time (on a factor 10) than RAW or SUB, we can only advise the use of the SURF descriptor for small datasets or given high parallelization capacities.

We also recommend to adapt the value of N_p to the variability of the position of the landmark in the training images. Indeed, given our sampling scheme at prediction, the more variable the position of the landmark, the more spread in the image will be the pixels tested to compute the final prediction. One should thus increase the number of tested pixels accordingly so as to ensure a fine enough coverage of the area where the landmark could be located. In our case, $N_p = 10.000$ seems a good option to start with.

Other parameters can either be set to some reasonable default value ($W = 8, D = 6$) or be set to their maximum value given the available computational budget (T, P, N_p, W). If computational resources are plentiful, a finer tuning of W and D and N_p can bring some further improvement but according to our experiments this improvement is expected to be small in most cases.

We carried out our experiments on three very different problems, which gives some generality to the previous guidelines. Note however that on the three datasets, the deformations between the images are small, which for example allows to limit the region of interest within images where the landmark can be searched for. To cope with larger deformations, one could play with the value of some of the method parameters, e.g. increasing N_r, and N_p. However, in order to increase the performance and reduce computing times, we recommend either to control image acquisition to avoid large deformations or, if this is not possible, to perform rigid registration between the images before further analysis.

4.5 Conclusion

The results of this chapter showed that it is possible to accurately detect landmarks using a combination of randomized trees and pixel-based multi-resolution features. Given the

small size of our datasets and the variance of the landmarks between the images, we think that these results are very satisfactory. The main advantages of our approach with respect to existing works are its efficiency, its independence to the number of landmarks, and its lower time and memory requirements. All evaluated algorithms are available through the open-source Cytomine platform [66], which provide proofreading tools so that end-users can actually speed-up their annotation processes by focusing on difficult landmarks.

In terms of future works, we think that improving accuracy on our three specific landmark detection tasks mostly require using more data: for some landmarks, ± 100 images does not seem to be enough to grasp the variability of the possible landmark visual representations. Moreover, it seems that some landmarks do not especially correspond to specific anatomical locations, but more to geometric positions or intersections (e.g. landmark 10 on CEPHA dataset. See Figure4.1). This kind of landmark will thus naturally be incorrectly detected by our approach which exploits repeatable visual appearances without using global spatial information. We expect further accuracy improvement might be obtained by taking into account the relative positions and the global structure of the landmarks either directly during the training stage or as a post-processing during the prediction stage. Note that this structure is taken into account by current state of the art algorithms we compared ourselves with. However, given that their accuracy is similar to ours despite this post-processing, more research seems to be needed to find methods more adapted to our datasets. Further improvement might also be brought by adding some parametrization possibilities: for example, the windows could be fine tuned with different shapes at each of the different resolutions and D could be adapted to different ranges of image resolutions. Interestingly, as our datasets and source code are available through Cytomine[66], we hope that other researchers will design new efficient landmark detection algorithms.

Finally, this algorithm can be extended and used as a first step to perform point-based 2D and 3D multimodal registration [86], and also for geometric morphometrics in 3D [3, 14].

Chapter 5

Landmark detection for 3D multimodal registration

In this chapter, we propose a new method for automatic 3D multimodal registration based on anatomical landmark detection. Landmark detectors are learned independently in the two imaging modalities using Extremely Randomized Trees and multi-resolution voxel windows. A least-squares fitting algorithm is then used for rigid registration based on the landmark positions as predicted by these detectors in the two imaging modalities. Experiments are carried out with this method on a dataset of pelvis CT and CBCT scans related to 45 patients. On this dataset, our fully automatic approach yields results very competitive with respect to a manually assisted state-of-the-art rigid registration algorithm. In Section 5.1, we introduce the problematic of CT-CBCT image registration. In Section 5.2, we present our 3D landmark detection method and explain how it is exploited for rigid registration. In Section 5.3, we introduce our dataset of simulation-CT and CBCT and summarize our landmark detection and registration results. Finally, we conclude in Section 5.4.

This chapter is based on the work published in

> *R. Vandaele, F. Lallemand, P. Martinive, A. Gulyban, S. Jodogne, P. Coucke, P. Geurts, and R. Marée.* Automated multimodal volume registration based on supervised 3d anatomical landmark detection. *In Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP) , 2017.*

5.1 Introduction

In radiotherapy, the 3D Computed Tomography Scanner (CT-Scan) is used as the reference for treatment dosimetry and patient positioning. During the treatment itself, a Cone-Beam-CT-Scan (CBCT) is acquired several times at the treatment machine to ensure the proper positioning of the patient with respect to the simulation CT-Scan so as to correctly

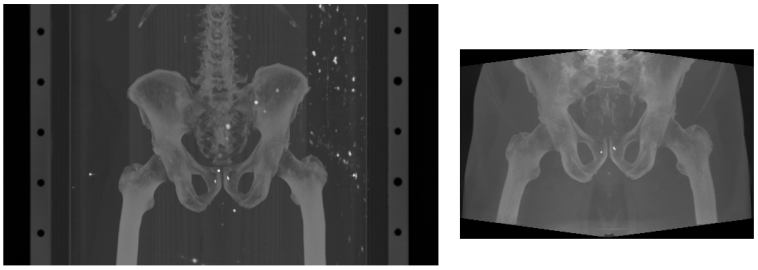


Figure 5.1: Sample volumes (MIP projections) from our dataset. On the left, a CT scan, on the right, a CBCT scan. Notice the differences between the scanned body regions.

deliver the treatment to the tumor. Registration of the two modalities are thus needed in routine applications. Usually, the registration is performed semi-manually by a human operator.

The problem of multimodal rigid volume registration consists in finding the deformation (translations and rotations) that will minimize the difference between the two images or volumes to register. This difference can be evaluated using several possible metrics such as voxel by voxel mutual information or normalized correlation, but also, as we do in this chapter, using the distance between common specific landmarks identified in both volumes. Several general optimization algorithms have been proposed for multimodal rigid registration [98],[71]. However, because the scanned regions can differ between the two volumes to register, these algorithms do not perform well enough without manual intervention for medical registration: an operator is required to manually define in the two images the region of interest (ROI) in which the registration procedure should be applied [42]. For example, as shown in Figure 5.1 for CT-CBCT registration in radiotherapy, CT images will typically correspond to large body scans, while CBCT images will correspond to specific parts of the body (e.g. organs). The application of out-of-the-box registration algorithms such as 3D-Slicer [33] or Elastix [50] on the whole CT and CBCT images will thus fail as it will try to register the full body in CT to a specific organ in CBCT. The ROI for the registration therefore needs to be manually selected in both images, which significantly slows down the registration process.

In this chapter, we propose, and evaluate, a novel fully automated (i.e., free from any manual ROI selection) multimodal rigid volume registration algorithm. The main idea of this approach is to first automatically detect several 3D anatomical landmarks in each image modality, using supervised machine learning techniques, and then to register the two images only on the basis of these landmarks. Our hypothesis is that although patients have different appearances, a specific anatomical landmark is likely to look very similar among different patients in a given imaging modality, hence each landmark appearance could be learned in each modality. We want therefore to evaluate such an approach where anatomical landmarks are detected independently in each modality using supervised learning, then registered, in contrast to commonly used approaches that rely on the design and matching of invariant features across modalities.

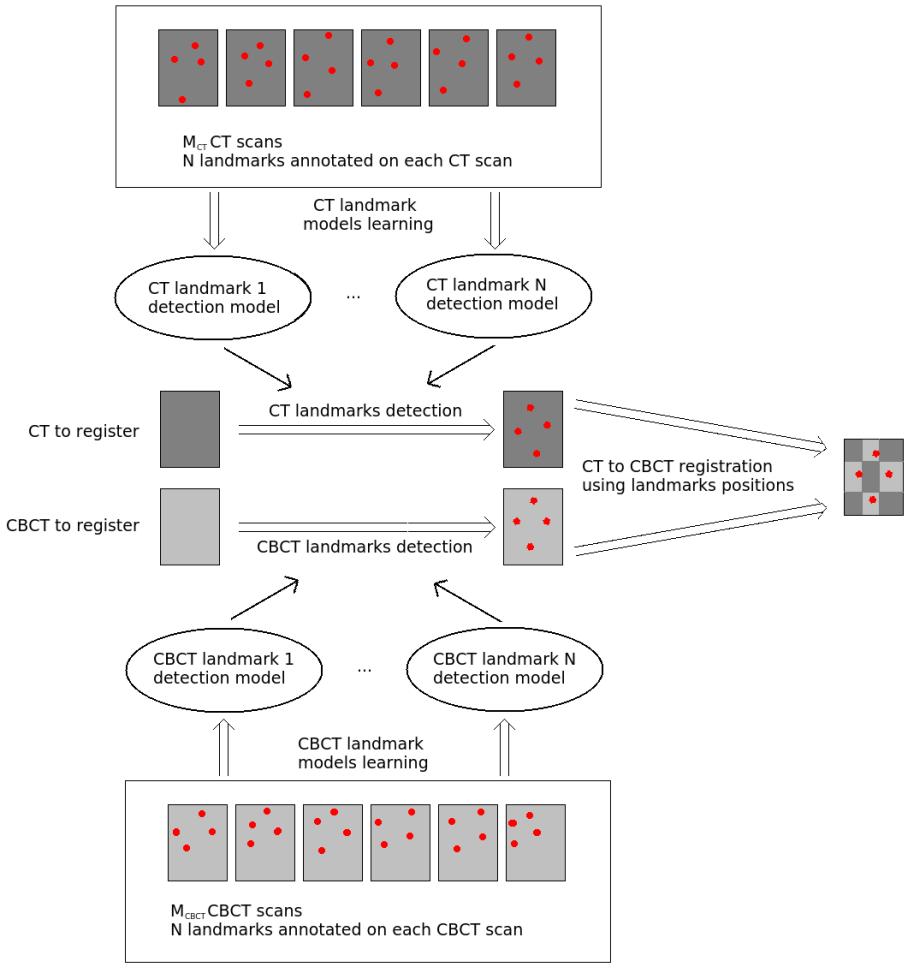


Figure 5.2: Representation of our CT-to-CBCT registration algorithm.

5.2 Method

In our approach, landmark detection models are built for each landmark and each modality independently using training images and expert ground-truth landmark positions. If N landmarks have to be detected, $2N$ detection models will be built (one for each landmark and each modality). For new volumes, once the landmarks are detected automatically in each modality, the registration is then performed through a matching point registration algorithm [5] using all the detected landmark position pairs. A graphical representation of our approach is given in Figure 5.2.

In this section, we first describe the learning approach we used for the landmark detection, and then the registration method we used in order to perform the multimodal volume registration.

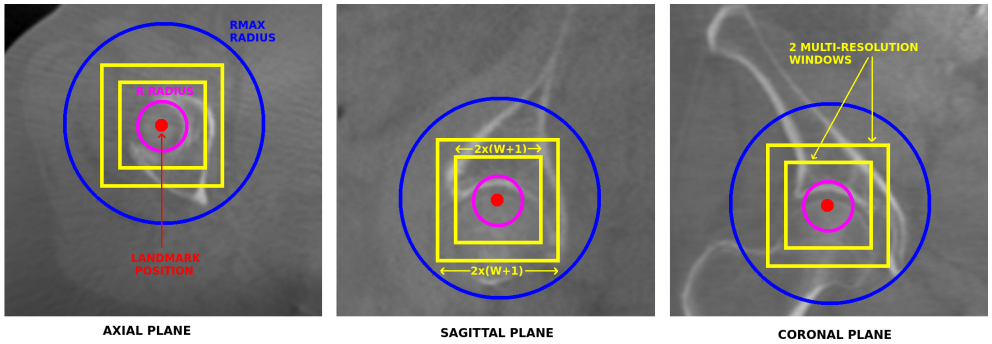


Figure 5.3: Illustration of the different parameters for one landmark in a CBCT scan. The multi-resolution windows describe the landmark voxel.

5.2.1 Supervised 3D Landmark Detection

Local, learning-based, feature detectors are promising approaches for landmark detection in 2D and 3D images. They have been shown recently to outperform global landmark matching algorithms in various applications [31, 91]. Here, we extended the 2D landmark detection method of [80] to 3D imaging.

Our algorithm is based on supervised learning: manually annotated volumes are used to train models (Extremely Randomized Trees [38]) able to predict the landmark positions in new volumes. As in [80], we propose and compare two approaches: in the first, a classification model is trained for each landmark to predict if a voxel corresponds to the landmark position. In the second, a regression model is trained to predict the euclidean distance between a voxel and the landmark position.

Voxel Description. Each voxel v in the training sample is described by D multi-resolution square voxel windows of side size $2W+1$ centered on v on each of the three axes. W and D are method parameters. It means that one voxel is described by $3D((2W+1)^2)$ features. In order to manage possible luminosity variations, the volume voxel values are normalized and the feature values are computed as the difference between each voxel value and the value of the voxel v. Parameters W and D are illustrated in Figure 5.3.

Classification Output. We consider a binary voxel classification model. The voxels can either belong to the landmark class (1) or to the non landmark class (-1). Only one position in each image corresponds to the landmark. If only these positions are considered as landmarks, and if N training images are available, only N positive examples will be available to train our voxel classification model. To extend the set of positive examples, we consider as positive examples all voxels that are at a distance at most R from the landmark, where R is a method parameter illustrated in Figure 5.3. If the landmark is at position (x_l, y_l, z_l) in an image, then the output class of a pixel at position (x, y, z) in the same image will be 1 if $(x - x_l)^2 + (y - y_l)^2 + (z - z_l)^2 \leq R^2$, or -1 otherwise.

Regression Output. With the regression method, the output associated to each voxel is the euclidean distance between this voxel and the landmark position in the training image. More formally, if the landmark is at position (x_l, y_l, z_l) in an image, then the output value of a pixel at position (x, y, z) in the same image will be $\sqrt{(x-x_l)^2 + (y-y_l)^2 + (z-z_l)^2}$

Voxel Sampling Scheme. In both cases, the classification or the regression model is trained from a learning sample composed of all the $\frac{4}{3}\pi R^3$ voxels that are located within a distance $d \leq R$ to the landmark position (the landmark class with the classification approach) and $P\frac{4}{3}\pi R^3$ voxels located at random positions within a distance $R < d \leq R_{\max}$ to the landmark position (the non-landmark class for the classification approach), where P and $R_{\max}$ are user defined parameter. R and $R_{\max}$ are represented in Figure 5.3. In classification, sampling all the pixels inside the R radius allows us to sample more landmark voxels in the positive class than uniform sampling. For the regression approach, this parameter allows us to sample more voxels close to the real landmark position, which helps the model to perform a better differentiation for the voxels close to the landmark position. On the other hand, the effect of the $R_{\max}$ parameter is to artificially reduce the number of distant voxels, which allows to reduce the size of the dataset, while having little to no effect on the prediction accuracy, as we will show in our experiments.

Model Training. The voxel classification or regression model is trained using the Extremely Randomized Trees algorithm [38]. This learning algorithm is a variant of the Random Forest algorithm [12] offering similar accuracy than regular Random Forest while speeding up model training. In this algorithm, an ensemble of T decision or regression trees are built from the original training sample (no bootstrapping), without pruning. At each node, the best split is selected among K features chosen at random, where K is a number between 1 and the total number of features. For each of the K (continuous) selected features, a separation threshold is chosen at random within the range of the feature in the subset of the observations (i.e., voxels) in the node. A score is computed for each pair of feature and threshold, and the best pair according to a score measure is chosen. We chose to use the Gini index reduction score for classification, and the variance reduction score for our regression trees.

Landmark prediction. During the radiotherapy process, the patients are placed in the same position according to the tumor location. When considering specific tumor locations, the landmarks will be found in close areas from one image to another. In consequence, it would be inefficient to search for each landmark in the whole volume. This is why instead of thoroughly scanning the volume, we are considering another solution: in a new volume, we extract N_p voxels taken at random locations following the normal distribution $\mathcal{N}(\bar{\mu}, \Sigma^2)$, where $\bar{\mu}$ is the mean position of the landmark in the training dataset, and Σ the corresponding covariance matrix. The predicted position of the landmark in a new volume will either be the median of the locations of the voxels predicted as landmarks with the highest probability (classification), or as the closest to the landmark position (regression).

Parameter setting. The method depends on several parameters: the radius R and $R_{\max}$, the ration of non-landmark versus landmark voxels P, the number of voxels N_p to extract for computing a prediction, the number of trees T, the size of the window W and the number of resolutions D. These parameters are either set to their maximum value given the available computing resources (T, N_p) or tuned through cross-validation. Trees were fully grown ($n_{\min} = 2$) and the K parameter was set to its default value $\sqrt{3D((2W+1)^2)}$ [38].

5.2.2 Multimodal Landmark-based Rigid Registration

Once anatomical landmark coordinates have been predicted in both images, the registration of the resulting matching pairs of landmark positions is formulated as the least-square optimization problem presented in (5.1).

$$\min_{X,T} \sum_{i=1}^{N} ||p'_i - (Xp_i + T)||^2 \tag{5.1}$$

N is the number of landmarks, p_i and p'_i are the coordinates of the ith landmark in the two images, X is a 3×3 rotation matrix, and T a 3×1 translation vector. To solve this problem, we use the noniterative SVD-based algorithm proposed in [5]. It is important to notice that, as opposed to volume registration based on local feature detectors and invariant descriptors (e.g. [62]), our method does not require matching of landmark descriptors accross modalities.

5.3 Experiments and results

In this section, we first describe our dataset, divided into a training and a test set. Then, we study systematically the influence of the main parameters of our landmark detection method by leave-one-patient-out validation on the training set. Finally, we present registration results on the test set and compare them to a semi-automated volume registration algorithm [33].

5.3.1 Datasets

Our dataset contains images related to 45 patients (male and female) and was acquired at the Radiotherapy and Oncology Department, University of Liège, Belgium. For each of these patients, we have one pelvic CT scan as the reference (45 CTs in total), and at least one corresponding CBCT scan of the pelvis (68 CBCTs in total). We divided this dataset into a training set of 30 patients, each with one CT and at least one CBCT (i.e 53 CBCTs in total), and a test set of 15 patients, each with exactly one CT and one CBCT.

Because our algorithm works better with volumes of identical resolutions and the original resolution information is always available, each CT and each CBCT were resized to $1\times$

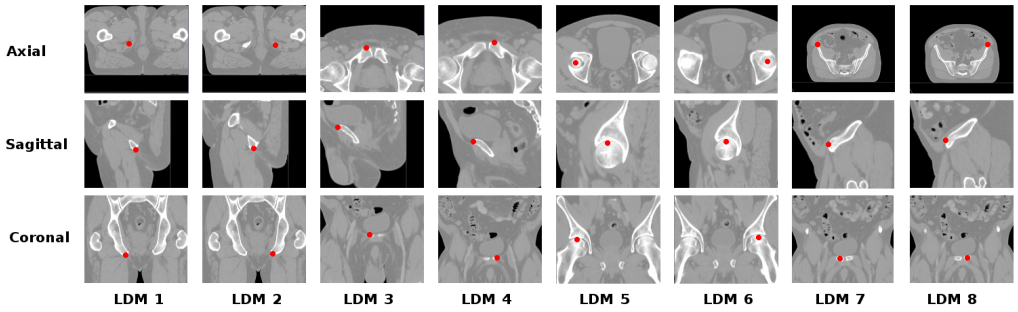


Figure 5.4: Representative pictures showing the position of the 8 landmarks on a CT-scan.

Parameter	Tested values
R	$2, 4, 5, 6, 7, \mathbf{8}, 10, 12, 14, 16$
$R_{\max}$	$10, 25, 40, 50, 75, 100, 200, \mathbf{500}, 1000, 2000$
P	$0.1, 0.25, 0.5, 1, 1.5, \mathbf{2}, 3, 4, 6, 8$
N_p	$1, 10, 100, 1000, 5000, 10000, 50000,$ $\mathbf{100000}, 200000, 500000$
T	$1, 5, 10, 25, \mathbf{50}, 75, 100, 150, 200, 300$
W	$2, 3, 4, 5, 6, 7, \mathbf{8}, 9, 10, 12$
D	$1, 2, 3, \mathbf{4}, 5, 6, 7, 8, 9, 10$

Table 5.1: Sets of values tested during cross-validation for each parameter. In bold, the default value of each parameter used in the first stage of cross-validation.

$1 \times 1mm$ voxel resolution. Originally, CT scan resolutions were comprised between 0.5 and 3mm. The CBCT scans were acquired with an Elekta XVI scanner, that were reconstructed to $1 \times 1 \times 1$mm resolution. More information about the quality of the CBCT image acquisition procedure can be found in [46].

On each CT and each CBCT, 8 landmarks distributed in the pelvis were manually annotated two times by the same skilled operator. The mean distance between the two annotation runs is shown in Table 5.2 (Manual Err.). The position of each landmark is presented in Figure 5.4 for CT scans. We used as ground-truth for each landmark the mean coordinates of the two manual annotations provided by the operator.

5.3.2 Landmark Detection Results

Protocol

For our experiments, we fixed extremely randomized tree parameters to recommended values ($K = \sqrt{3D((2W+1)^2)}, n_{\min} = 2$) [38]. Other parameter values were evaluated in the ranges presented in Table 5.1.

These values were tested for both the regression and the classification approaches using

leave-one-patient-out in the training set. Since it is not possible to explore all parameter combinations, we use a two-stage approach. In the first stage, for each parameter in turn, all its values were tested with the other parameters set to some default value (in bold in Table 5.1). In the second stage, the exact same procedure was applied by using as a new default value for each parameter the value that led to the lowest CV error (in average over all landmarks) in the first stage. The best values for each parameter in this second round were then identified, this time for each landmark separately, and used to retrain a model using all training images. In total, 4480 parameter combinations were tested using computer clusters.

Influence of Method Parameters

The influence of method parameters is shown in Figure 5.5. We did not notice major differences between the classification and the regression approaches. For some particular landmarks, the performance was worse for the CBCT scans. We believe that this difference is mainly due to one particular patient for which our algorithm had difficulties because of its particular CBCT localization: the regions containing the landmarks $3, 4, 7$ and 8 was not acquired. For the classification approach, the R parameter clearly needs to be tuned: too small R will lead to too few positive examples in the dataset, while too large R will associate too distant voxels to the positive class. The regression approach is less sensitive to too large R values. We noticed that small values of $R_{\max}$ (25-40 voxels) work better for both classification and regression. We explain that by the fact that the landmark structure is unique inside the volume, and thus learning to discriminate close voxels is more effective than comparing more distant voxels. Increasing the proportion P improves the performance for classification but smaller P values can be used for regression (which decreases the size of the dataset). Increasing the number of predictions N_p always improves the performance as expected. However, optimal performance is already attained with $N_p = 100000$. The same effect is observed with the number of trees T, with optimal performance reached at $T = 50$. The windows size W controls the number of features and the locality of the information that is provided for each voxel. This parameter clearly needs to be tuned with values in the range 6–8 being optimal in most cases. Increasing the number of resolutions D quickly increases the error, most likely because it leads to overfitting. Small values of $D \simeq 2$–3 are optimal in most cases.

Test Set Errors

Table 5.2 reports for each landmark the error obtained on the test set (column 'Test Err') using the optimal parameter setting determined with the two-stage CV explained above. For comparison, columns 'CV Err' and 'Manual Err' provide respectively the optimal CV error on the training set and the error between the two manual annotations of the human operator.

Results are satisfactory although the difference between the algorithmic and the manual errors remain important. When interpreting these results, we have to take into account

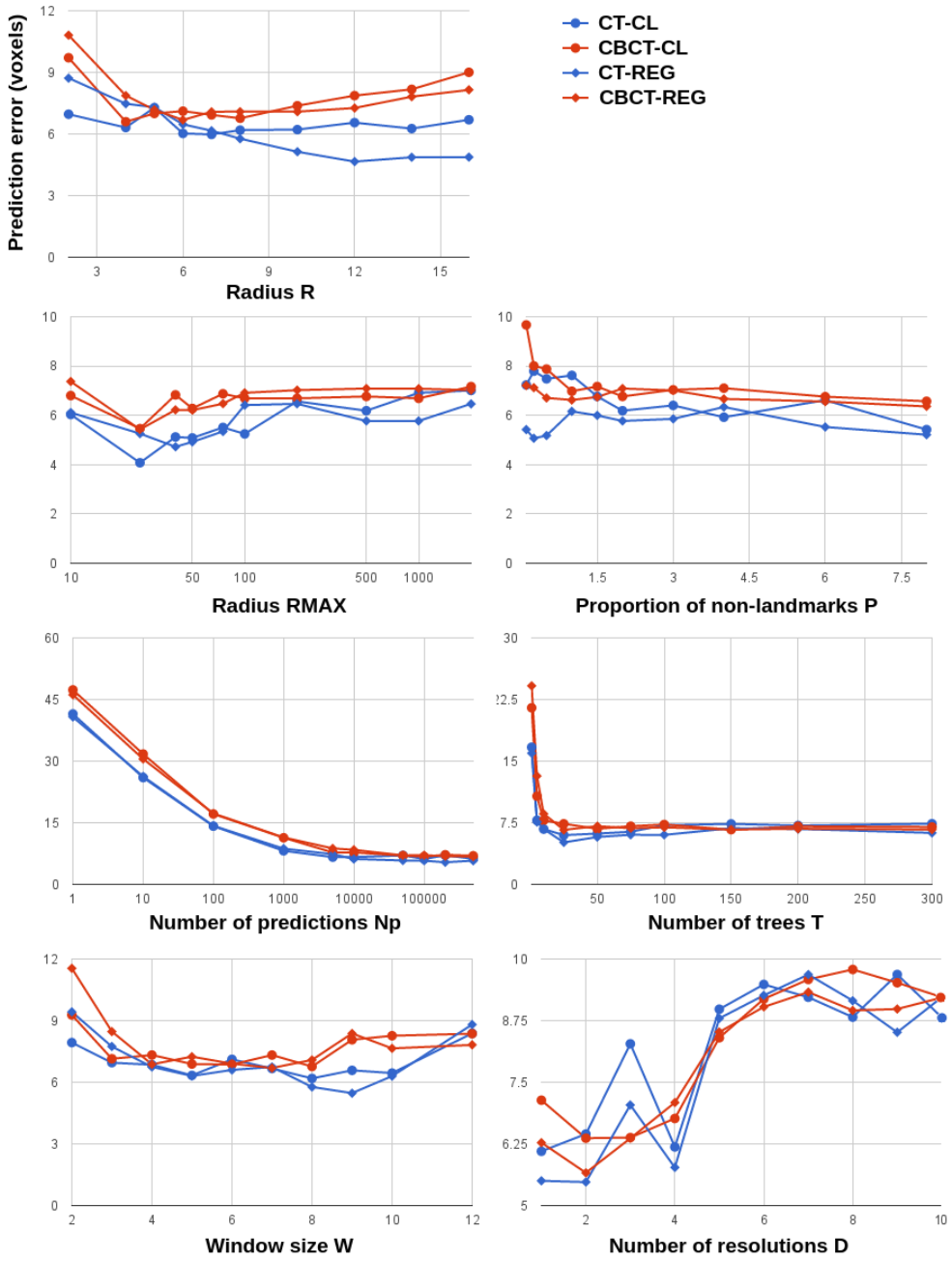


Figure 5.5: Influence of method parameters *From top to bottom, left to right:* $R, R_{\max}, P, N_p, T, W, D$. The $y-$axis is the CV error (in mm) averaged over all landmarks.

.

Landmark	CV Err	Test Err	Manual Err
CT-1	3.23	3.33	1.04
CT-2	2.68	2.77	2.11
CT-3	2.84	2.71	1.81
CT-4	3.43	3.36	2.65
CT-5	2.83	3.28	0.73
CT-6	2.09	3.91	0.84
CT-7	2.92	3.2	0.94
CT-8	2.61	3.7	0.78
Avg	2.83	3.28	1.36
CBCT-1	3.49	3	2.01
CBCT-2	4.53	3.8	2.23
CBCT-3	9.44	4.98	1.34
CBCT-4	5.69	6.39	1.10
CBCT-5	2.84	4.03	1.41
CBCT-6	3.65	3.41	0.98
CBCT-7	8.75	3.56	1.50
CBCT-8	6.08	5.13	1.79
Avg	5.56	4.4	1.54

Table 5.2: Test set results (error in mm).

the low resolution of the CT and CBCT images that forced us to resize our voxels to a $1 \times 1 \times 1$mm resolution. Given this resizing, an error of only 2 or 3 voxels directly translates into an error of 2 or 3mm. With CBCT scans of higher resolution, we could have resized the images to a higher common resolution, which should have led to a lower global error (in mm). Performance on the CBCT scans are worse than on the CT scans. We attribute this difference to poorer image acquisition quality [46].

5.3.3 Multimodal Volume Registration Results

The registration results on all 15 CT-CBCT pairs in the test set are shown in Figure 5.6. The quality of the registration is measured by the average distance between the ground-truth positions of the landmarks in the two images after their registration. **LDM** stands for landmark registration. It corresponds to the proposed approach, i.e., the application of the registration algorithm of [5] after the 8 pairs of landmarks were automatically detected in the CT and CBCT images using our algorithm. **MANUAL** corresponds to the application of the same registration algorithm but using the exact ground-truth positions of the landmarks. Its error is thus a lower bound on the error we can expect to achieve with our method. For comparison, we also provide the error obtained using the state-of-art (**SOA**) semi-automatic registration method implemented in 3D-slicer [33] and described in [44]. We applied the method within the smallest box-sized ROI surrounding all landmark positions and using the Mattes mutual information, which we found to be the best cost metric to use when compared to mean squared error and normalized correlation.

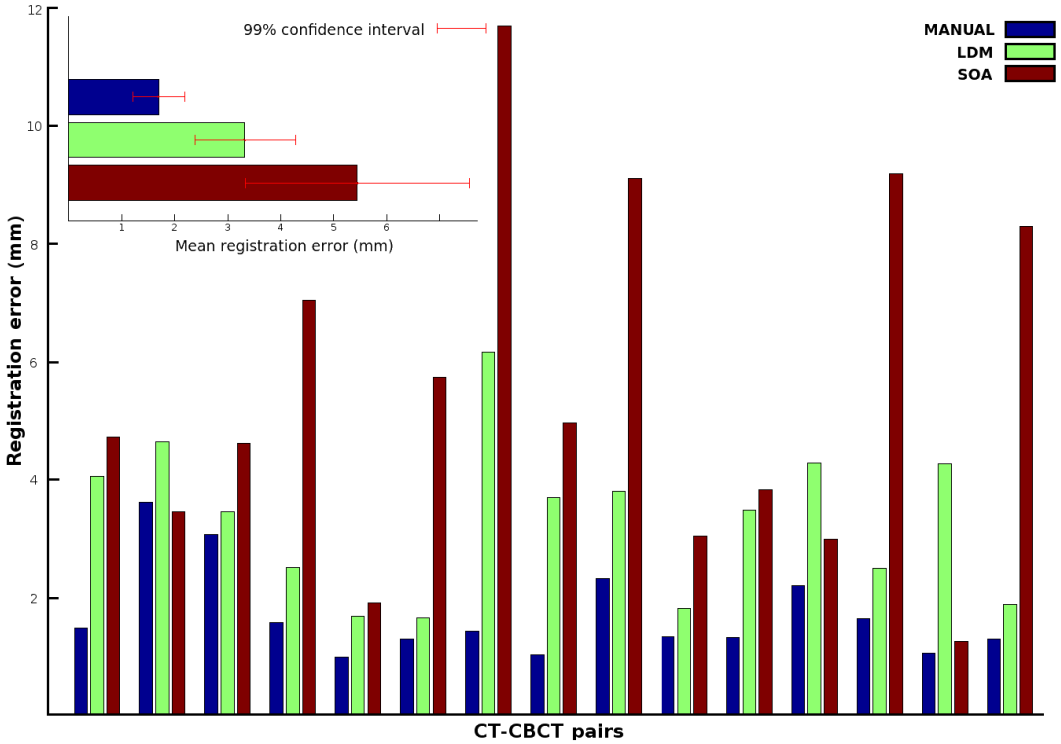


Figure 5.6: CT to CBCT registration results on the test set of 15 CT-CBCT pairs.

As shown in Figure 5.6, the performance of **SOA** is unstable compared to our method and with respect to manual ground truths. For most patients, our algorithm performs much better. Globally, our results are very good. They show that our fully automatic approach performs better than [44] which in addition requires a manual intervention for the delineation of the ROIs. The manual ground truth approach is most of the time much better than our approach, suggesting that improving the quality of the landmark detectors, e.g. by collecting more training images, could potentially improve even further the performance of our method.

On a Ubuntu 15.04 8×2.4Ghz, our paralellized python implementation of our algorithm takes 4 seconds for the complete registration ($T = 50, D = 3, N_p = 100000$). We only take into account the CBCT landmark detection and the volume registration, given that in radiotherapy practice, CT landmarks can be detected offline. On the same computer, the registration of the box-sized ROI of the CT and CBCT took approximately 7 seconds using 3D Slicer, which is also parallelized [44].

5.4 Conclusion

In this work, we proposed a simple but efficient method for fully automated 3D multimodal rigid registration based on automated anatomical landmark detection using supervised machine learning. We applied our approach for pelvis CT-CBCT registration for patient positioning in radiotherapy. Our results showed that our automated approach is competitive with current state-of-art registration algorithms that require manual assistance. Given any kind of body location and modality, interesting landmarks to register can be selected and detected by experts on a small training dataset, and then multi modal registration can be performed on new volumes by using our algorithm. In future works, we would like to manage the possibility to have landmarks out of the volume(s). Future works will also focus on non-rigid registration, where a higher number of landmarks will most probably be required in order to perform plausible registrations. To specifically address this issue, another interesting future research direction would be to design techniques for the automatic selection of the most appropriate landmarks given pre-registered data. Beyond this specific application, we also think that our 3D landmark detection method could be interesting in other areas such as 3D morphometrics [3].

Chapter 6

Advanced analyses of landmark detection methods for morphometric studies

In this chapter, we focus on three distinct and practical aspects of landmark detection for 2D images in the context of morphometric studies. The underlying motivations for these additional analyses are introduced in Section 6.1. In Section 6.2, we study the impact of the number of annotated images on the performances of the algorithm, and propose potential solutions to reduce the number of images to annotate while trying to keep the best performances. In Section 6.3, we carry out an in-depth analysis, on three datasets, of existing post-processing techniques for improving landmark detection methods. We then propose and evaluate in Section 6.4 a new post-processing approach based on the observations drawn from this analysis. In Section 6.5, we carry out preliminary experiments studying the opportunity to improve landmark detection methods by incorporating manual corrections of initial method predictions. Finally, we conclude in Section 6.6.

6.1 Introduction

In the previous chapters, we proposed and extended an anatomical landmark detection method, that we applied in two different contexts: morphometric image analysis [85] and 3D image registration [86]. Although these methods are competitive with the state of the art methods in their specific areas of interest, some interesting aspects of our landmark detection method were not sufficiently analyzed. In this chapter, we will focus on three of these aspects.

The first aspect that we will study, in Section 6.2, is the impact of the number of available annotated images to learn a visual model. Through this study, we would like to estimate how many images are required to obtain (close to) optimal performance, or, from another point of view, how many images should be annotated in order to reach a specific detection

error. We will also check whether the number of images required is landmark dependent and if so, try to characterize visually the most demanding landmarks. While more annotated images is expected to always lead to better performance, determining the minimal number of images required is important in practice, first to reduce as much as possible user annotation efforts (by allowing to focus this effort on the most demanding landmarks) and second to reduce also computing times (as more images means higher model training times and also, in the case of tree ensemble methods, more complex models). At the end of this section, we will also study how the number of images affects the setting of the hyper-parameters of our method. One could indeed expect that some parameters will need to be tuned appropriately to compensate for the lack of images.

We showed in the previous chapters that our method could match or even outperform the performance of several landmark detection algorithms from the literature. Unlike our method, all these methods [60, 28, 43] exploit a sophisticated post-processing step that refines the landmark positions, initially obtained from a visual model, on the basis of a model of the structure formed by the landmarks. In our algorithm [85], we chose not to use such post-processing step and simply compute the median position of the pixel locations with the highest landmark probability. Although our method is competitive as such, an interesting question is whether its performance could be improved further by adding some post-processing. To answer this question, we will first assess in Section 6.3 the combination of our method with post-processing approaches from the literature and then propose and evaluate in Section 6.4 an original post-processing approach based on supervised regression.

So far, the method we proposed is fully automatic, in the sense that it does not require any human intervention at prediction time. In the case where some minimal human intervention would be permitted for the sake of maximizing detection accuracy, it could be interesting to investigate also semi-automated methods. The idea of such methods is to ask a human expert to correct (or set) the positions of a limited amount of landmarks, and then to exploit these corrections to automatically improve the predicted positions of the other landmarks. This could be achieved for example by adapting accordingly the post-processing step. We will describe in Section 6.5 some preliminary work in this direction.

6.2 Analysis of the influence of the number of images

In order to be trained, our landmark detection method based on machine learning needs a dataset of annotated images. We saw in the previous chapters that the image annotation process is a time-consuming and tedious task for a manual operator that can lead to potential annotation mistakes [26, 19, 63]. It is therefore desirable to reduce as much as possible the required annotation effort. Using smaller datasets can also lead to simpler models, and smaller memory consumption. Also, we saw that our method can greatly benefit from parallelization, given that the detection is independent for each landmark: smaller datasets would mean that it could be possible to use more models in parallel, because these models would consume less memory. These observations motivate the study of the influence of the number of images on the performances of the method that we carry

out in this section. We describe the evaluation protocol in 6.2.1, present the main results in Section 6.2.2, and provide practical guidelines in Section 6.2.3.

6.2.1 Evaluation Protocol

The analysis is performed on the three 2D image datasets that were presented in Chapter 4 (see Section 4.2): DROSO (138 images), CEPHA (100 images) and ZEBRA (113 images).

In order to test the influence of the number of images on the performance of the algorithm, we need to build landmark detection models with restricted numbers of images N. These numbers are given in Table 6.1 for each dataset.

Because reducing the number of images means that the size of our training dataset is also reduced (given a set of parameters, we extract a fixed number of observation per training image), we want to test several combination of our method's parameters in order to evaluate if it was not possible to compensate the reduction of the number of images by increasing the number of observations sampled in the training images. Only a subset of our method parameters have an influence on the size of the training sample: R (the *landmark* radius), P (the proportion of not-landmark observations that are extracted), s (the spacing in the landmark extraction grid) and N_r (the number of additional rotated versions of an image added to the training set) with maximal angular rotation α. The values of these parameters that are tested in our experiments are presented in Table 6.1.

The other parameters, which do not influence the size of the training dataset, are fixed to default values identified from the experiments in Chapter 4:

- The R_{max} value is set to 100 for DROSO, 400 for CEPHA and 1000 for ZEBRA.
- The number of pixels extracted at prediction is set to 50.000.
- The depth D is fixed to 5.
- The window descriptor F is RAW for DROSO, SUB for CEPHA and MRDONNER for ZEBRA.
- The number of trees T is fixed to 50.
- The window size W is fixed to 8.

Detection errors in the experiments are estimated using leave-one-out cross-validation: for each image of a dataset, N images are randomly selected among the others in order to train the landmark detection model. This model is then used for detecting the landmark on the test image and computing detection error (the euclidean distance between the predicted landmark position and its ground-truth position).

In what follows, we will report average detection errors as a function of the number of images for different parameter settings. Average errors however do not allow to fully appreciate the behavior of the algorithm over the different landmarks. An algorithm can indeed be very accurate for some landmarks but completely fail for others. To better

DROSO					CEPHA/ZEBRA				
R	P	s	N_r	α	R	P	s	N_r	α
6	2	2	1	0	15	2	2	1	0
9	2	2	1	0	18	2	2	1	0
12	2	2	1	0	21	2	2	1	0
6	3	2	1	0	15	3	2	1	0
6	4	2	1	0	15	4	2	1	0
6	5	2	1	0	15	5	2	1	0
6	2	1	1	0	15	2	1	1	0
6	2	2	2	15	15	2	2	2	15
6	2	2	3	15	15	2	2	3	15
6	2	2	4	15	15	2	2	4	15
N (DROSO)	$5, 15, 25, 35, 45, 55, 65, 75, 85, 95, 105, 115, 125, 135$								
N (CEPHA)	$5, 15, 25, 35, 45, 55, 65, 75, 85, 95$								
N (ZEBRA)	$5, 15, 25, 35, 45, 55, 65, 75, 85, 95, 105$								

Table 6.1: Cross-validation parameters for the analysis of the influence of the number of images on our method.

capture how the algorithm performs over the landmarks for a given setting, we will use a new type of graphical representation, that we call *cumulative error graphs*. Examples of such plot are given in Figures 6.1 C,D and E. The cumulative error graph shows, for a given detection error threshold (horizontal axis, in pixels), the percentage of landmarks that are detected with an error below this threshold (vertical axis, in %). The closer the resulting curve is to the point $(0, 1)$ (resp. $(1, 0)$), the better (resp. the worse) the performance.

6.2.2 Results

Best result for each N

For each given number of images N, Figure 6.1optimal A gives the best mean landmark detection error that was obtained among the different parameter settings that were explored (presented in Table 6.1) for each of the three datasets. Thus, this gives an idea of the global performance increase that can be obtained by annotating additional images on each of these datasets. From this figure, we can observe that the global impact of the number of annotated images is similar on the three datasets. There is a large increase of performance when annotating the first 20-30 images. While annotating additional images still increases the performances, the improvement slows down significantly beyond this threshold. On the DROSO dataset, there is no significant decrease of the detection error after 50 images, while on CEPHA and ZEBRA, the error is still decreasing when reaching the largest number of images, which suggests that these two problems could potentially benefit from more annotated images. This confirms the hypothesis made in Chapter 4, where we noticed that landmarks on the DROSO dataset were easier to detect than landmarks in

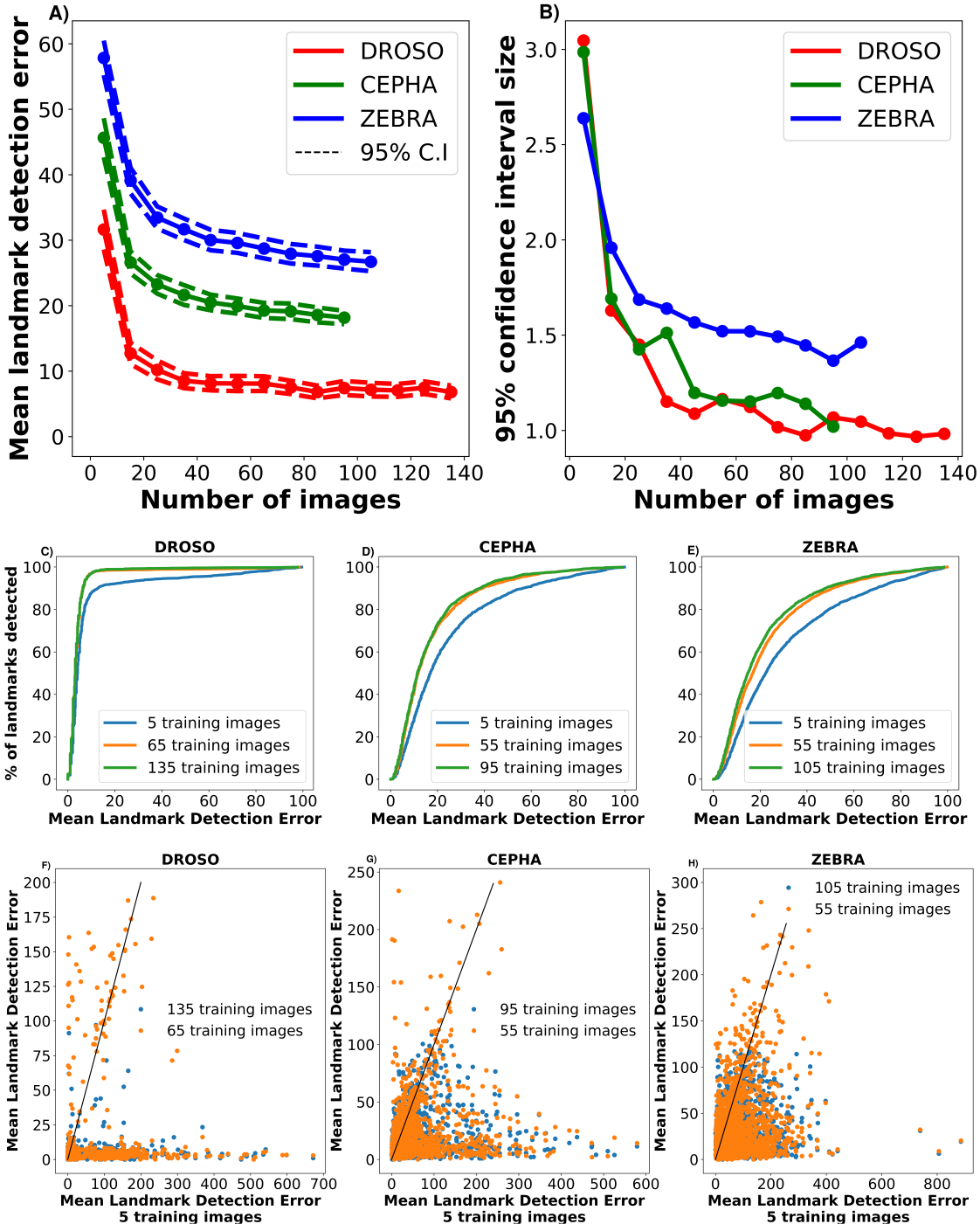


Figure 6.1: A) Influence of the number of images on the mean landmark detection error. B) Corresponding evolution of the confidence interval with the number of images. C),D),E): Corresponding *cumulative error graphs* obtained respectively on DROSO, CEPHA and ZEBRA. F),G),H): Scatter plots showing the evolution of the landmark detection error when using additional images. Each point corresponds to a landmark in a given test image.

CEPHA and ZEBRA. Indeed, these landmarks mainly consist of visual edges and corners, which have less variability in their visual appearance than the landmarks in CEPHA and ZEBRA. In consequence, they need fewer images in order to be correctly identified by our algorithm.

Figure 6.1 B shows that increasing the number of training images also reduces the size of the confidence intervals of the mean landmark detection error. On CEPHA and ZEBRA, this can be partly explained by the fact that the average error decreases (and is always greater than 0). However, on DROSO, the size of the confidence interval continues to slowly decrease when the average error has reached its plateau. This means that the errors (in pixels) made by the algorithm are stabilizing. This stabilization of the predicted landmark positions should translate into a stabilization of distance measurements based on these landmarks, which can be very important in morphometric applications.

Our experiments so far thus show that 40 images are enough in most cases, while additional images can be beneficial to improve further detection error, in problems where landmarks are difficult to detect visually, or, in all cases, to reduce the variability of the predictions. These conclusions are confirmed by the corresponding cumulative graphs that are presented in Figures 6.1 C, D, and E and by the scatter plots in Figures 6.1 F, G, and H. From Figures 6.1 C and F (DROSO), we observe that a small number of images is enough to accurately detect the easiest landmarks of the dataset. However, an important number of landmarks are still very poorly detected with only 5 training images. Using 65 images brings a significant improvement in the detection of those landmarks: the maximum detection error is reduced from about 700 to about 200. Increasing the number of images from 65 to 135 has not much visible effect on the cumulative error plot, but one can nevertheless see on the scatter plots that it decreases even further the detection error of the worse landmarks: the largest errors decrease from 200 to 100. Results are similar on the two other datasets. Indeed, using 55 images instead of 5 decreases the detection error for most of the landmarks. Increasing further the number of images has only a marginal effect on the cumulative error graph but it still improves the detection of the hardest landmarks, as shown in the scatter plots.

Best result for each N, landmark per landmark

For each given number of images, Figure 6.2 shows the smallest average detection error that can be reached for each landmark among the different parameter settings that were explored (given in Table 6.1). For each dataset, we highlighted two landmarks with specific behaviors that support the generic observations we made previously. In Figure 6.3, we give visual examples for each of these selected landmarks. On the DROSO dataset, we can see that landmark 8 benefits from the addition of up to 137 images, while landmark 10 only needs 20 images to be detected with an optimal accuracy. This could be potentially surprising because landmarks 8 and 10 correspond visually both to a corner. However, this can be explained by the observation in Figure 6.3 of larger variations around landmark 8 than around landmark 10. Indeed, tears and blurs seem to be more likely to occur at the position of the wing corresponding to landmark 8 and thus, this landmark will benefit more from additional images. On the CEPHA dataset, we highlighted landmarks 5 and 14. Both

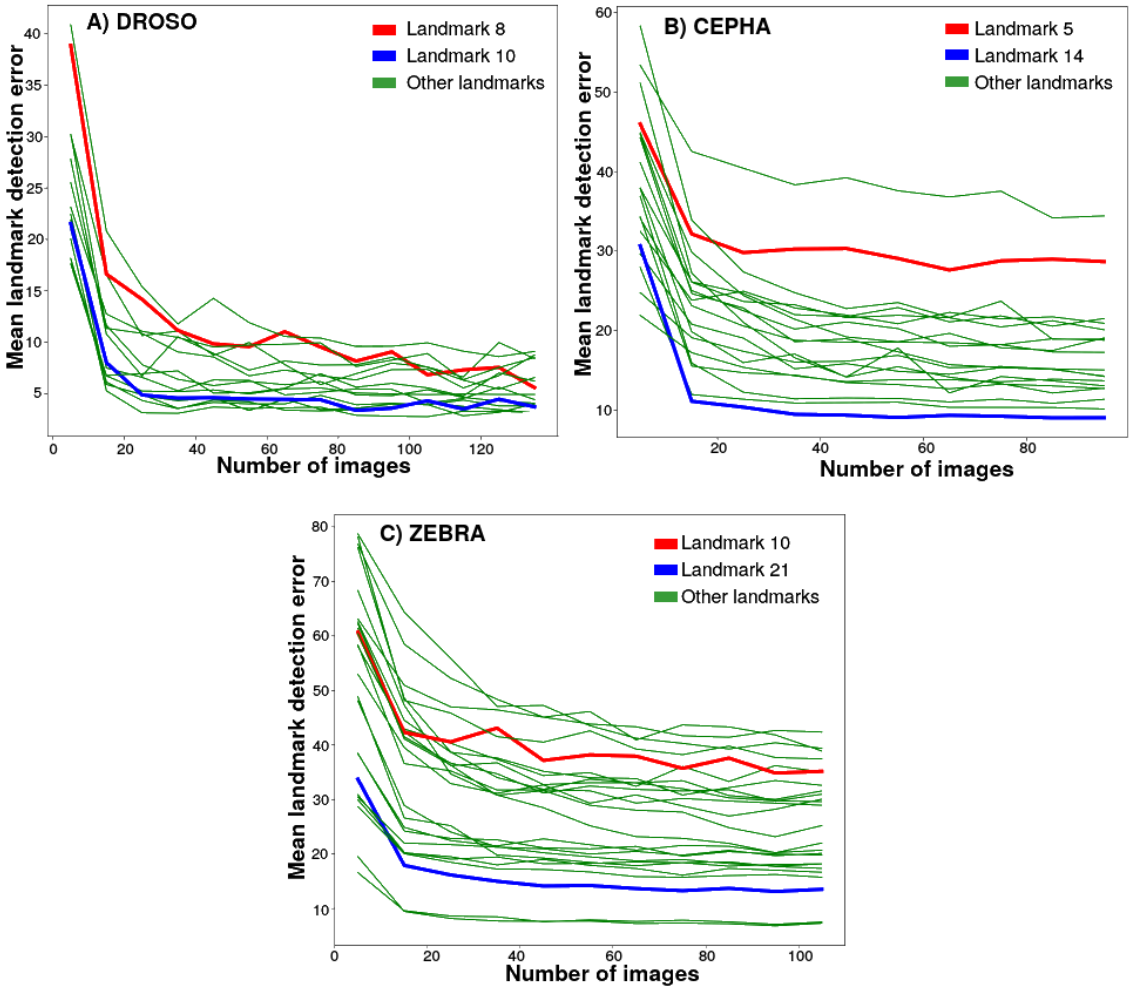


Figure 6.2: Influence of the number of images on the mean landmark detection error (landmark specific) for A) DROSO, B) CEPHA and C) ZEBRA

need less than twenty annotated images to be detected at the highest possible accuracy. However, landmark 5 is detected with a significantly worse accuracy than landmark 14. Figure 6.3 shows that the visual appearance of landmark 5 is varying a lot from one image to another. Annotating new images does not allow to grasp this variation. Landmark 14 is easier to detect visually, because it corresponds to a well-defined edge (the edge of the bottom lip). In consequence, annotating a few images is enough to detect this landmark with high precision. On the ZEBRA dataset, we highlighted landmarks 10 and 21. The detection error of landmark 21 continues decreasing when adding more images, but remains one of the worst errors in the dataset. On the other hand, landmark 10 is one of the landmarks detected with the highest accuracy and it only needs a few images to reach this performance. This is confirmed visually in Figure 6.3. The region around landmark 21 is mostly uniform and contains thus practically no visual information, while the appearance of landmark 10 is more stable.

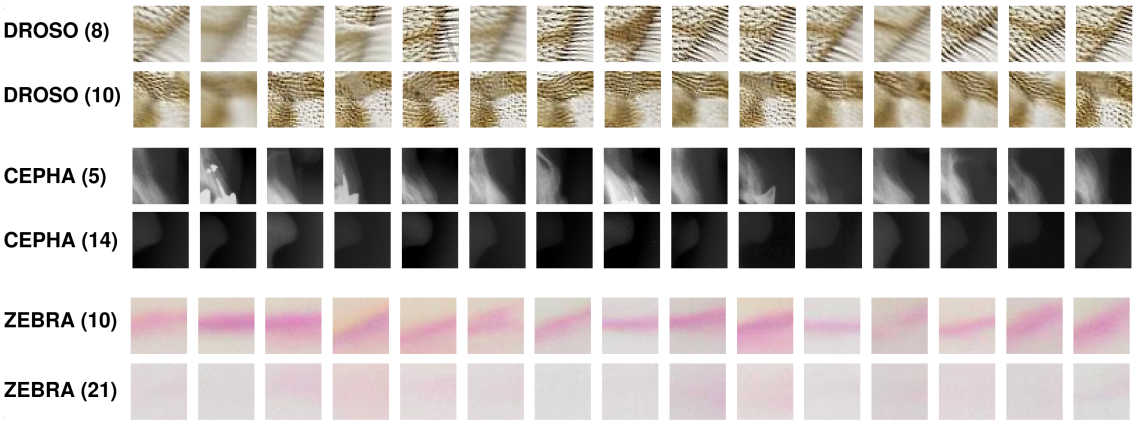


Figure 6.3: 15 random landmark samples for 2 given landmarks of each dataset. The landmark is positioned in the center of the square image.

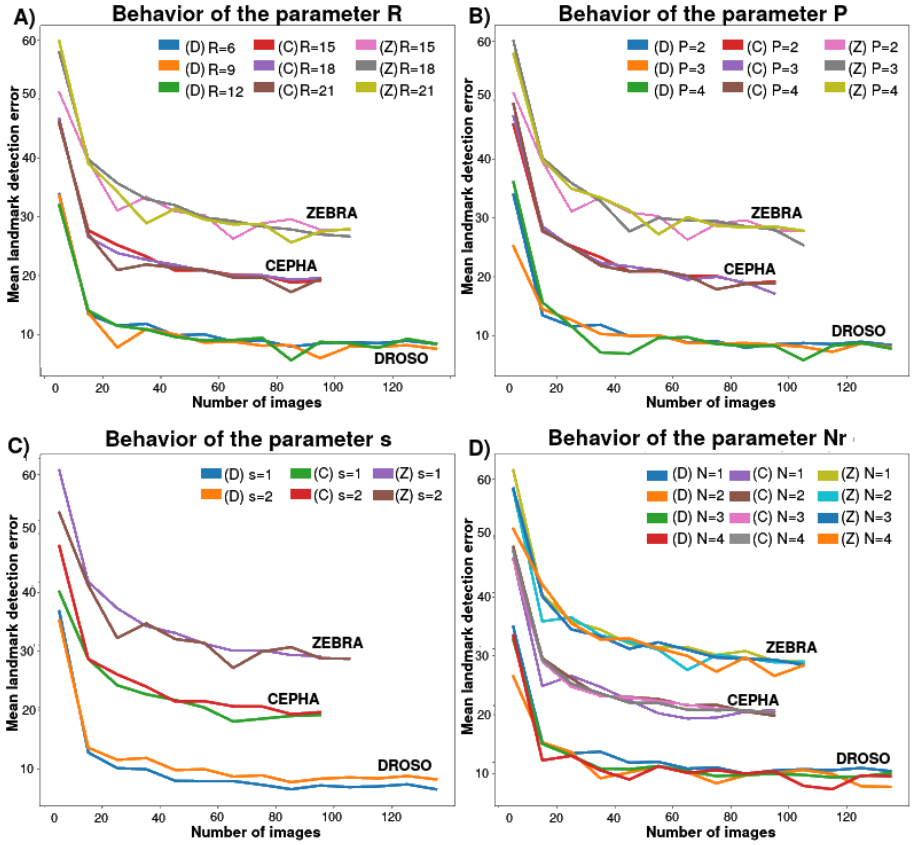


Figure 6.4: Relation between the number of images and the method's parameters.

DROSO						CEPHA						ZEBRA					
N	R	P	s	N_r	α	N	R	P	s	N_r	α	N	R	P	s	N_r	α
5	6	2	2	2	15	5	15	5	2	1	0	5	15	2	2	4	15
65	6	2	1	1	0	45	15	5	2	1	0	65	15	2	2	2	15
135	6	2	2	2	15	95	15	5	2	1	0	105	18	2	2	1	0

Table 6.2: Best set of parameters found for each N.

Influence of the parameters

Figure 6.4 studies the relation between the number of images and the method parameters. For a given dataset and each possible value of a given parameter (among R, P, s, and Nr), we draw the evolution of the best average detection error over all tested values of the other parameters (in Table 6.1) as a function of the number of images. The purpose of this experiment is to see whether the value of these parameters have an influence on the impact of the number of images on detection errors. Whatever the parameter, we hardly see any impact of the parameter value on the detection error curve with N. Similar performance can be obtained for all values in the selected ranges, when the other parameters are tuned to minimize the error. Only a marginal positive effect can be observed when the step size s is set to 1 instead of 2, but even in this case, the effect is essentially independent of the number of images.

This result is confirmed in Table 6.2, where we report the optimal set of parameters, in terms of average detection error, for different values of N. Onthe three datasets, the optimal parameter setting turns out to be clearly independent of the number of training images, except for some small fluctuations most probably only due to randomness. On DROSO and CEPHA, the optimal setting is exactly the same for both the minimum and the maximum values of N.

From these experiments, it thus appears that there is no need for reoptimizing the parameters when considering different number of images. This is obviously an advantage. First, it means that no additional parameter tuning is needed if we want to add new manually annotated images in the dataset. Second, it suggests that parameter tuning can be performed on a subset of the dataset, which can severely reduce computing times when training the models since parameter tuning requires several cross-validation rounds.

6.2.3 Guidelines for manual annotation

From our observations, we can conclude that choosing the exact number of images to annotate in advance, or trying to guess the performance of the method for a given number of annotated images can be tricky. Some generic guidelines can however be drawn from our experiments:

- We suggest to first annotate 20-30 images for every landmark in order to get a first intuition about the algorithm performance. This applies even for the landmarks that seem easy to detect (landmarks located at corners or edges).

- Knowledge of the subject anatomy can help to highlight the landmarks requiring more data: high variance in its appearance, possible close defects (bubbles on the microscope blade, torns, etc.).
- During the annotation process, it is a good idea to choose a dataset of images with large visual variations to capture as many different landmark structures as possible.
- From our observations, annotating more than 70 images is probably not worthy, as this will only slightly improve average detection errors and slightly reduce the largest errors.

When using small datasets, we also saw that it was not necessary to adapt the parameters of the visual algorithm in order to virtually increase the size of the training dataset. Tuning them according to what was advised in Chapter 4, Section 4.4.4, where the parameters are chosen according to the landmarks appearance, is still the best approach.

6.3 Analysis of post-processing methods

In Chapter 4, we proposed a method that mostly only uses the visual appearance of the landmarks to perform the detection. It only exploits landmark positions by sampling candidate landmark positions around the average position of each landmark in the training data. We compared our method with other algorithms that all use an additional post-processing step to refine the landmark positions using information about the global shape formed by these landmarks in the training images. Given that we reached the performance of those methods, we concluded that most of the relevant information was already captured in the distribution of individual landmark positions. However, we never tested this hypothesis and therefore, it is not clear that our method can not benefit from such post-processing steps. Our objective in this section and in the next is precisely to answer this question. In this section, we will evaluate the application of existing post-processing methods on the top of our method, while in Section 6.4, we will propose a new post-processing approach based on supervised regression.

6.3.1 Extension with existing post-processing methods

Experimental Protocol

The existing post-processing methods that are tested correspond to the ones described in Chapter 2 (Sections 2.5.2 and 2.5.1), from LC [60] and DMBL [28]. LC and DMBL apply their post-processing methods on a **vote map** produced by a visual model. A vote map gives, for every position in an image, a score, e.g., a probability, for this position to be the landmark position. There is thus one vote map per landmark. For a test image, our method estimates the landmark probability for N_p pixel positions per landmark. Then, it extracts the landmark position as being the median of the positions with the highest landmark probability. In order to apply the LC and DMBL post-processing methods, we need to represent the results of our visual model as a vote map.

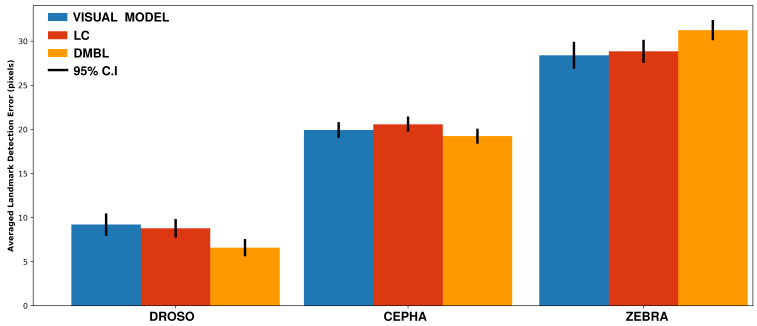


Figure 6.5: Comparison of the best results obtained between Median, LC and DMBL post-processing.

For this, we consider our N_p predictions as the vote map for a given landmark. The N_p position scores are set to the probability value estimated by our tree-based visual model. The other positions that are not scored are set to a landmark probability of 0. We considered first applying a gaussian smoothing on the resulting sparse vote maps, but we noticed that better results were always obtained without any smoothing.

During validation, we optimized the post-processing method parameters using leave-one-out cross-validation on the whole image dataset. We compared these results with the median approach that we used with our initial method. The vote maps were also created using leave-one-out validation, using the corresponding default parameters given in Chapter 4, Table 4.2. Notice that the results of this experiment will be slightly positively biased: the post-processing model trained to predict the landmark positions of a given image using its vote map was trained using the vote maps of the other images of the dataset. However, these other vote maps were built using visual models trained from data coming in part from this given image (because we use leave-one-out cross-validation).

In the rest of this chapter, we will refer to the extraction of the landmark position by using the median position of the pixels with the highest landmark probability (presented in Chapter 4) as the Median post-processing.

Comparison of the post-processing methods

The best cross-validation results obtained during the validation of the LC and DMBL post-processing parameters are summarized and compared to the Median approach (no post-processing) in Figure 6.5. On the DROSO dataset, the DMBL post-processing approach seems to obtain the best results on average. We also notice that DMBL is always better than LC and Median, with the 95% confidence intervals furthermore not overlapping. On the CEPHA dataset, the LC post-processing approach obtains the worst average results, while the DMBL post-processing approach gives slightly better performances than Median. Given the close confidence intervals of these average errors, it remains difficult to consider a clear winner. On the ZEBRA dataset, the DMBL post-processing approach obtains worse results than the two other approaches. For this dataset, we see that LC and Median are

the most interesting approaches to use.

From these observations, we can thus conclude that there is no evidence suggesting to blindly trust the corrections brought by any post-processing method. While it is true that those methods can bring improvements, in some cases, they can either be ineffective or even deteriorate performance.

Evolution of the error and cumulative error graphs

Figure 6.6 shows the evolution of the error when using post-processing methods. In Figures 6.6 A (DROSO), B (CEPHA) and C (ZEBRA), each point represents the evolution of the detection error of a landmark in an image between Median (horizontal axis) and LC or DMBL (vertical axis). The Median and post-processing methods are applied on the same vote maps. The corresponding cumulative error graphs are given in Figures 6.6 C, D and E. Notice that we restricted the range of the cumulative error graphs to a maximal error of 50 pixels in order to get a clear understanding of the method behavior with landmarks that are actually detected by at least one of the methods. The corrections brought by post-processing methods with higher detection errors can still be observed on the plots showing the effect of the corrections (Figures 6.6 A, B and C).

On **DROSO** (Figures 6.6 A and D), we observe that LC worsens the detection of landmarks that are detected with a high accuracy by Median (< 20 pixels), but still corrects some of its biggest detection errors. DMBL makes better corrections, and is not outperformed by Median.

On **CEPHA** (Figures 6.6 B and E), LC also has a negative impact on the best detections made by Median, but slightly improves the error of landmarks detected with low accuracy (error > 40 pixels). The DMBL post-processing approach also makes some interesting corrections for landmarks already detected with a good accuracy by Median, but it has no impact on landmarks detected with higher errors. By looking at the three cumulative error graphs, we can see that this dataset is the least influenced by the use of post-processing methods.

On **ZEBRA** (Figures 6.6 C and F), LC worsens the results for landmarks detected with good accuracy by Median (error < 30 pixels), but brings some improvements for landmark of higher errors. DMBL always worsens the results, but converges with Median and LC on landmarks detected with error thresholds above 40 pixels. If considering higher acceptable error thresholds, the difference between the three methods is thus decreasing.

Additional observations

The main interest of LC is its relative independence from the vote maps generated by the visual model: due to the fact that LC post-processing will primarily focus on the construction of the most likely landmark shape, the landmark detection method is more likely to detect the general area in which a landmark is located. However, for the same reason, the post-processing method can ignore the local maximas generated by high quality

maps, and thus generate small detection errors. This is for example what we observe on DROSO: the method fails to correct the smaller detection errors, but succeeds on the largest ones.

DMBL tends to produce good results with high quality vote maps, but confusions between some of the image areas are still possible. This is for example the case with DROSO: most of the wing intersections (where the landmarks are positioned) are visually close. However, unlike LC, DMBL worsens the detection with vote maps of low accuracy, when compared to Median. Indeed, if a landmark position is not correctly highlighted by a vote map, the post-processing algorithm can not converge to this position, because only the best local maximas are taken into account by the algorithm.

Median works correctly on the three datasets. However, DMBL can easily make the distinction with similar looking visual structures, and LC can correct the largest detection errors that are made by this approach.

6.3.2 Influence of the number of landmarks and images

As we hypothesize that the number of landmarks as well as the number of images have an influence on the quality of the landmark structure model used for the post-processing methods, we want to see in this section how they would behave with smaller numbers of landmarks and images.

Experimental protocol

For the reduction of the number of images, we use the same approach as in Section 6.2: for a given test image, N training images are randomly chosen among the other images of the dataset, with the process repeated for each image and each landmark. The reduced set of images is then used by the post-processing methods to learn their structure model. Note that, since we want to study the impact of the number of images on post-processing only, the visual model is trained in all cases using the full image set, without random subset selection.

To artificially reduce the number of landmarks, they are randomly partitioned into groups of L landmarks and the post-processing method is then run on each of these groups independently, as if the other landmarks did not exist. When the total number of landmarks is not exactly divisible by L, the group with less than L landmarks is filled in with some randomly chosen landmarks. There are thus in this case landmarks that belong to two groups, for which two predictions are obtained, one for each group. For those landmarks, the detection error that is reported below is the average detection errors of the two predicted positions.

Results

Figure 6.7 shows the evolution of the detection error as a function of the number of images and landmarks for the three datasets and the different post-processing methods. Concerning the **DMBL** approach, we see that the number of images (Figure 6.7 A) has a positive impact on the results: adding additional images seems to improve the quality of the landmark structure model. This influence is more important on DROSO than on CEPHA and ZEBRA. It suggests that adding images at post-processing does not make up for a visual model of lesser quality: if the visual model is not able to approximate the real landmark location, none of these two post-processing models will be able to find it. We observe that the number of landmarks does not have a significant impact on the performances of the DMBL algorithm on our datasets. Using only 3 landmarks selected randomly already seem to bring the best results the method can obtain. Moreover, even with a small number of images, the method always obtains results close to Median. This is due to the fact that even if the method is unable to model the landmark structure and outputs random decisions, the DMBL algorithm will still select one candidate among the most likely landmark positions predicted by the visual model.

For the LC post-processing approach, we observe that a small number of images makes the model completely useless, but the detection models seem to improve to solutions of better accuracy until a specific threshold is reached (approximately 30 images). Once this threshold is reached, we see that increasing the number of images further does not significantly improve the accuracy of the post-processing method. Concerning the number of landmarks, we do not observe a clear positive or negative trend. Three randomly chosen landmarks seem to be enough to obtain a good model of the landmark structures.

From these observations, we can thus conclude that using additional landmarks does not bring better accuracy to the post-processing methods, at least when combined with our visual model: for both approaches, the minimum of 3 landmarks randomly selected brings the same results as using the maximum available number of landmarks. Using as many images as possible is, as it could be expected, important, but just like for the visual model, 40 to 50 images seem to be enough for the models to understand the landmark structure.

6.4 SCA: Extension with a new post-processing method

In the previous subsection, we analyzed two distinct approaches allowing to perform the post-processing of vote maps for landmark detection: the LC approach, based on an iterative correction of the landmark positions, and the DMBL approach, based on the selection of the best combination of landmark candidates given a MRF model of the landmark structure. We have seen that even if the DMBL approach gives the best results on average, its results are not always satisfactory, especially when the visual models is of low quality, i.e., when several candidates far from the real positions of the landmarks are selected. In this case, the MRF model is too much biased by the landmark probabilities, although they are of low significance.

In this section, we propose a new post-processing approach that tries to circumvent the limitations of existing methods when applied with our visual model. This approach is also based on the scoring of landmark candidates. The general idea of this approach is to train a supervised regression model to predict the average detection error of a given prediction of the landmark positions and then to use this model to correct the errors made on the candidate positions initially predicted by the visual landmark detection model. The scoring model will be trained on candidate landmark predictions derived by cross-validation from our visual model, with the idea that this training set will capture typical visual errors that will happen at prediction time. We will refer to this method as the SCA algorithm, standing for Shape Correction Algorithm.

We first describe the method in details in Section 6.4.1 and then perform extensive experiments with it on our three datasets in Section 6.4.2.

6.4.1 Method

Before going into a formal description of the method, let us first give an intuitive description of its main steps. The main idea of SCA is to train a *scoring model*, denoted S below, that tries to predict the average detection error for a given candidate *landmark structure*, i.e. a set of (predicted) positions for all the landmarks. This model will be used at prediction time to select the optimal landmark structure, i.e. the one with the lowest predicted detection error, among several candidate structures generated on the basis of the vote maps obtained for each landmark from the visual landmark detection model. To train this scoring model, we will derive its training samples of structures using the same algorithm as used for extracting candidate landmark structures at prediction time. This algorithm generates a given candidate structure by randomly selecting positions among individual sets of candidate positions generated for each landmark. Candidate positions for each landmark are selected that have both a high probability in the vote maps and that are not too close to each other (to maximize coverage). To obtain vote maps at training time that are representative of vote maps at prediction time, these latters will be obtained by leave-one-out cross-validation. More precisely, the vote map for each landmark and each training image will be generated by a visual detection model trained using all images except the one for which the vote map needs to be predicted. Landmark structures given as input to the scoring model will be described by a vector of features composed of distances defined by pairs of landmarks and angles defined by triplets of them.

We describe below successively the algorithm to train the structure scoring model, the actual post-processing algorithm that predicts a landmark structure using this scoring model (page 111), the canditate generation algorithm (page 111), and the feature description of landmark structure (page 112). The training algorithm is illustrated graphically in Figure 6.8, while the post-processing algorithm is illustrated in Figure 6.9.

Training the structure scoring model

Let us denote the training dataset of N_I images as $\{T_1, ..., T_{N_I}\}$ and the corresponding positions of the N_L landmarks as $P(i,l) \in \mathbb{R}^2$ (with $i = 1, \ldots, N_I$ and $l = 1, \ldots, N_L\}$. We want to build a scoring model denoted S able to predict the average detection error of a given landmark structure. In order to do so, we extract a dataset of N_E pairs $\{(d(e), q(e))|e = 1, \ldots, N_E\}$, where each $d(e)$ is a candidate landmark structure and $q(e)$ its corresponding average detection error. The number N_E of candidate structures in the training set is a user-defined parameter. The algorithm also depends on another parameter N_C, which is the number of positions extracted from each landmark vote map from which the candidate structures are generated (see Step 1.(c) below).

Training proceeds as follows (see Figure 6.8 for an illustration of the different steps):

1. For each training image T_i $(i = 1, \ldots, N_I)$:

   (a) N_L visual landmark detection models are built using the $N_I - 1$ other images of the training dataset.

   (b) The visual landmark detection models are applied each on T_i in order to obtain one vote map per landmark denoted $V_{i,l} (l = 1, \ldots, N_L)$.

   (c) N_C candidate landmark positions are extracted from each vote map (candidate extraction is described on page 111). Let us denote by $C(i, l, c) \in \mathbb{R}^2$, the cth position (with $c \in \{1, \ldots, N_C\}$) extracted for image T_i and landmark l.

2. N_E candidate landmark structures $d(e)$ are created using the N_C candidate positions for each landmark and training image and their detection error $q(e)$ is evaluated as follows (with $e \in \{1, \ldots, N_E\}$):

$$d(e) = (C(i_e, 1, k_{e,1}), ..., C(i_e, N_L, k_{e,N_L})), q(e) = \frac{1}{N_L} \sum_{j=1}^{N_L} \|P(i_e, j) - C(i_e, j, k_{e,j})\|^2 \tag{6.1}$$

   where $k_{e,i}$ are random integers $\in \{1, \ldots, N_C\}$ and i_e are random integers $\in \{1, \ldots, N_I\}$. $P(i, j)$ denotes the true position of landmark j in image i.

3. Candidate structures $d(e)$ are transformed into feature vectors $lsd(d(e)) \in \mathbb{R}^p$ of size p using a landmark structure descriptor lsd (see page 112). This yields a learning sample:

$$D' = \{(lsd(d(e)), q(e))|e = 1, \ldots, N_E\} \tag{6.2}$$

4. A regression model $S : \mathbb{R}^p \to \mathbb{R}$ is trained on D' to predict the average detection error from a landmark structure feature description.

In order to learn the regression model, we use the Extremely Randomized Trees algorithm [38] presented in Chapter 2, with N_T trees.

Using the scoring model at post-processing

Now that we have shown how the SCA scoring model S is built, we describe here how it is integrated in a post-processing step to find the optimal landmark structure for a given test image. The principle of our approach is to create N_V candidate landmark structures $d_t(i)$ for the test image. Each structure $d_t(i)$ is then given a score $S(lsd(d_t(i)))$ using the scoring model S. The final landmark structure is then obtained by computing the median position of the $r \times N_V$ vectors having the smallest score (predicted error), with $r \in [0, 1]$ a user-defined parameter. As in the training stage, candidate structures $d_t(i)$ will be generated by first extracting N_C candidate positions for each landmark (using the algorithm described later in this section, on page 111) and then randomly combining positions from the candidates for each landmark.

More formally, for a test image T', a visual landmark detection model, a landmark structure descriptor lsd and a structure scoring model S, the positions $\hat{P}_l$ for each landmark are computed as follows (see Figure 6.9 for an illustration of the different steps):

1. Use the visual model to generate the vote maps V'_l, for $l \in \{1, \ldots, N_L\}$.
2. Extract N_C landmark candidate positions $\{C'(l, c)|c = 1, \ldots, N_C\}$ from each vote maps V'_l.
3. Create N_V candidate landmark structures $d_t(i) = (C'(1, k_{i,1}), \ldots, C'(N_L, k_{i,N_L}))$, for $i = 1, \ldots, N_V$, with $k_{i,l}$ random integers $\in \{1, \ldots, N_C\}$.
4. Associate an error score $s_i = S(lsd(d_t(i)))$ to each structure $d_t(i)$ using the model S.
5. Let X be the set of the $r \times N_V$ indices with the smallest error scores: $X = \{j|s_j < sorted(s_i)[r \times N_V]\}$.
6. $\hat{P}_l = median(\{C'(l, k_{j,l})|j \in X\})$, for $l = 1, \ldots, N_L$.

Candidate landmark position extraction

In order to extract candidate positions from a landmark vote map (in step 1.(c) of the training algorithm and step 2 of the post-processing algorithm), SCA uses the same approach as DMBL. As a reminder, a vote map V_l for landmark l gives the probability that each pixel (x, y) of the image corresponds to landmark l. Let us denote by $V_l(x, y)$ this probability at pixel (x, y). A set D of candidate positions for landmark l are extracted from V_l using the following algorithm:

1. $i = 0$
2. $D = \emptyset$
3. While $i < N_C$:
   (a) $(x', y') = \arg\max_{x',y'}(V_l(x', y'))$
   (b) if $\sqrt{(x' - x)^2 + (y' - y)^2} \geq R_s \forall (x, y) \in D$:

i. $D = D \cup (x', y')$

ii. $i = i + 1$

(c) $V_l(x', y') = -1$

This algorithm depends on two user-defined parameters: the number N_C of candidates to extract from the vote map and the selection radius R_s. The basic idea behind this algorithm is to iteratively select each candidate as the position of maximum probability in the image, while preventing the selection of a candidate position if it is too close (according to the distance threshold R_s) to an already selected position.

Note that the vote maps used in this algorithm are the same vote maps as the one used with the LC and DMBL post-processing in Section 6.3.1. In particular, no smoothing was applied to the vote maps, as it was detrimental with LC and DMBL, and each vote map thus contain only at most N_p non-zero values.

Description of the structure (lsd)

Each candidate landmark structure needs to be described by a feature vector. In our context, we use descriptors directly linked to our morphometric application: each candidate is described by the distances and angles between the different landmark positions. These features have the advantage to be independent of the absolute positions of the landmarks and are thus robust to rotations and translations. With N_L landmarks, there are $\frac{N_L^2 - N_L}{2}$ possible pairwise distances, and, if we consider all three angles defined by triplets of landmarks, we have $3C_3^{N_L}$ angles. For our ZEBRA dataset, this would mean about 300 distances and 6900 angles. To reduce the number of angles, we propose simply to select N_A angles at random per landmark, where N_A is a user-defined parameter. This allows us to avoid a potential combinatorial increase of the number of features with the number of landmarks.

Notice that when the number of landmarks is large, a quadratic increase in the number of features could also be problematic (for example, 124750 features at least can be defined for 500 landmarks). In this case, we would suggest to use other type of shape descriptors, such as the one used in LC [60], based on a PCA approximation of the structure.

6.4.2 Results

In this section, we carry out experiments with the SCA post-processing method. We first describe our experimental protocol. We then perform an analysis of the influence of the main method parameters and study the impact of the number of landmarks and images, as done previously for the LC and DMBL post-processing methods. We compare the SCA post-processing with the other post-processing methods when all are applied on the predictions of our visual model. To get a better understanding on how the SCA approach works, we exploit feature importance scores extracted from the tree-based scoring model to highlight the most relevant distances and angles on each dataset. The SCA approach is

	Description	Tested values
N_C	Number of candidates per landmark and image	$1, 3, 5, 7, \mathbf{10}, 12, 15, 20, 25, 30, 40, 50$
N_E	Total number of training samples extracted	$100, 500, 1000, 2000, 5000, 10000,$ $20000, 30000, 40000, \mathbf{50000}$
R_s	Selection radius at candidate extraction	$1, 3, \mathbf{5}, 7, 10, 12, 15, 17, 20, 25, 30$
N_V	Number of combinations scored at prediction	$50, 100, 500, 1000, 3000, \mathbf{6000}$
N_T	Number of trees of the regression forest	$1, 5, 10, 20, 40, 50, 70, 80, 100, 200,$ $\mathbf{500}, 1000$
N_A	Number of angles per landmark	$0, 3, \mathbf{6}, 12, 18, 24, 30, 36, 42, 48, 64, 80$
r	Ratio of accepted structures at prediction	$0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.15, 0.1,$ $\mathbf{0.05}, 0.03, 0.01$

Table 6.3: Parameter description and values tested at validation. In bold, the default values that were used.

then compared against the full LC and DMBL methods on the same three datasets as well as on some additional Cephalometric challenge data. Finally, to illustrate the behaviour of the algorithm, we conclude the section by giving examples of corrections obtained by the SCA approach on the three datasets.

Experimental Protocol

To test our method, we use a double cross-validation. An external cross-validation is used to estimate the performance of the post-processing method, while the internal cross-validation is used to generate the training set for the scoring model. To reduce computing times, we use 10-fold cross-validation at both stages, instead of leave-one-out as exposed earlier. More precisely, the dataset is divided in 10 folds. For each of the 10 test folds, visual models are trained on the remaining 9 folds and used to produce the vote maps on the images of the test fold. Then, a second 10-fold cross-validation is used on the 9 folds to generate the training set of candidate landmark structures that is used to train the scoring model. The latter is then used to obtain the final landmark structure prediction on the images in the test fold of the external cross-validation.

The parameter of the method are presented in Table 6.3, with the set of values that are tested in the experiments below. To determine the optimal parameter setting, we proceed in two steps. In the first step, we test the parameter values presented in Table 6.3 for each parameter, fixing all the other parameters to their default values (in bold in the table). In the second step, we repeat the same process with the best parameter values found in the first step replacing the default value.

Parameter Analysis

The influence of the parameters of our method is shown in Figure 6.10 on the three datasets. This figure gives the influence of the parameters as considered in the first step of our

optimization procedure (ie. with all parameters fixed to their default values).

For **the number of training samples to extract** N_E, we see that with only a few samples (100), the method is already outperforming the median approach, on the three datasets. However, it takes the method a little bit more data to converge to the optimal performance (around 5000 observations).

For the **number of candidates** N_C, the behavior is different between the datasets: while increasing the number of candidates always seems to be a good idea on ZEBRA, increasing the number of candidates above 30 slightly increases the error on CEPHA, and increasing the number of candidates above 7 significantly increases the error on DROSO. Note that such non monotonic behavior with N_C was expected. Indeed, increasing N_C means that we will include more and more positions with low probability scores among the candidates (the higher R_s, the more it is the case). The better the vote maps, the more we expect the detection error to increase with N_C, as very poor candidate positions will be considered. And the worse the vote maps, the better it can be to explore positions with lower probability scores. Given that DROSO is the dataset with the lowest detection errors when no post-processing is performed, it is not surprising that we observe a fast increase of its error with N_C. Since ZEBRA is the hardest problem, it is also not surprising that its error decreases monotonically with N_C. Note that even on this dataset, we expect that the error will always eventually start increasing if N_C is chosen too high. When N_C is large, candidate positions are spread over very large portions of the images and the visual model becomes useless. In addition, the number of possible combinations of candidates grows very quickly with N_C, making it more difficult for our post-processing model to work properly.

The **selection radius** R_s around the best candidates also seem to show its importance: increasing it on DROSO, our most accurate model, always seems to be a bad idea. This shows that the corrections made on this dataset are often a matter of 1 or 2 pixels. Just like with the number of candidates, increasing the selection radius will bring positions of low probability scores in the vote map as landmark candidates, which can deteriorate the results. On the CEPHA and ZEBRA dataset, increasing it up to 7 pixels seems to improve the results, while going beyond this value is detrimental. As expected, this shows that less accurate models benefit more from candidates that are further apart.

As expected, the **number N_V of combinations** used for selecting the best landmark at prediction time has to be set to its maximum value for all datasets. Testing more candidate landmark structures allows to reach lower detection errors. This parameter should thus be set only taking into account computing times.

This is also true for the **number of trees** N_T in the extremly randomized trees model used for scoring. Detection errors reach convergence at approximately 100 trees for each of our datasets.

The **number of angles per landmark** N_A used to describe the landmark structure does not seem to have any kind of influence on the results on any of the datasets. We thus recommend not to use angles to describe the structures, and just use the distances, mainly to reduce computing times. This behavior could be explained by the fact that distances and angles share a biunivocal relationship in our context: the information brought by the

angles is indeed redundant with the information brought by the distances, since the angles can be retrieved,non-linearly, from the distances.

Concerning the **ratio of accepted structures** r**,** it seems that using low values (0.1 for ZEBRA, 0.05 for CEPHA, 0.01 for DROSO) is the best approach. The difference in optimal values between datasets can be due to the fact that ZEBRA has more landmarks than CEPHA, that has more landmarks than DROSO: with the increase in the number of landmarks, the perfect structure is thus more difficult to find through random candidates combinations, which makes the (median) averaging of the positions of a higher number of top-scoring structures more robust.

Influence of the number of landmarks and images

The influence of the number of landmarks and images is evaluated in Figure 6.11. For the analysis of the number of images, we did not retrain the visual models, and only reduced the number of images used to build the scoring model. As in previous experiments, the training images were selected randomly. The influence of the number of landmarks was studied exactly as in Section 6.3.2, by randomly partitioning the landmarks into groups of a given size at post-processing.

We can observe that the number of images and landmarks have a different influence with SCA than with the DMBL post-processing approach, for which these influences were shown in Figure 6.7. The **number of landmarks** has now clearly a positive influence on the three datasets. Increasing the number of landmarks tends to decrease the detection error. This is expected as, with more landmarks, the landmark structure is better defined. In Section 6.4.2, we will show that the most important distances that are used for building the model are a mix of close and remote relationships focusing on the landmarks that are generally very well or at the opposite, not correctly detected: decreasing the number of landmarks will prevent the model to make this type of selection. Comparing the datasets, it appears that CEPHA benefits less from additional landmarks than DROSO and ZEBRA. This could be explained by differences between the datasets concerning the spatial distribution of the landmarks over the images. In CEPHA, most of the landmarks are indeed grouped spatially into small clusters, while on DROSO and ZEBRA, the landmarks are more uniformly spread over the images. For landmarks that are far away from a given cluster, the information brought by landmarks of this cluster is likely to be redundant. Because of this redundancy, adding further landmarks is less effective on average on CEPHA than on the other two datasets. The influence of the **number of images** is also interesting to analyse. The post-processing method already brings improvement for the smaller number of images. Still, the same amount of training data will be extracted, may they come from 1 or 137 images. This means that with a small number of images, the model is already able to learn to make the distinction between the worse and best candidates.

Comparison with LC and DMBL

The comparison is presented in Figure 6.12 using the same three graphs as in the previous section. For all methods (SCA, LC, and DMBL), these results are the best cross-validation results obtained on the full dataset of images over all parameter settings explored. We observe that our approach obtains significantly better results on all three datasets. On DROSO, we observe that SCA enables better corrections of large errors than the two other methods. The largest difference between SCA and the other methods in terms of cumulative error graphs is observed on CEPHA. The main improvements seem to correspond to landmarks detected with an error in the range $[10, 20]$ pixels. On ZEBRA, the SCA model gives slightly better results on small detection errors, but seems to bring its main benefits for landmarks detected with an error in the interval $[30, 40]$.

Feature importances

In addition to the experiments performed with cross-validation, we wanted to understand how the landmark correction models were built and, in particular, which features the scoring model is exploiting to make its prediction. To do so, we built a SCA score model using all images of the dataset (with vote maps built using leave-one-out cross-validation) and the parameters set to their optimal values as found by cross-validation. Ony landmark distances were used, with each feature thus corresponding to a distance between two specific landmarks. From the resulting Extremely Randomized Forest model, we derived importance scores for all features using the mean decrease of impurity measure [61]. These importance scores represent for each feature the percentage of the output variance that it explains in the model. We arbitrarily choose to represent in Figure 6.13 the distances of highest importances that in total explain 50% of the variance on each dataset (surimposed on one representative image).

On the DROSO wings, 31 ditances among 105 (29.5%) explain 50% of the output variance. We observe that the model focuses on Landmarks 7 and 12, as well as on local relationships in the upper left corner of the wing. Landmark 9, an easy landmark to spot, is not used in the model. Landmarks 7 and 12 are the only landmarks connected to Landmark 8, which is the most difficult landmark to spot on the DROSO images. The distance between Landmarks 8 and 7 is the most important in the dataset. Because of its remote position and its poor visual detection, we believe that the distances between landmark 8 and the other landmarks is unstable. In consequence, the model focuses on selecting good candidates to its closest landmarks, landmarks 7 and 12, that will then help locating the best landmark 8 candidate.

On the CEPHA images, 46 distances among 171 (26.9%) are needed to explain 50% of the output variance. The model focuses on Landmarks 13 and 14, as well as on landmarks 7,8, and 9. This could be explained by the fact that the upper (Landmark 13) and the bottom lip (Landmark 14) are easy to be confused for the visual model, while landmarks 7, 8, and 9 are easy to detect by the visual model. They can thus be used as stable positions that can be be used to evaluate the error made on the localization of the surrounding

landmarks.

On the ZEBRA images, 64 distances among 300 (21.3%) explain 50% of the output variance. We noticed that the model again focused on distances between landmarks easier to detect (e.g landmark 11 and 3), as well as on landmarks that are more difficult to find (e.g landmarks 13 and 2).

On all the datasets, mixes between long and short distances are selected by the model: this probably allows the model to perform corrections on several levels of accuracy: long distances can be used to quickly eliminate outliers, while smaller distances allow the distinction between closer candidates.

Comparison with other landmark detection algorithms

To compare our method against state-of-the-art landmark detection approaches, we propose in this section to update the results that were presented in Chapter 4, Section 4.4.2. First, our method, which includes the visual model and the SCA post-processing, is compared against our implementation of the full LC and DMBL methods on our three datasets. Second, our method is compared against several competitors on two additional datasets released in the context of the 2015 ISBI Cephalometric Challenge. Note that an additional test set was since released for this challenge, and we will thus also compare the results of the algorithms on this dataset.

For the visual model, we reused the best parameters found by cross-validation in Chapter 4, Section 4.4.2. Most parameters of the post-processing models were tuned by leave-one-out cross-validation (LOO-CV) on the training set. Some of them were however set to fixed value: the number of trees N_T (200), the number of training samples N_E (30000) and the number of structures to score at prediction N_v (5000). The number of angles could either be 0 or 5. Other parameters were tested in the set of values in Table 6.3. Unlike in Section 6.4.2, to reduce the computation burden, we perform two independent, instead of two nested, LOO CV for parameter tuning. A first LOO-CV is used to obtain vote maps for each training image and each landmark, with the parameters of the visual model fixed as in Section 4.4.2. Then, a second LOO-CV is used to tune the parameters of the SCA post-processing method. More precisely, for each training image, a scoring model is trained on the other images, using the vote maps predicted from the first LOO-CV, and this scoring model is used to make a prediction of the landmark structure on this training image. The combination of parameters that leads to the best detection error over the second LOO-CV is then used to retrain a scoring model on the full training set, again using the vote maps obtained from the first LOO-CV. With respect to two nested LOO-CV, this procedure introduces some bias, as the scoring models are tested on images that have been exploited to obtain the vote maps from which the scoring models are trained. This bias can make parameter tuning suboptimal and thus be detrimental to our method. We believe however that this bias will be minimal.

The results we obtained on our three datasets using this protocol are shown in Figure 6.14. SCA obtains the best average error on the three datasets when compared to the other

methods: LC, DMBL, and our visual method using a simple median post-processing. The error variance of the SCA method is also lower than that of the other methods on the three datasets.

The results obtained at the 2015 ISBI Cephalometric challenge are presented in Figure 6.15. SCA has only a slight advantage over the other methods on Test1. Our method is only detecting 5 landmarks with the highest performances, while 3 others are better detected with our simple visual model (with median post-processing). On the Test2 dataset, we obtain the second best performance with SCA, while our visual model obtains the worst performances among the other methods. Note that, as explained in Section 4.4.2, other methods use additional, manually labeled, landmarks to improve their performance. Given that we did not exploit such landmarks, we believe that our approach combined with SCA post-processing is even more competitive with the state of the art than our approach without.

Visual examples

To conclude our experiments, we propose to show some examples of typical results obtained by our SCA landmark detection algorithm on biomedical images when compared to the Median approach presented in Chapter 4. For each type of 2D image, we show examples of landmark detection on three images: one detection is performed using Median, and the other is performed using our SCA post-processing in addition to our visual model presented in Chapter 4 (Right). These examples are given in Figures 6.16 (DROSO), 6.17 (CEPHA), and 6.18 (ZEBRA).

Visually speaking, we can see that the improvements brought by SCA are made on landmarks that look more challenging to find: on CEPHA, the landmark 10 (corresponding to a geometric localization depending on other landmarks) tends to be more accurately detected. On ZEBRA, landmarks located at positions with few visual cues also seem to be more correctly detected (landmark 16 for example).

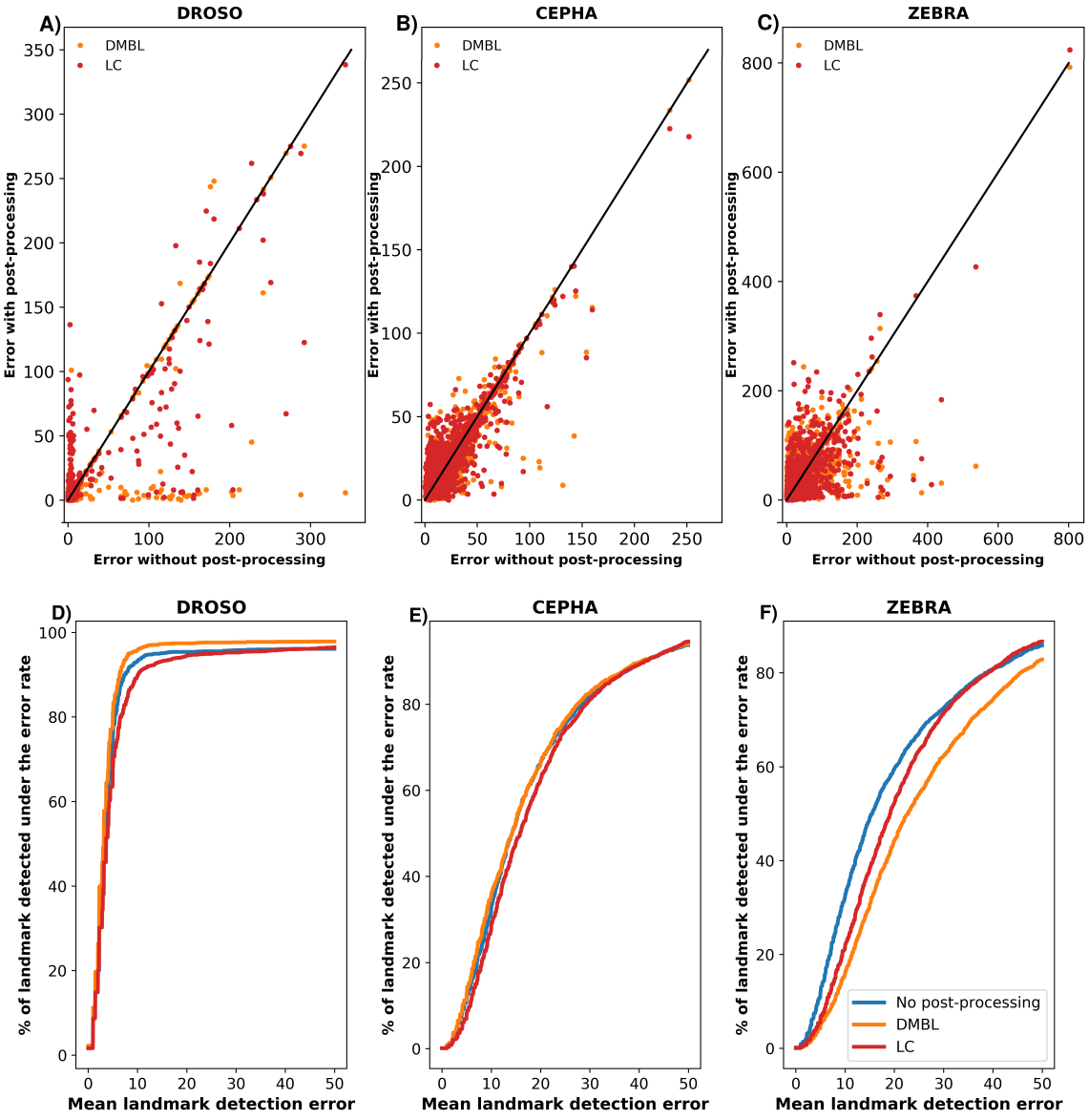


Figure 6.6: Evolution of the error using post-processing methods.

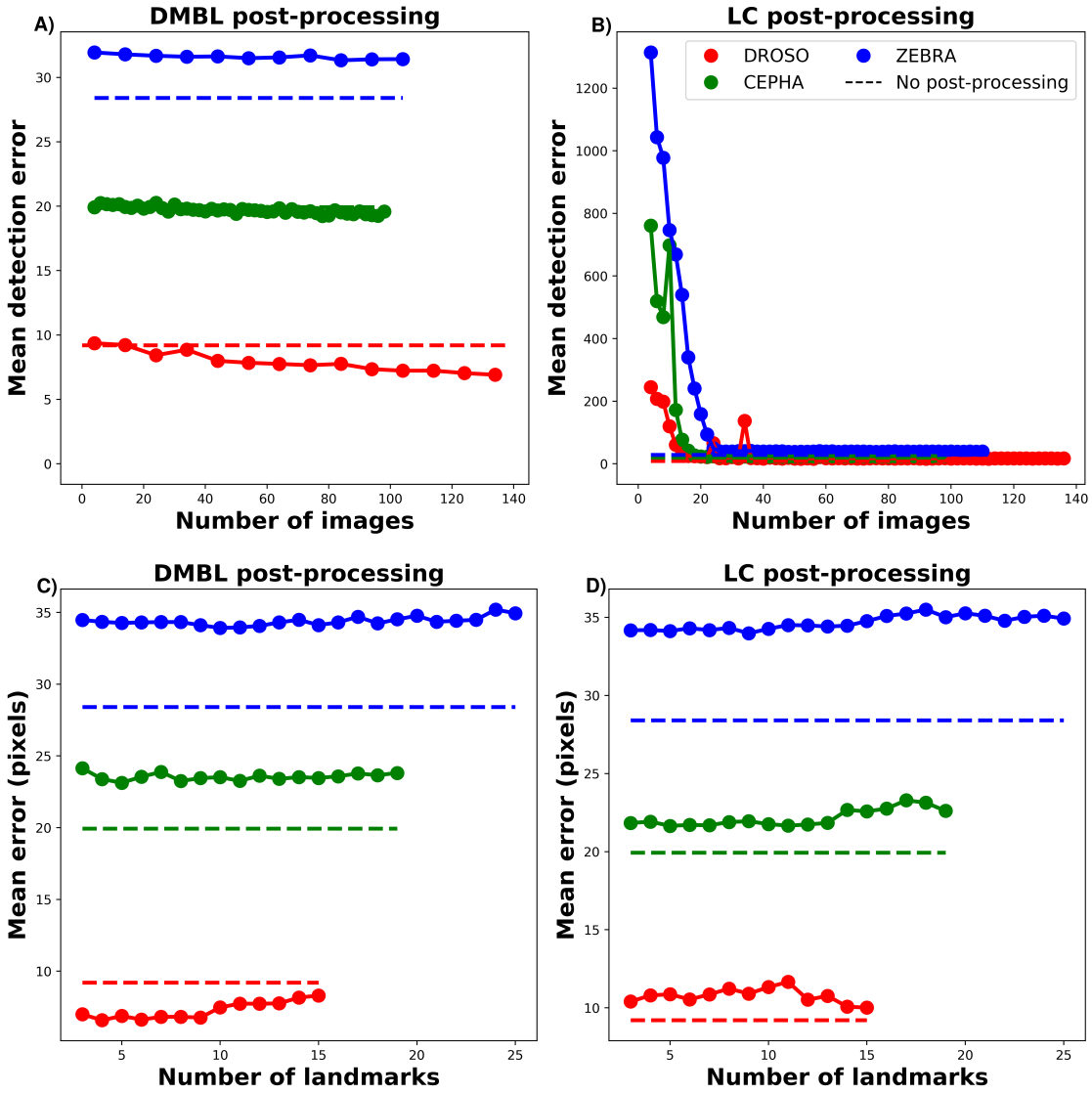


Figure 6.7: Influence of the number of images and landmarks on the post-processing methods.

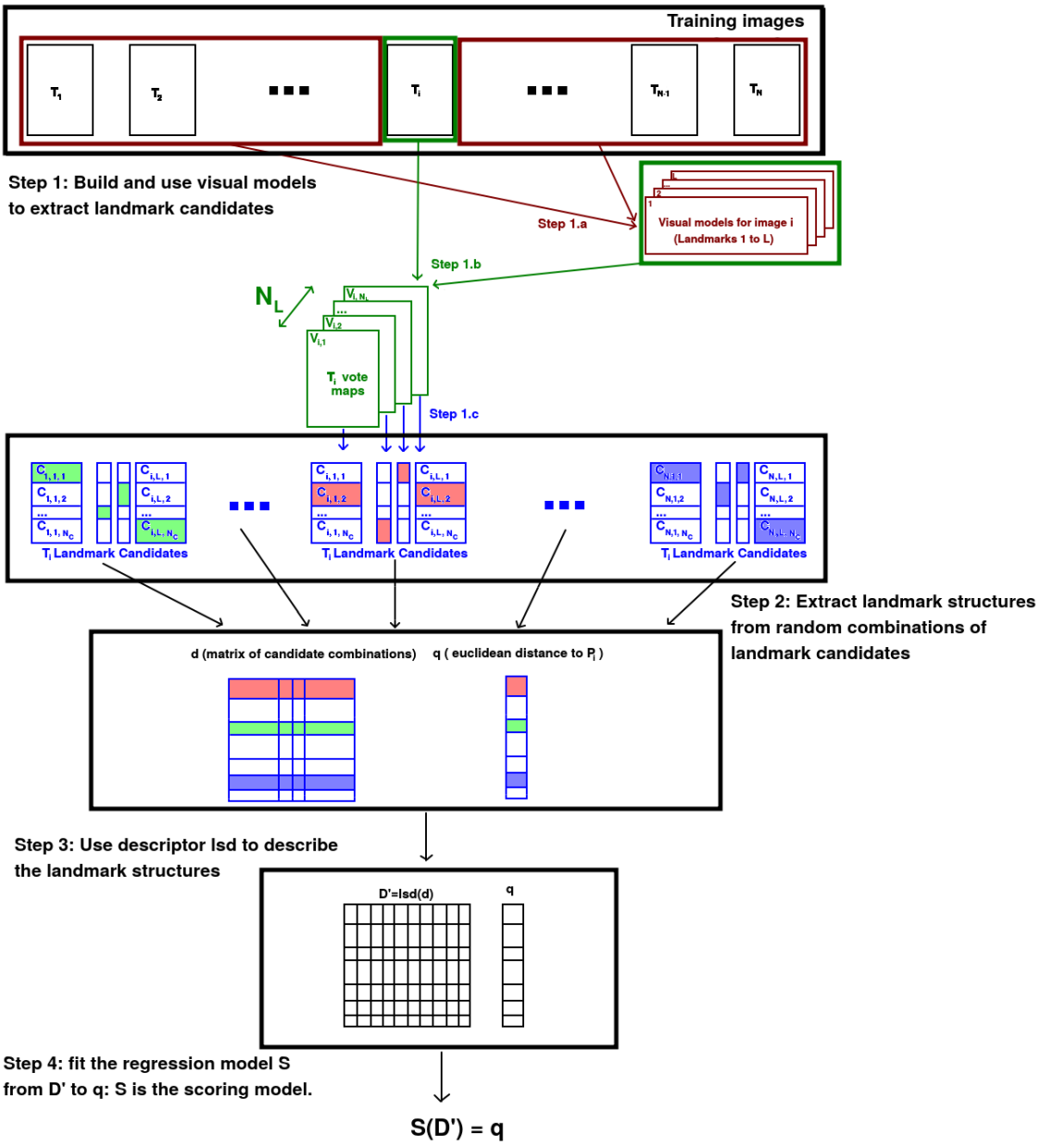


Figure 6.8: Summary of the scoring model building for SCA post-processing.

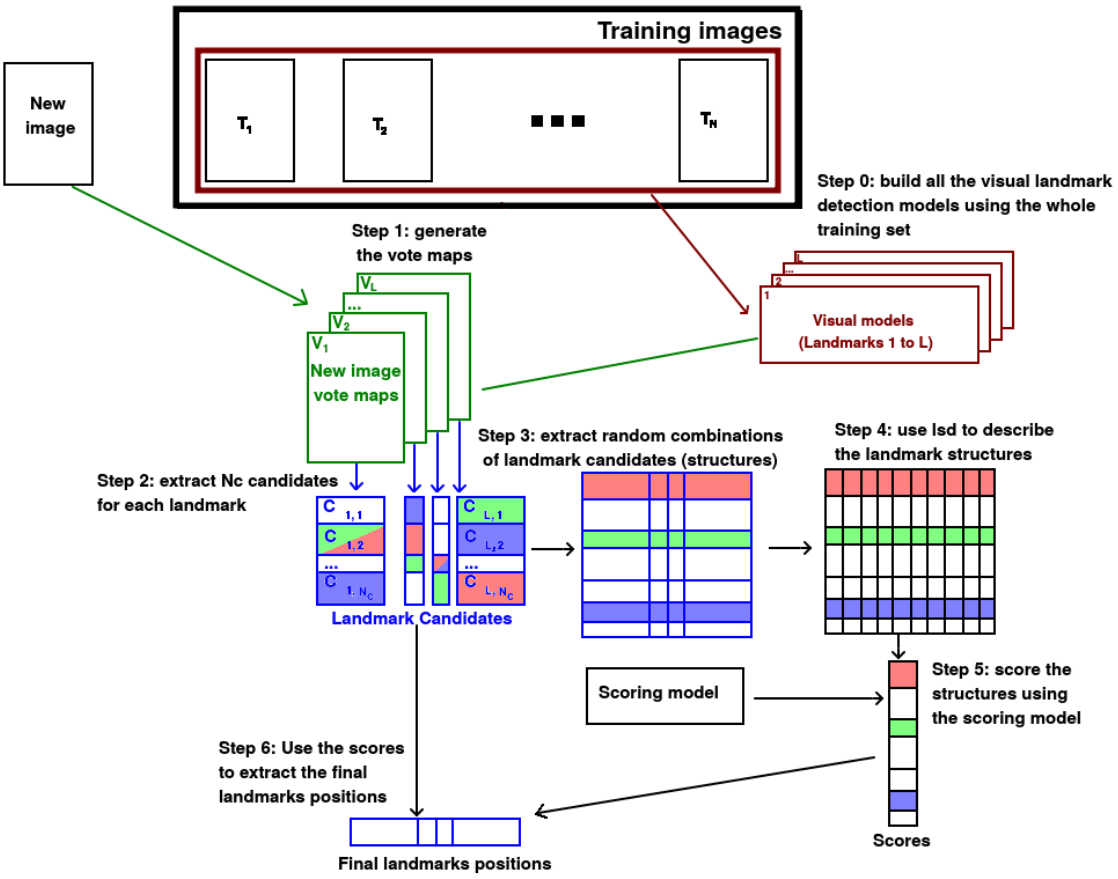


Figure 6.9: Summary of the use of SCA post-processing at prediction.

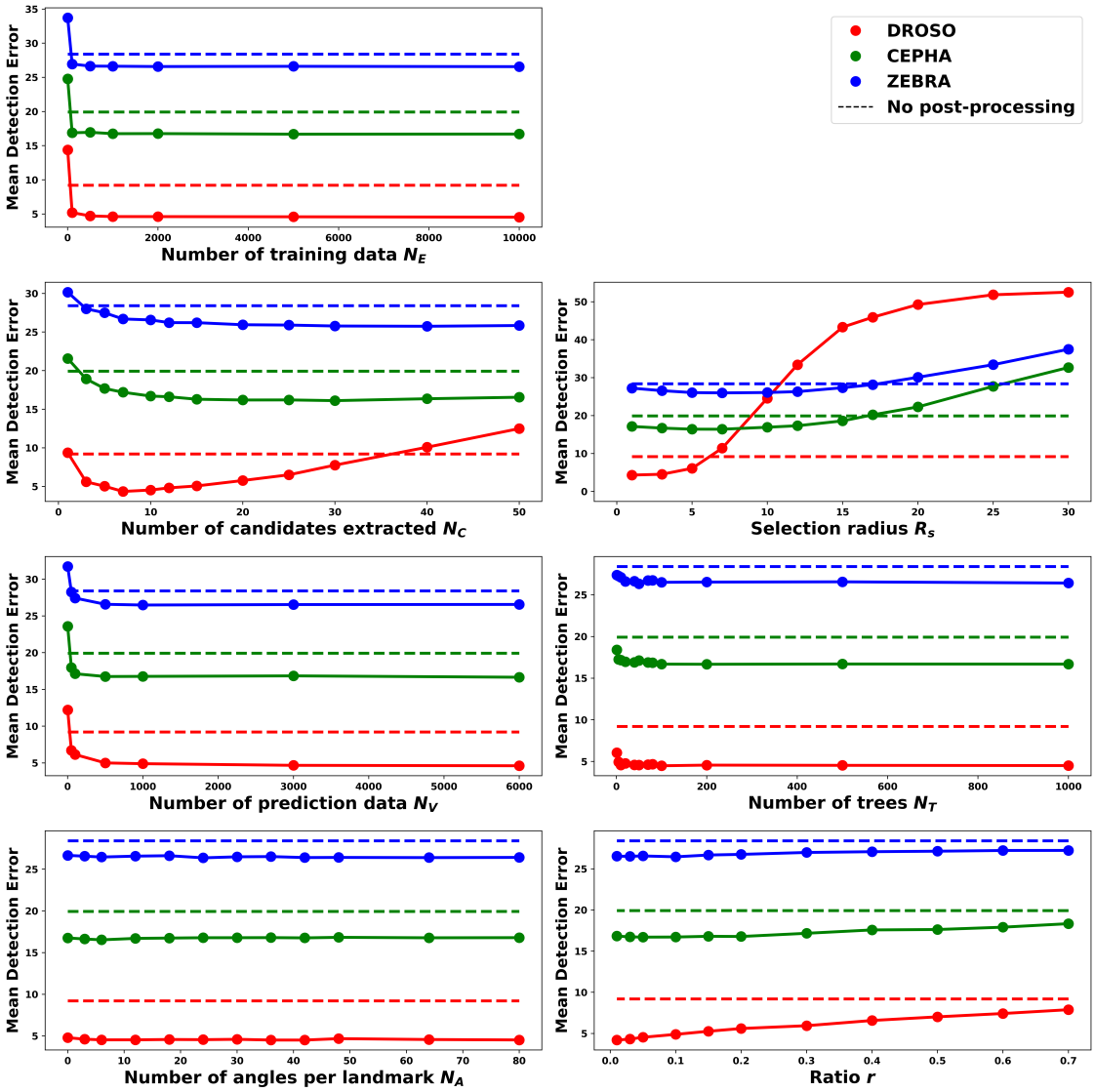


Figure 6.10: Influence of the parameters of our post-processing method.

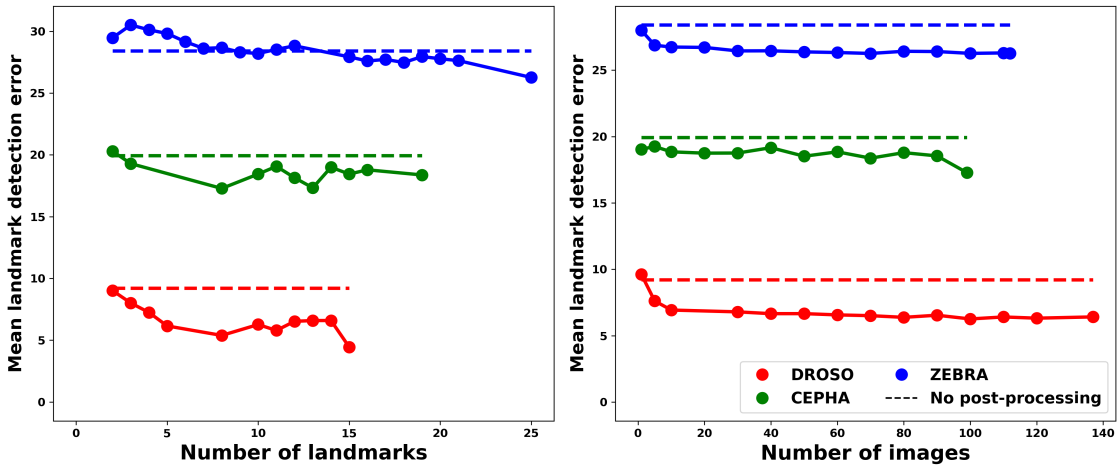


Figure 6.11: Influence of the number of images and landmarks on our post-processing method.

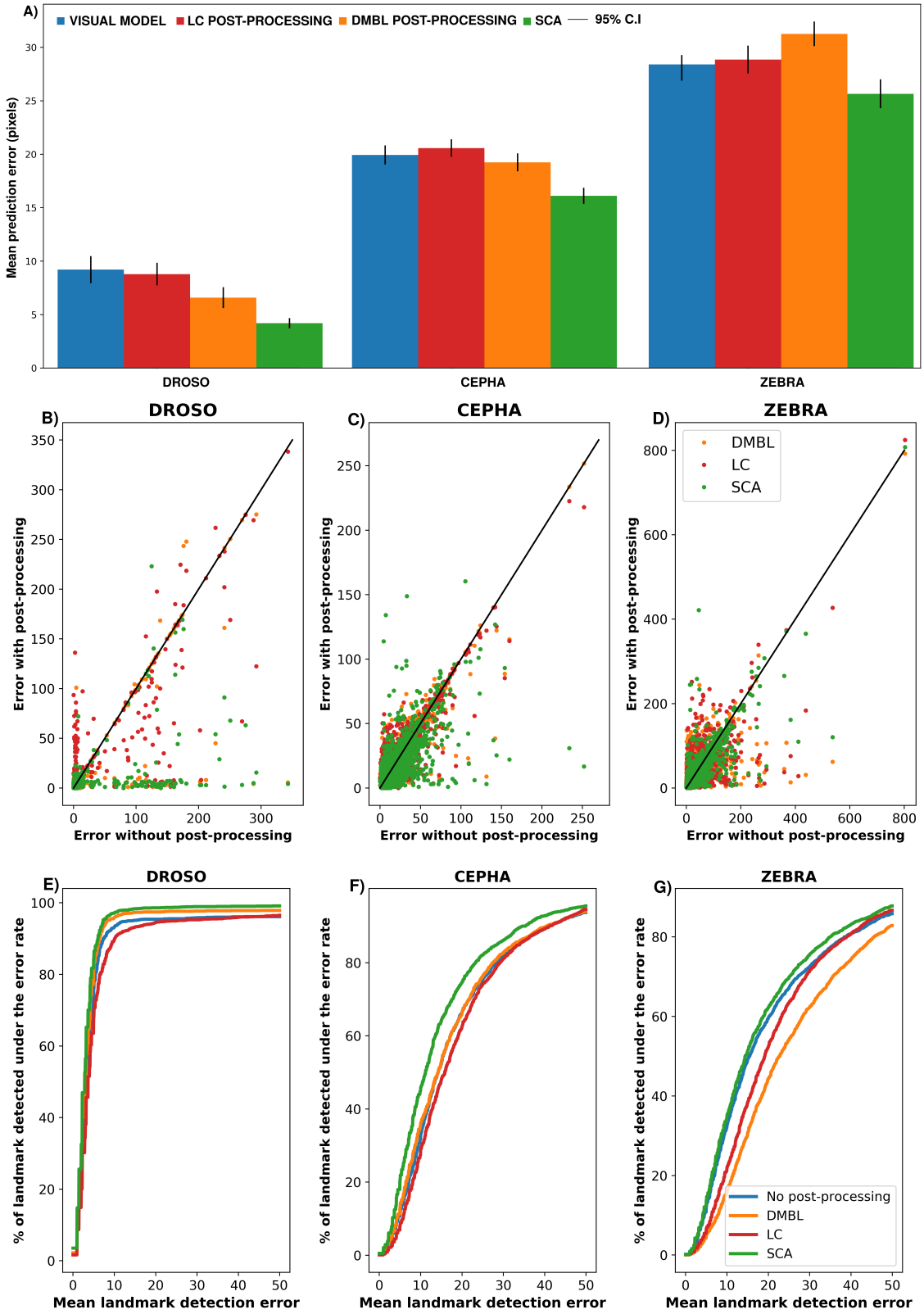


Figure 6.12: Evolution of the error using post-processing methods.

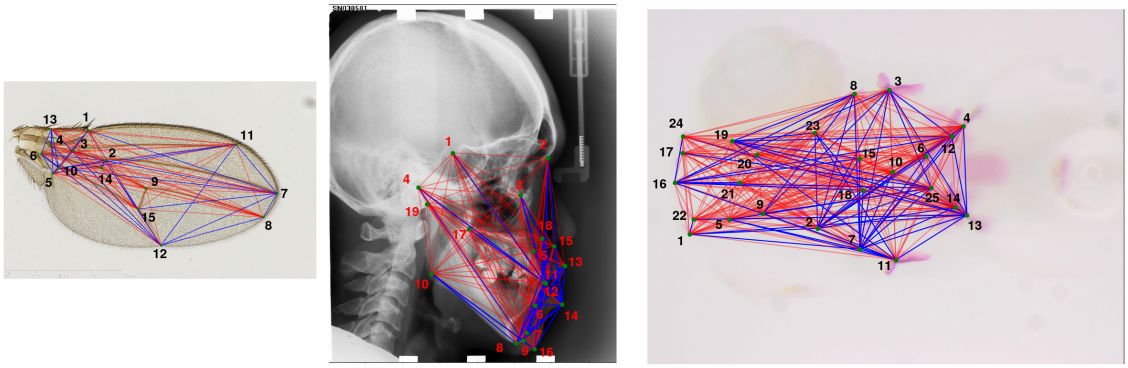


Figure 6.13: Most important features used to build the SCA model (in blue).

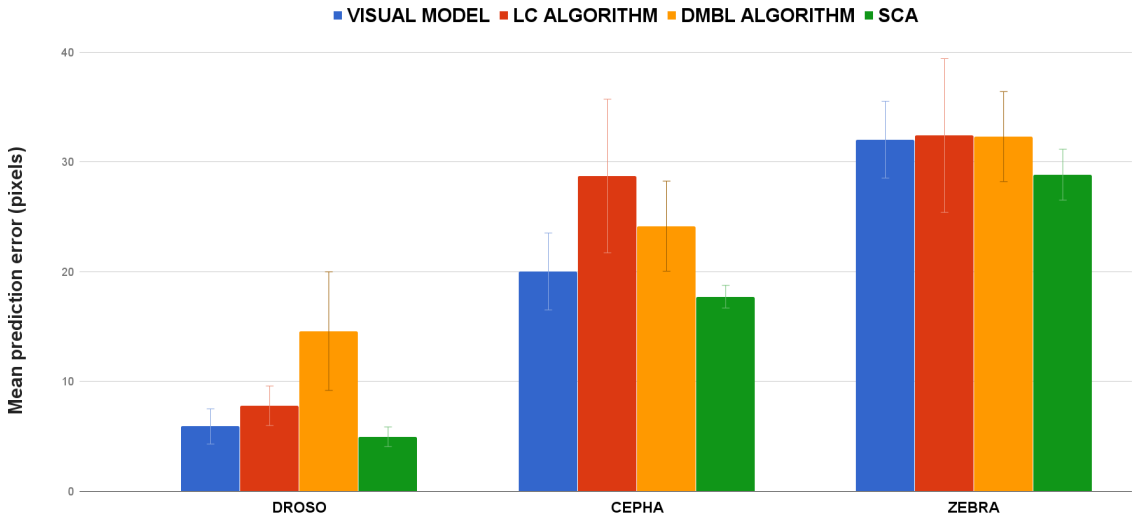


Figure 6.14: Comparison of SCA with other results obtained on the tests sets.

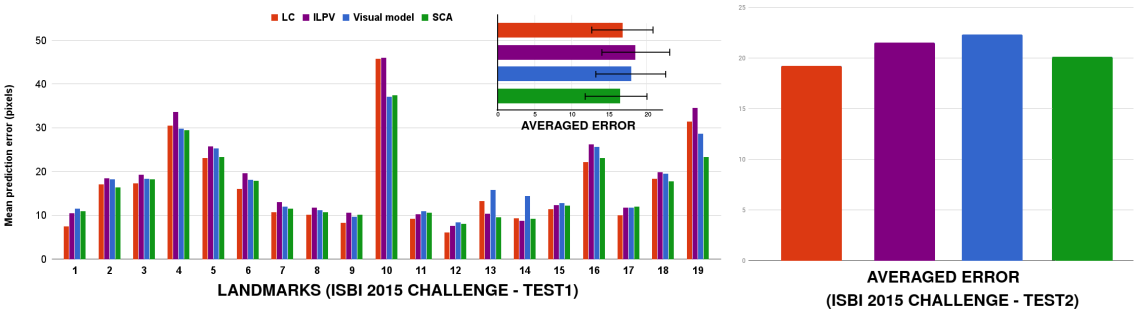


Figure 6.15: Comparison of SCA with other results obtained during the 2015 ISBI Cephalometric challenge.

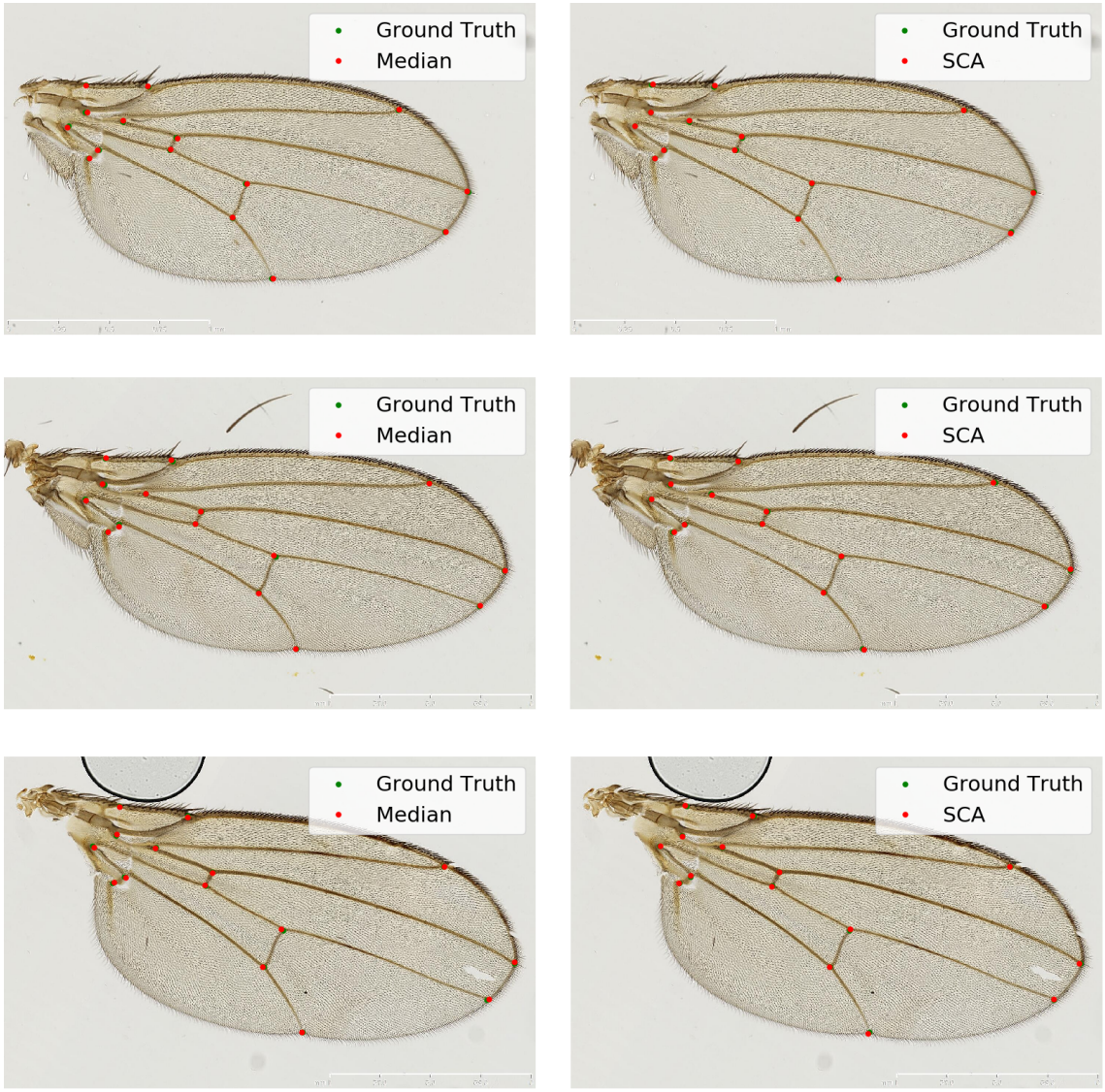


Figure 6.16: Sample of landmark detection on three DROSO images. On the left, using the visual model with median. On the right, using SCA.

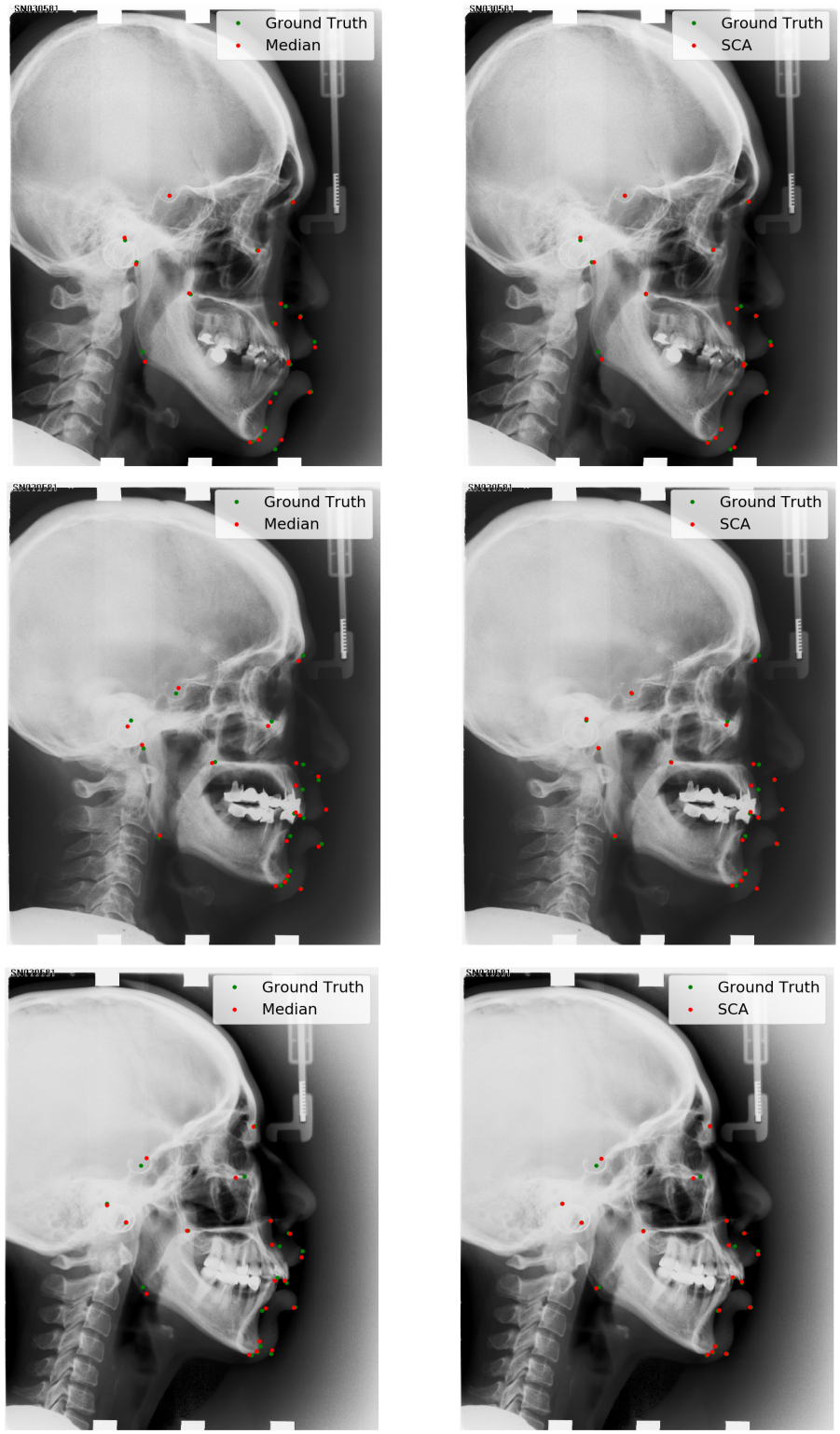


Figure 6.17: Sample of landmark detection on three CEPHA images. On the left, using the visual model with median. On the right, using SCA.

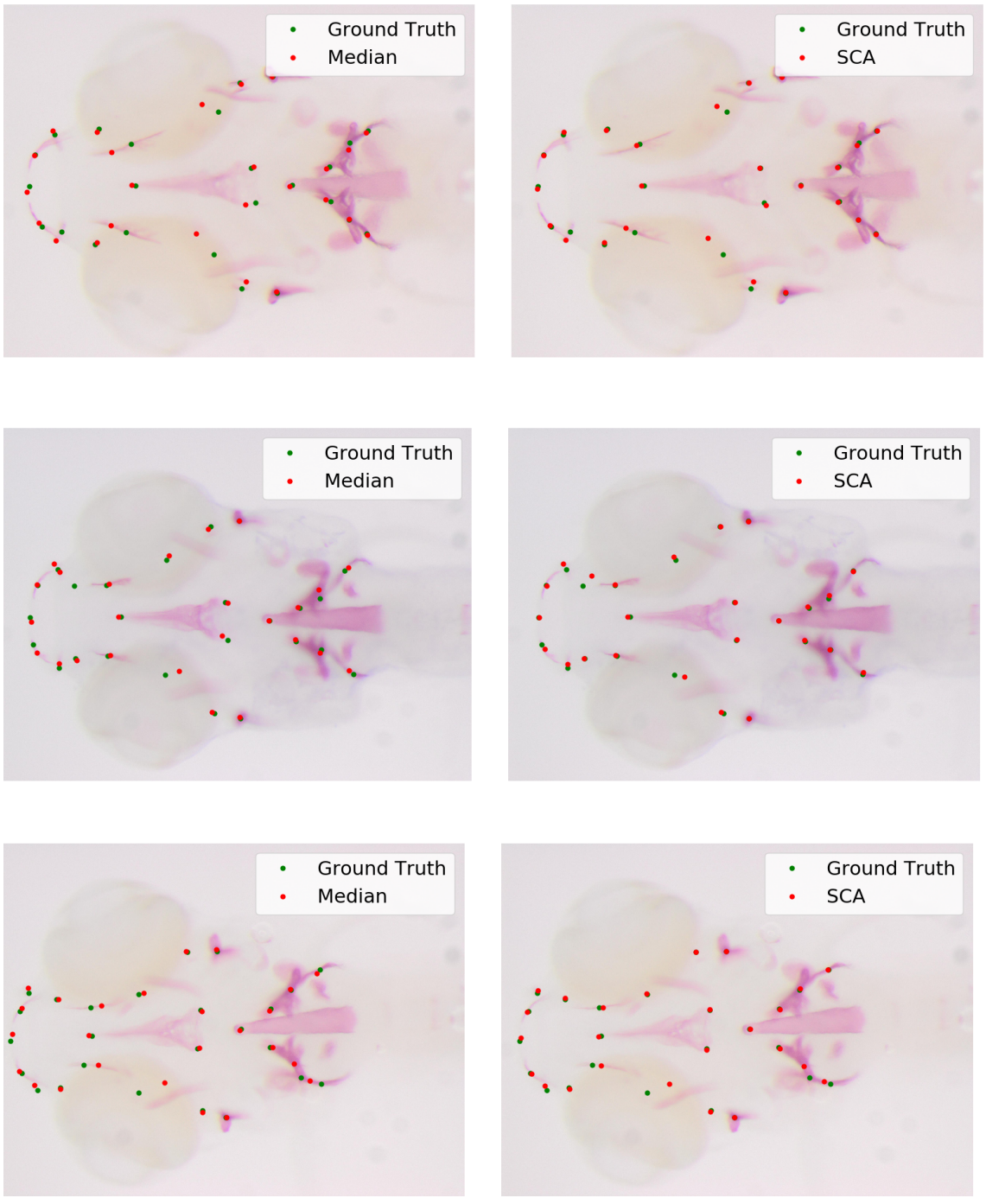


Figure 6.18: Sample of landmark detection on three ZEBRA images. On the left, using the visual model with median. On the right, using SCA.

6.5 Experiments with semi-automatic landmark annotation

In the previous section, we developed a new post-processing approach for refining landmark positions. The idea of this method is to first train a model for predicting the detection error of a given candidate landmark structure and then to use this model for finding the best possible landmark structure among a set of candidate structures highlighted through our visual model. In this section, we would to explore the possibility to improve this method by incorporating in the procedure corrections done by a human user on some of the predicted landmark positions. The main idea of our extension using human feedback is to use manual corrections to reduce the set of candidate landmark structures considered during post-processing, with the hope that this will help improving the prediction of the positions of the other uncorrected landmarks. Our study of this topic in this section is very preliminary and should be considered only as a first step towards turning our method into a semi-automatic approach for landmark correction.

In Section 6.5.1, we will first study for each dataset the evolution of the average detection error when the landmark positions are manually corrected and no modification is performed on the positions of the other landmarks. This result will serve as a baseline to assess the improvement brought by our proposed extension. The latter will be explained and studied empirically in Section 6.5.2.

6.5.1 Influence of human corrections on average detection error

Given some landmark detection results, we want first to evaluate the impact on the results of a possible human corrections of the automated landmark detection. This could potentially benefit to potential users, and will help us to get a better understanding of our method.

On the three datasets, we consider corrections of the predictions obtained with the visual model only (i.e., with the simple median post-processing) and with the SCA post-processing method. When simulating a human correction, we assume that the detection error becomes equal to 0 for each corrected landmark (i.e., the human makes no mistake), while in this section the predictions for the uncorrected landmark remains unchanged. For each image, we suppose that the observer would correct the N landmarks that were detected with the lowest accuracy. Figure 6.19 shows in these conditions the evolution of the mean landmark detection error as a function of the number of corrected landmarks. In other words, these curves show on each problem how many landmarks must be corrected manually if the average detection error is to be lower than a predefined threshold.

The first thing that can be observed is that it is always better to correct the predictions made by the method using post-processing. Even if the first corrections bring proportionally smaller improvements, the average detection error when correcting the positions obtained with post-processing are always smaller, for all of the datasets. The difference between the two methods is smaller with CEPHA and ZEBRA than with DROSO, but it is still observable in these cases. On the three datasets, the difference between the visual model and the post-processing method vanishes with the number of corrected landmarks.

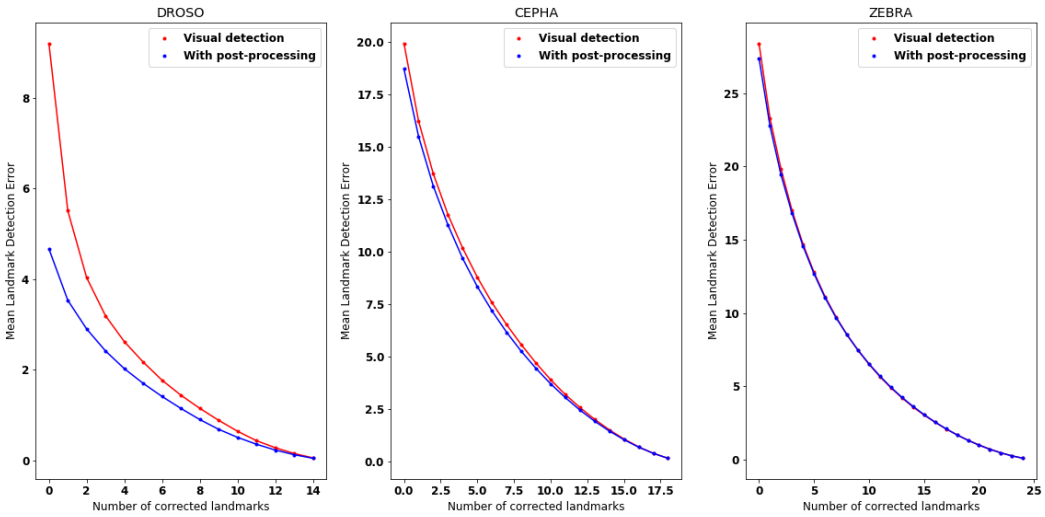


Figure 6.19: Impact of manual correction on the mean landmark detection error.

The correction of the positions of the 5 to 6 worst predicted positions will lead to a halving of the average detection error.

6.5.2 Exploiting human corrections at post-processing

The SCA post-processing method tries to select the best landmark structures from a set of candidate structures, where these structures are generated from the vote maps predicted by the visual model. The idea of the approach developed here is that the corrected landmark positions can be exploited to reduce the set of candidate landmark structures explored by the post-processing method. This could allow the method to detect the other landmarks more accurately. We give below a more formal presentation of the approach.

Landmark position prediction

Let us denote by $\mathcal{N}_L = \{1, \ldots, N_L\}$ the set of all landmark indices and by $\mathcal{K}_L \subset \mathcal{N}_L$ the indices of the landmarks that have been manually corrected on a test image T'. For this image, we want to compute an estimate $\hat{P}_l$ of the exact landmark positions P_l for all $l \in \mathcal{N}_L \backslash \mathcal{K}_L$. Given a visual landmark detection model, a landmark structure descriptor lsd, a structure scoring model S, the modified post-processing algorithm works as follows (steps that are modified with respect to the algorithm in Section 6.4.1 page 111 are highlighted in blue):

1. Use the visual model to extract the landmark vote maps V_l', for $l \in \mathcal{N}_l \setminus \mathcal{K}_l$ (the uncorrected positions).
2. For each $l \in \mathcal{N}_l \backslash \mathcal{K}_l$, extract N_C landmark candidate positions $\{C'(l,c) | c = 1, \ldots, N_C\}$ from each vote maps V_l' using the algorithm described in Section 6.4.1, page 111.

3. For each $l \in \mathcal{K}_l$, set all N_C candidate positions at the true position: $C'(l,c) = P_l$ for $c \in \{1,\ldots,N_C\}$.
4. Create N_V candidate landmark structures $d_t(i) = (C'(1,k_{i,1}),\ldots,C'(N_L,k_{i,N_L}))$, for $i = 1,\ldots,N_V$, with $k_{i,l}$ random integers $\in \{1,\ldots,N_C\}$.
5. Associate an error score $s_i = S(lsd(d_t(i)))$ to each structure $d_t(i)$ using the model S.
6. Let X be the set of the $r \times N_V$ indices with the smallest error scores: $X = \{j | s_j < sorted(s_i)[r \times N_V]\}$.
7. $\hat{P}_l = median(\{C'(l,k_{j,l}) | j \in X\})$, for $l = 1,\ldots,N_L$.
8. For each $l \in \mathcal{N}_L \setminus \mathcal{K}_L$, compute $\hat{P}_l = median(\{C'(l,k_{j,l}) | j \in X\})$.

Because of step 3, the positions of corrected landmarks is fixed to their true positions in all candidate landmark structures. Only the positions of the other landmarks are modified.

Generic versus specific scoring model training

The previous algorithm requires a scoring model. We will consider below two strategies to train this model. The first strategy is to train it using the exact same algorithm as in Section 6.4.1, i.e., without taking into account the fact that some landmark positions are corrected. We will call this strategy *generic training*. The main advantage of this approach is that a single scoring model needs to be trained and can be used whatever the landmarks that are corrected at prediction time. We will test below another strategy, called *specific training*, that takes into account the landmarks that are corrected. It works exactly as in the algorithm of Section 6.4.1 page 110, except that the positions of the landmarks corrected in the test images are now fixed to their exact positions in all N_E candidate landmark structures used for training the scoring model. Note that there is no need anymore to train a visual model for the corrected landmarks at the post-processing stage, although such model might still be required to provide the first predictions to be corrected by the user. This specific training has the drawback that it requires to train a specific scoring model for each potential subset of corrected landmarks. However, since the scoring model is trained in conditions that are closer to the conditions of its applications, we can expect some improvement with respect to generic training.

6.5.3 Results and observations

The tests that were performed for this analysis were all completed using leave-one-out cross-validation, using several sets of parameters for our post-processing method with manual correction. These tests were realized using the three datasets DROSO, CEPHA and ZEBRA. For each dataset, we analyzed the mean detection error brought by correcting the position of different numbers of landmarks: from 0 to all the landmarks of the dataset. When we simulate a correction, we modify the position of the landmark to its real position in the dataset, thus cancelling its detection error totally. For each test image, we consider

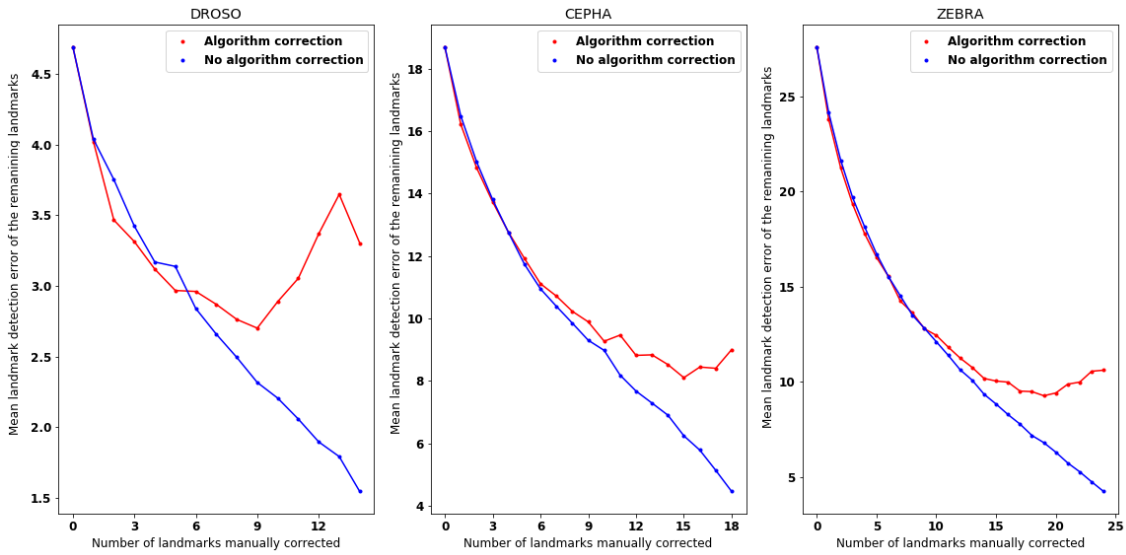


Figure 6.20: Comparison of the evolution of the error with and without applying our post-processing method on the corrected landmark positions.

that the observer corrects first the landmark positions that are the less accurately detected by the fully automatic approach. As a consequence, the corrected landmarks are likely to be different from one image to another. This also means that our model will need to focus on correcting landmark positions that were detected with relatively higher accuracy.

Figure 6.20 shows the detection error that our modified post-processing method obtains (with generic training), compared with the error obtained by these manual corrections, without modification of the positions of the uncorrected landmarks. The errors that are reported on the y-axis corresponds to the average error over the landmarks that are not manually corrected. For our method, we kept the set of parameters with the best average results over the whole numbers of corrected landmarks. We observe that our post processing method brings slight improvements when a small number of landmarks are corrected. Suprisinglt, when the number of corrected landmarks grows, it becomes better not to use the post-processing method to improve the positions of the other landmarks. On DROSO, the detection error for the uncorrected landmarks grows strongly when most landmarks are corrected. This might be due to the fact that the uncorrected landmarks are the one with the lowest detection errors. The accuracy of their detection might have reached their limit and post-processing them further only introduces further variance that deteriorates the detection error.

The results reported in Figure 6.21 compares the specific and the generic approaches for training the scoring model. There is no significant difference between the two approaches. This means that there is no need to build specific models for post-processing corrections, which significantly speeds up an assisting tool using the post-processing method for automated correction. We also tried our post-processing refinements if the corrections were

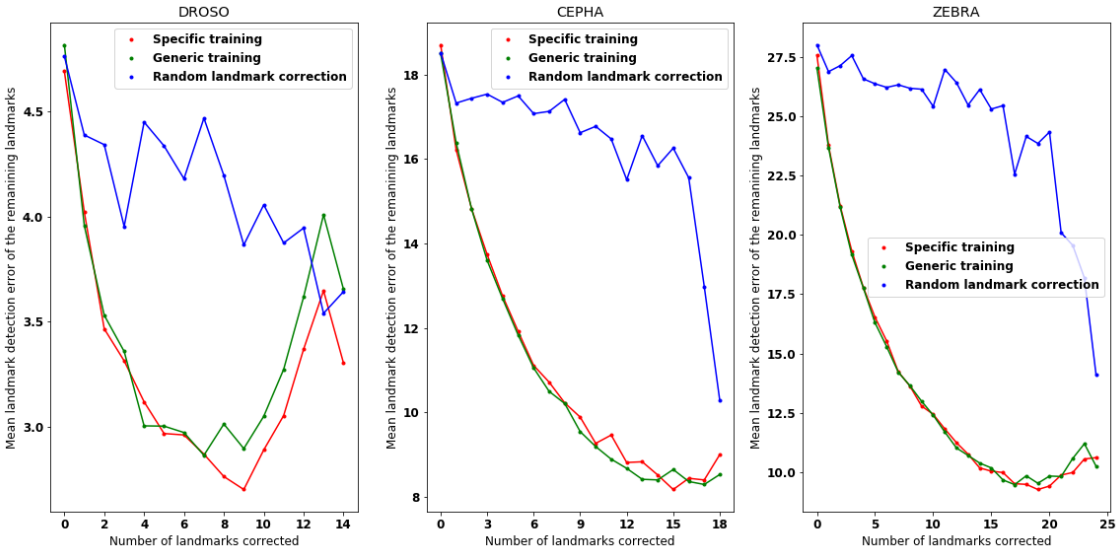


Figure 6.21: Comparison of different strategies for using the post-processing model on corrected landmark positions.

performed on landmarks selected randomly in order to consider the relevance of using our post-processing approach. We can see that even if the corrections are not as efficient as when the worst detections are corrected first by the observer, the average error over the remaining landmarks tends to decrease, which shows that our method is still able to bring improvements to the detection in specific scenarios.

6.6 Conclusion

The goal of this chapter was to propose a practical analysis of current state-of-the-art landmark detection methods through the prism of our own algorithm. Indeed, most landmark detection algorithms are divided into two similar steps: a first phase in which a visual model is used to generate vote maps for the likelihood of the landmark's appearances. In a second step, these vote maps are used in combination with a model of the global landmark structure in order to refine the final landmark's positions.

In Section 6.2, we analyzed, with our visual landmark detection algorithm, the influence of the number of images on the results, and extracted guidelines to help the potential user annotate his set of training images. The main observation that was made was that 30 images were enough for the easiest landmarks, while 70 images could be necessary for the hardest landmarks. On average on our three datasets, using additional annotated images will only give slightly better results on landmarks incorrectly detected.

In Section 6.3, we analyzed existing post-processing methods when applied on vote maps produced by our visual model. Our goal was to evaluate the impact of post-processing

methods on our results and compare these post-processing approaches to the simple median post-processing that we used in Chapter 4. We showed that correcting landmark positions with a given post-processing method was not always beneficial to the results when compared to our Median approach: the LC iterative post-processing approach only improved over median post-processing on DROSO, while the DMBL post-processing approach improved the results for DROSO, but significantly worsened them on ZEBRA. The main advantage of the LC post-processing approach is that it is easy and fast to learn, and is able to efficiently correct the larger visual errors and confusions, because it converges to the most likely shape. However, it fails to correct the smaller errors and sometimes even increases them, for the same reason. DMBL is able to efficiently correct the largest errors without increasing the errors on well detected landmarks only when the vote maps are accurate. With inaccurate vote maps, the model seems unable to understand which are the best candidates to select.

In Section 6.4, following these observations, we proposed a new post-processing method, SCA. Our post-processing approach, based on the same candidate selection than DMBL, does not try to directly learn the landmark structure, but learns the error made by the combination of landmark candidates output from the visual model. This method always significantly improves over median post-processing on our three datasets. We think that the main advantage of our method is that we only focus on the typical errors made by the visual model that was trained. Its main disadvantage is that the approach is difficult to tune, especially if several visual models are tested, and also introduces an important computational burden. While it is the most efficient approach, it could become difficult to apply it on landmark structures with large number of landmarks (more than 100, as in face detection for example), because of the exponential growth of possible landmark combinations.

In Section 6.5, we analyzed the impact of manual correction on the landmark detection. We saw that by correcting the positions of 5 to 6 landmarks on each image, the mean landmark detection error could be reduced by half. We proposed an extension of our post-processing method to take into account those human corrections to potentially improve the predicted positions of the uncorrected landmarks. Unfornutaly, we could not obtain satisfactory results. At the present stage of our experiments, the interest of using post-processing approaches with manual corrections is of very limited interest.

As future work, we think that it could be interesting to extend our post-processing approach to 3D images, and analyze its impact in this context. In the context of the SCA approach, it could also be interesting to test different ways to represent the landmark structure: we could for example try to extract a structure using feature reduction methods such as PCA, as proposed in LC [60]. Another interesting approach would be to analyze the results obtained using an unsupervised methodology such as Random Sample Consensus (RANSAC) [35], that could focus on the elimination of *outlier* structures among the candidates. Additionally, an in-depth study of methods able to take advantage of human correction could be interesting. We could for example try different structure descriptors, but also different ways to build the training dataset. In the specific training approach, it could be interesting for example to generate candidate structures for training the scoring model that includes some small perturbations around the positions of the manually

corrected landmarks, instead of keeping them fixed.

Chapter 7

Conclusion

In the first part of this chapter, we highlight the main results and observations that we have obtained during the realization of this work regarding the study and the development of supervised algorithms for the detection of landmarks on biomedical images. In a second time, we consider the perspectives concerning the future development of our supervised landmark detection algorithms for biomedical images, as well as for different problems that could potentially benefit from the use of such algorithms. Please notice that the different contributions (papers, posters, presentations, challenges...) were highlighted in the Introduction chapter.

7.1 Summary

Our work primarily focused on the development and the analysis of landmark detection methods for biomedical images. These algorithms were applied in two different contexts: morphometric analysis on biological bodies, and CT-CBCT image registration for radiotherapy. Along the chapters of this thesis, we have progressively described the improvements and extensions of this method, while comparing it to the different competitive approaches in the field.

Chapters 1 and 2 provided an introduction to the problematic of landmark detection tackled during this work as well as a short presentation of supervised machine learning and competitive methods.

In **Chapter 3**, we developed a basic method for the detection of landmarks based on supervised learning and we validated it on a cephalometric image database. This method is based on the construction of visual pixel classification or regression models that tries to evaluate whether or not the appearance of a pixel corresponds to the one of a given landmark. This approach uses multi-resolution pixel descriptors, as well as an intelligent approach aimed at reducing the number of pixels to be extracted both at the training of the learning model and at the prediction of the position of these different landmarks on new images. In this chapter, we reviewed the potential interest of these two ideas, and we

compared our method with state-of-the-art algorithms used in the context of a challenge associated to the cephalometric dataset. As a result of this latter comparison, it appears that the most competitive methods in the domain, like ours, mainly used Random Forests as a supervised learning algorithm.

In **Chapter 4**, we have extended the basic method presented in Chapter 3 in order to improve its accuracy and genericity on new datasets. We also performed an in-depth study of the influence of its different parameters. With this extension, we tried to use new ways to describe a pixel, and we proposed to parametrize some aspects of the algorithm, in order to adapt it more generically to new datasets. Using three databases of biological images with different numbers of images, landmarks, but also different resolutions and visual landmark representations to find, we were able to highlight the importance of correctly choosing the parameters of the landmark detection method. We have proven the importance of multi-resolution features on these three datasets, and compared ourselves to state-of-the-art algorithms that were reimplemented. On these three datasets, we obtained results equivalent or superior to the state of the art, while proposing a more generic and faster approach than these other algorithms. Finally, we also analyzed the robustness of our algorithm. We showed in particular that using raw pixel descriptors (RAW, SUB, GAUSSIAN SUB) could give competitive results. They were however negatively affecting the performances of the algorithm on the dataset with the largest deformations (DROSO).

In **Chapter 5**, we extended our method of landmark detection to 3D multimodal rigid image registration on a CT-CBCT radiotherapy dataset. The approach we proposed to tackle this image registration problem is to use our automated landmark detection algorithm in order to automatically find the positions of common anatomically interesting landmarks in the images of the two modalities. The positions of these landmarks are then matched by calculating the corresponding rigid transformation. This transformation is then applied to the whole images to be registered. During this study, we showed that this approach is competitive with the performances achieved by regular state of the art rigid registration algorithms, and that in addition, this approach does not require manual pre-selection of regions of interest. This gives to our approach a considerable advantage in processes where the speed of registration and the reduction of manual intervention are important factors, which is the case in the radiotherapy context studied in this work. However, we observed that the alignment results were not as good as what an operator could achieve by manually registering the two volumes without using any registration algorithm. We also showed that the use of multi-resolution voxel descriptors was less relevant in this particular context. Indeed, the landmarks being easily recognizable, the use of multi-resolution features seemed to bring less information than in the 2D context.

In **Chapter 6**, we performed three types of analysis for the practical use of the different landmark detection algorithms. First, we performed experiments that allowed us to extract general instructions that should assist in the manual annotation of a training image database for landmark detection algorithms using supervised learning. These instructions focused on the choice and number of images and landmarks that should be annotated. We also observed that the parameters of the detection method should not be specifically adapted to the number of images on which it was applied. The second part of this chapter focused on the analysis and the development of post-processing methods in the context

of landmark detection. We found that post-processing methods based on an unsupervised modeling of the landmark structure does not always improves the quality of the detection when compared to the simple consideration of a visual model. Following this observation, we developed a post-processing method based on the correction of the errors committed by the visual model. We showed that this algorithm was able to significantly improve the quality of the landmark detection in images. Finally, we proposed a simple approach to take into account the possibility of using human interaction in order to improve the results, and analyzed the possibilities brought by this method.

In addition, all algorithms proposed in Chapters 4 and 6 were implemented in Cytomine [66], an open-source platform for the analysis of (biomedical) images. Instructions on how to use the landmark detection module in Cytomine are given in the supplementary material of [66]. The source code is also available for direct use outside Cytomine, but can also be modified for further optimization and/or in order to fit different needs.

7.2 Future perspectives

To conclude this thesis, we consider in this section several ways to improve, extend, or use the landmark detection methods that we developed during this thesis.

7.2.1 Deep neural networks

Since the start of this thesis, deep neural networks have reached an unprecedented popularity in computer vision. Due to new methodological developments and to improvements of computing architectures and softwares, these methods have reached excellent performance in a large number of image processing tasks [53, 79, 55]. While we have based our approach on tree-based supervised learning techniques, it would be obviously interesting to investigate the use of deep neural networks for landmark detection. At the time this thesis is written, little concrete progress has however been made in the specific field of landmark detection in biomedical images using these methods. A few studies using neural networks to detect landmarks in cephalometry have been published [57, 4], but, in view of the results presented in these papers, neural networks do not seem to bring very significant improvements with respect to tree-based supervised learning methods. This conclusion must be taken with caution however, since the results reported in these different papers are obtained on datasets that are not available for comparison. Further work is therefore needed to extend these methods for landmark detection in the biomedical domain and to compare them fairly with state of the art tree-based methods. We discuss in the rest of this section several ways to apply neural networks for our problem.

We believe that the classical deep learning approach that would aim at creating a convolutional neural network directly outputting the landmark positions from the image pixel values will not be the most efficient in our context. Indeed, even if this type of approach seems to work well for face detection for example [7], we must consider the biomedical context on which we focused in this thesis. Indeed, those methods are generally mostly

successful on problems where large training datasets of images are available. However, as we have already pointed out, one of the central challenges in the biomedical domain is that we do not have this large number of training images. Moreover, we also have images with a high resolution from which it should be difficult to create a model able to accurately (pixel-wise) predict all of the landmarks positions. However, it seems to us that the use of neural networks remains possible in the context of our problem, if they are applied at the level of subwindows, as in our approach, or if appropriate data augmentation techniques are considered.

A first straightforward idea would be to simply replace the Random Forest model used in our method with a convolutional neural network, keeping all other steps of our algorithms unchanged, including the extraction of subwindows at training and at prediction as well as the use of multi-resolution descriptors. An additional small modification could also be to develop a neural network structure where different convolutional layers would be used for each of the resolutions we consider. We could then aggregate the output of these different networks into a final multi-layer perceptron in order to obtain a classification or regression result. In order to facilitate the training of these specific networks, at their different resolutions, we could use characteristics extracted from the internal layers of neural networks trained on very large numbers of images, such as GoogleNet [82], AlexNet [54] or ResNet [41]. This approach is used in many image classification tasks for the extraction of image features [97, 96]. A priori, we do not have conclusion about the performances that could be obtained with this approach. However, we expect that it would require a lot of training and validation time in order to give optimal results, contrary to the methods we have developed in this thesis.

Multi-layer perceptrons could also replace our Random Forests scoring model in the post-processing stage described in Chapter 6. Indeed, it seems very likely that these models could learn the structures formed by a combination of landmark candidate positions. In Chapter 6, we described this landmark structure using the distances and angles between the different landmarks candidates. This representation, while functional, remains debatable. Since neural networks are based on (non) linear combinations of input characteristics, we could imagine that instead of directly developing a definition of the structure, we could provide the raw coordinates of the landmark candidates to the model, which would then be tasked to discover the most relevant coordinate combinations in order to build the scoring model. In this context, it would be easy for us to generate a large number of candidate combinations as required to train multilayer perceptrons.

Moreover, it seems that certain techniques based on the use of deep neural networks would be useful in the detection of landmarks. For example, the Generative Adversarial Networks (GAN) [40] caught our attention. The principle of a GAN is to develop two neural networks. The first must generate artificial data from noise, and is optimized to confuse the second network, which is optimized to differentiate the artificial data generated by the first network from real data. Using this technique to generate new artificial images and/or annotations could potentially allow to overcome the problem of a lack of images and/or annotations in the biomedical domain.

Other approaches generally associated with deep neural networks could also be imagined

to tackle the problem of landmark detection. Given the good practical results obtained in this thesis, it seems to us, however, that the other supervised learning algorithms, and in particular the ensemble of trees, are not completely out of date, and in this context deserve to be at least considered as a point of comparison. This is also relevant since most neural network training methods rely on the intensive use of dedicated computing infrastructures, such as GPU clusters. Such infrastructures may indeed not be available to people who want to apply a landmark detection method to their own data.

7.2.2 Dealing with the lack of annotated images

As illustrated in this work, the main problem encountered was the annotation of the data. While many non-annotated images are potentially available, few images have been annotated with landmarks. This prevents us from using methods that require large numbers of landmarks and images to reach optimal performance. This also leads to potentially less interesting and representative results.

Chapter 6 has been partly developed to study some aspects of this question. However, it seems to us that it would be relevant to further study this particular problem. Several approaches can be considered in this direction.

First, as explored in Chapter 6, one possibility would be to use semi-automatic annotation methods, where the operators is assisted by the algorithm in his annotations. The study in Chapter 6 remains however very limited: we do not consider human annotation times or potential annotation problems related to the user interface. A first track would therefore be to improve this semi-automatic learning approach.

A second possible research direction could be to use data augmentation techniques to increase the number of annotated images. As stated above, GANs could be an interesting possibility. Along with the same idea, given the large number of non-annotated images often available in some domains, it would be interesting to take them into account during the development of the algorithm for improving landmark detection performances. This could be achieved by exploiting so-called semi-supervised or transductive learning methods [16].

One last possibility could be the use of transfer learning methods. The principle of transfer learning is to use a model developed for a specific application in the context of another application. This could for example mean using a visual model developed for detecting landmarks on drosophila images when the final goal is to develop a landmark detection model for zebrafish images. For example, a visual model developed for one kind of images could be useful to detect new, initially not available, landmarks in other kinds of images that might help finding the original landmarks in these latter images by incorporating the new landmarks in the post-processing step. We have indeed shown that increasing the number of landmarks could improve the performance of post-processing landmark correction techniques.

7.2.3 Extension of post-processing methods to 3D images

In the context of Chapter 6, we decided to focus on our 2D image databases for purely practical reasons: the validation of the post-processing approaches studied would have required a considerable number of additional computations. In addition, visual model detection errors on these volumes appeared to be more satisfactory than on 2D images, and therefore the potential contribution of post-processing methods seemed less interesting to study. However, there is no reason, that the value of these post-processing methods can not be studied further on 3D images. In order to carry out this analysis, we would also like to be able to study the potential impact of these post-processing methods on other types of 3D volumes: 3D cephalometry, other body regions, etc.

7.2.4 Integration of the visual and post-processing steps

All of the landmark detection methods that we have studied, including ours, perform the post-processing phase once the visual models have been trained. We believe that landmark detection methods could potentially benefit from the integration of both steps.

Chaining. The idea of this approach would be to detect a first landmark via a purely visual model. The second landmark would then be detected using a visual model that would also consider the position of the first detected landmark (for example, through its pixel descriptor). The third point would then be detected considering the position of the first two landmarks, and so on. Preliminary tests were carried out with this approach, which showed encouraging results.

Common descriptor. The idea of this approach is to simply describe a landmark structure through the combination of the visual descriptions of each potential candidate landmark positions as well as descriptors for the geometrical relationships between those landmarks. More concretely, for representing a given combination of landmark candidate positions, we would concatenate all the visual descriptors of each position, as well as the descriptors of the structure (angles and distances). Preliminary tests were also conducted with this approach, but obtained results were not conclusive: the models developed were not able to identify correctly the best candidate structures. Moreover, we have not managed to develop an efficient approach for selecting the best possible combination that avoids the combinatorial explosion due to the number of different landmark positions to test.

7.2.5 Interactive landmark detection

We think that one of the main challenges of future landmark detection algorithms will be to integrate the information brought by human experts in order to further increase detection performance.

We made a first experiment for approaching this challenge in Section 6.5. In this experiment, we observed the behavior of our post-processing model when corrections were made on the landmarks that were detected. We saw that the improvements brought by this approach were limited, but we think that there is still ample room for improvements. For example, other strategies to train the post-processing model when considering human interaction could be studied: for example, small perturbations could be added to the training dataset in order to help the model understand the variations between the landmark structures. Additionally, it could be interesting to retrain the visual detection models with this information. This approach would then become linked to the suggestions made in Section 7.2.4.

Finally, we only proposed one way to use informations coming from human observers, i.e., through the correction of some landmarks. Other possibilities could also be considered: the expert could for example influence the detection by highlighting region(s) of interest in which the landmark(s) could be found. Another possibility could also be to re-train the post-processing and visual models by using the corrections that were brought by the human observer on (a subsample of) the dataset on which the models were applied.

7.2.6 Application to other research domains

In this thesis, we focused on two practical applications that benefit from using automated landmark detection algorithms: morphometric studies as well as the rigid registration of images. However, we believe that other applications could benefit from the possibilities offered by the automatic detection of landmarks.

Non-rigid Image Registration In Chapter 5, we showed that it is possible to align images using our landmark detection method. We believe that by using more landmarks, intelligently distributed over the bodies to be registered, we could also calculate and apply non-rigid transformations using the same approach. In this context, an important part of the work will probably be to automate the choice of landmarks to use in order to obtain a representative non-rigid deformation once the landmarks are detected.

Image segmentation. A second application of the detection of landmarks could concern the segmentation of images. Indeed, by detecting landmarks, we could try to model the contours of an object, and therefore allow its segmentation.

Image classification and regression. Finally, its usefulness could also be in problems of classification and regression of more generic images: the landmarks could then be considered as intermediate information that can subsequently be reused to defined features used as inputs by another supervised algorithm. For example, we could imagine applications where distances between landmarks would be decisive in determining the class of an image.

Bibliography

[1] J. Aceto, R. Nourizadeh-Lillabadi, R. Marée, N. Dardenne, N. Jeanray, L. Wehenkel, P. Aleström, J. J. van Loon, and M. Muller. Zebrafish bone and general physiology are differently affected by hormones or changes in gravity. *PloS One*, 10(6):e0126928, 2015.

[2] T. Ahonen, A. Hadid, and M. Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, pages 2037–2041, 2006.

[3] D. Aneja, S. R. Vora, E. D. Camci, L. G. Shapiro, and T. C. Cox. Automated detection of 3d landmarks for the elimination of non-biological variation in geometric morphometric analyses. In *Proceedings of the IEEE 28th International Symposium on CBMS*, pages 78–83, 2015.

[4] S. Ö. Arık, B. Ibragimov, and L. Xing. Fully automated quantitative cephalometry using convolutional neural networks. *Journal of Medical Imaging*, 4(1):014501–014501, 2017.

[5] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 698–700, 1987.

[6] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *Proceedings of the European Conference on Computer Vision*, pages 404–417. Springer, 2006.

[7] S. Belharbi, C. Chatelain, R. Hérault, and S. Adam. Input/output deep architecture for structured output problems. *CoRR abs/1504.07550*, 2015.

[8] F. L. Bookstein. *Morphometric tools for landmark data: geometry and biology.* Cambridge University Press, New York, 1997.

[9] A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *Proceedings of the IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007.

[10] G. Bradski and A. Kaehler. *Learning OpenCV: Computer vision with the OpenCV library.* O'Reilly Media, Inc., 2008.

[11] L. Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[12] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[13] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. CRC press, 1984.

[14] P. A. Bromiley, A. C. Schunke, H. Ragheb, N. A. Thacker, and D. Tautz. Semi-automatic landmark point annotation for geometric morphometrics. *Frontiers in Zoology*, 11(1):1, 2014.

[15] X. P. Burgos-Artizzu, P. Perona, and P. Dollár. Robust face landmark estimation under occlusion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1513–1520, 2013.

[16] O. Chapelle, B. Schlkopf, and A. Zien. *Semi-Supervised Learning*. The MIT Press, 1st edition, 2010.

[17] N. Chazot, S. Panara, N. Zilbermann, P. Blandin, Y. Le Poul, R. Cornette, M. Elias, and V. Debat. Morpho morphometrics: Shared ancestry and selection drive the evolution of wing size and shape in morpho butterflies. *Evolution*, 70(1):181–194, 2016.

[18] C. Chen and G. Zheng. Fully-automatic landmark detection in cephalometric x-ray images by data-driven image displacement estimation. In *Proceedings of the ISBI International Symposium on Biomedical Imaging, Automatic Cephalometric X-Ray Landmark Detection Challenge*, pages 17–24, 2014.

[19] Y. J. Chen, S. K. Chen, H. F. Chang, and K. C. Chen. Comparison of landmark identification in traditional versus computer-aided digital cephalometry. *The Angle orthodontist*, 70(5):387–392, 2000.

[20] C. Chu, C. Chen, L. Nolte, and G. Zheng. Fully automatic cephalometric x-ray landmark detection using random forest regression and sparse shape composition. *Proceedings of the ISBI International Symposium on Biomedical Imaging, Automatic Cephalometric X-Ray Landmark Detection Challenge*, 2014.

[21] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001.

[22] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models-their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.

[23] A. Criminisi and J. Shotton. *Decision forests for computer vision and medical image analysis*. Springer Science & Business Media, 2013.

[24] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 2(4):303–314, 1989.

[25] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 886–893, 2005.

[26] A. E. F. de Oliveira, L. H. S. Cevidanes, C. Phillips, A. Motta, B. Burke, and D. Tyndall. Observer reliability of three-dimensional cephalometric landmark identification on cone-beam computerized tomography. *Oral Surgery, Oral Medicine, Oral Pathology, Oral Radiology, and Endodontology*, 107(2):256–265, 2009.

[27] V. Debat, S. Bloyer, F. Faradji, N. Gidaszewski, N. Navarro, P. Orozco-terWengel, V. Ribeiro, C. Schlötterer, J. S. Deutsch, and F. Peronnet. Developmental stability: a major role for cyclin g in drosophila melanogaster. *PLoS Genet*, 7(10):e1002314, 2011.

[28] R. Donner, B. H. Menze, H. Bischof, and G. Langs. Global localization of 3d anatomical structures by pre-filtered hough forests and discrete optimization. *Medical Image Analysis*, 17(8):1304–1314, 2013.

[29] N. Duta and M. Sonka. Segmentation and interpretation of mr brain images. an improved active shape model. *IEEE Transactions on Medical Imaging*, 17(6):1049–1062, 1998.

[30] I. El-Feghi, M. A. Sid-Ahmed, and M. Ahmadi. Automatic localization of craniofacial landmarks for assisted cephalometry. *Pattern Recognition*, 37(3):609–621, 2004.

[31] G. Fanelli, M. Dantone, J. Gall, A. Fossati, and L. Van Gool. Random forests for real time 3d face analysis. *International Journal of Computer Vision*, 101(3):437–458, 2013.

[32] J. L. Fearon and D. J. Varricchio. Morphometric analysis of the forelimb and pectoral girdle of the cretaceous ornithopod dinosaur oryctodromeus cubicularis and implications for digging. *Journal of Vertebrate Paleontology*, 35(4):e936555, 2015.

[33] A. Fedorov, R. Beichel, J. Kalpathy-Cramer, J. Finet, J.-C. Fillion-Robin, S. Pujol, C. Bauer, D. Jennings, F. Fennessy, M. Sonka, et al. 3d slicer as an image computing platform for the quantitative imaging network. *Magnetic Resonance Imaging*, 30(9):1323–1341, 2012.

[34] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, 2005.

[35] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[36] A. F. Frangi, W. J. Niessen, K. L. Vincken, and M. A. Viergever. Multiscale vessel enhancement filtering. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 130–137. Springer, 1998.

[37] J. Friedman, T. Hastie, and R. Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[38] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, 2006.

[39] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[40] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[41] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[42] D. L. Hill, P. G. Batchelor, M. Holden, and D. J. Hawkes. Medical image registration. *Physics in Medicine and Biology*, 46(3):R1, 2001.

[43] B. Ibragimov, B. Likar, F. Pernus, and T. Vrtovec. A game-theoretic framework for landmark-based image segmentation. *IEEE Transactions on Medical Imaging*, 31(9):1761–1776, 2012.

[44] H. Johnson, G. Harris, K. Williams, et al. Brainsfit: mutual information rigid registrations of whole-brain 3d images, using the insight toolkit. *Insight J*, pages 1–10, 2007.

[45] R. Kafieh, S. Sadri, A. Mehri, and H. Raji. Discrimination of bony structures in cephalograms for automatic landmark detection. *Advances in Computer Science and Engineering*, pages 609–620, 2009.

[46] S. Kamath, W. Song, A. Chvetsov, S. Ozawa, H. Lu, S. Samant, C. Liu, J. G. Li, and J. R. Palta. An image quality comparison study between xvi and obi cbct systems. *Journal of Applied Clinical Medical Physics*, 12(2), 2011.

[47] A. Kamoen, L. Dermaut, and R. Verbeeck. The clinical significance of error measurement in the interpretation of treatment results. *The European Journal of Orthodontics*, 23(5):569–578, 2001.

[48] A. Kaur and C. Singh. Automatic cephalometric landmark detection using zernike moments and template matching. *Signal, Image and Video Processing*, 9(1):117–132, 2013.

[49] Y. Kessentini, T. Paquet, and A. B. Hamadou. Off-line handwritten word recognition using multi-stream hidden markov models. *Pattern Recognition Letters*, 31(1):60–70, 2010.

[50] S. Klein, M. Staring, K. Murphy, M. A. Viergever, and J. P. Pluim. Elastix: a toolbox for intensity-based medical image registration. *IEEE Transactions on Medical Imaging*, 29(1):196–205, 2010.

[51] C. P. Klingenberg. Evolution and development of shape: integrating quantitative approaches. *Nature Reviews Genetics*, 11:623–635, 2010.

[52] G. Köksal, İ. Batmaz, and M. C. Testik. A review of data mining applications for quality improvement in manufacturing industry. *Expert systems with Applications*, 38(10):13448–13467, 2011.

[53] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[54] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, pages 1097–1105. Curran Associates, Inc., 2012.

[55] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[56] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. volume 86, pages 2278–2324, 1998.

[57] H. Lee, M. Park, and J. Kim. Cephalometric landmark detection in dental x-ray images using convolutional neural networks. In *Proceedings of the SPIE Medical Imaging*, pages 101341W–101341W. International Society for Optics and Photonics, 2017.

[58] C. Lindner, P. A. Bromiley, M. C. Ionita, and T. F. Cootes. Robust and accurate shape model matching using random forest regression-voting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1862–1874, 2015.

[59] C. Lindner and T. F. Cootes. Fully automatic cephalometric evaluation using random forest regression-voting. In *IEEE International Symposium on Biomedical Imaging*, 2015.

[60] C. Lindner, C.-W. Wang, C.-T. Huang, C.-H. Li, S.-W. Chang, and T. F. Cootes. Fully automatic system for accurate localisation and analysis of cephalometric landmarks in lateral cephalograms. *Scientific Reports*, 6, 2016.

[61] G. Louppe, L. Wehenkel, A. Sutera, and P. Geurts. Understanding variable importances in forests of randomized trees. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 431–439. Curran Associates, Inc., 2013.

[62] P. Lukashevich, B. Zalesky, and S. Ablameyko. Medical image registration based on surf detector. *Pattern Recognition and Image Analysis*, 21(3):519, 2011.

[63] P. W. Major, D. E. Johnson, K. L. Hesse, and K. E. Glover. Effect of head orientation on posterior anterior cephalometric landmark identification. *The Angle orthodontist*, 66(1):51–60, 1996.

[64] R. Marée, P. Geurts, and L. Wehenkel. Towards generic image classification using tree-based learning: An extensive empirical study. *Pattern Recognition Letters*, 74:17–23, 2016.

[65] R. Marée and R. Hoyoux. Cytomine user guide, October 2016.

[66] R. Marée, L. Rollus, B. Stévens, R. Hoyoux, G. Louppe, R. Vandaele, J.-M. Begon, P. Kainz, P. Geurts, and L. Wehenkel. Collaborative analysis of multi-gigapixel imaging data using cytomine. *Bioinformatics*, 2016.

[67] H. Mirzaalian and G. Hamarneh. Automatic globally-optimal pictorial structures with random decision forest based likelihoods for cephalometric x-ray landmark detection. In *Proceedings of the ISBI International Symposium on Biomedical Imaging, Automatic Cephalometric X-Ray Landmark Detection Challenge*, pages 25–36. IEEE, 2014.

[68] K. P. Murphy. *Machine learning : a probabilistic perspective*. MIT Press, Aug. 2013.

[69] T. Ojala, M. Pietikainen, and D. Harwood. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, volume 1, pages 582–585. IEEE, 1994.

[70] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research*, 12:2825–2830, 2011.

[71] J. P. Pluim, J. A. Maintz, and M. A. Viergever. Mutual-information-based registration of medical images: a survey. *IEEE Transactions on Medical Imaging*, 22(8):986–1004, 2003.

[72] J. R. Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[73] T. Rakosi. An atlas and manual of cephalometric radiology. *London, UK: Wolfe Medical*, 1982.

[74] B. S. Reddy and B. N. Chatterji. An fft-based technique for translation, rotation, and scale-invariant image registration. *IEEE Transactions on Image Processing*, 5(8):1266–1271, 1996.

[75] M. Rogers and J. Graham. Robust active shape model search. In *Proceedings of the European Conference on Computer Vision*, pages 517–530. Springer, 2002.

[76] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015.

[77] A. Rosas, L. Pérez-Criado, M. Bastir, A. Estalrrich, R. Huguet, A. García-Tabernero, J. F. Pastor, and M. De la Rasilla. A geometric morphometrics comparative analysis of neandertal humeri (epiphyses-fused) from the el sidrón cave site (asturias, spain). *Journal of Human Evolution*, 82:51–66, 2015.

[78] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533, 1986.

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

[80] O. Stern, R. Marée, J. Aceto, N. Jeanray, M. Muller, L. Wehenkel, and P. Geurts. Automatic localization of interest points in zebrafish images with tree-based methods. In *Proceedings of the IAPR International Conference on Pattern Recognition in Bioinformatics*, pages 179–190. Springer, 2011.

[81] F. M. Sukno, S. Ordas, C. Butakoff, S. Cruz, and A. F. Frangi. Active shape models with invariant optimal features: Application to facial analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(7):1105–1117, 2007.

[82] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[83] T. van der Niet, C. P. Zollikofer, M. S. P. de León, S. D. Johnson, and H. P. Linder. Three-dimensional geometric morphometrics for studying floral shape variation. *Trends in plant science*, 15(8):423–426, 2010.

[84] B. Van Ginneken, A. F. Frangi, J. J. Staal, B. M. ter Haar Romeny, and M. A. Viergever. Active shape model segmentation with optimal features. *IEEE Transactions on Medical Imaging*, 21(8):924–933, 2002.

[85] R. Vandaele, J. Aceto, M. Muller, F. Péronnet, V. Debat, C.-W. Wang, C.-T. Huang, S. Jodogne, P. Martinive, P. Geurts, and R. Marée. Landmark detection in 2d bioimages for geometric morphometrics: a multi-resolution tree-based approach. *Scientific Reports*, 8(1):538, 1 2018.

[86] R. Vandaele, F. Lallemand, P. Martinive, A. Gulyban, S. Jodogne, P. Coucke, P. Geurts, and R. Marée. Automated multimodal volume registration based on supervised 3d anatomical landmark detection. In *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 2017.

[87] R. Vandaele, R. Marée, S. Jodogne, and P. Geurts. Automatic cephalometric x-ray landmark detection challenge 2014: A tree-based algorithm. *Proceedings of the ISBI International Symposium on Biomedical Imaging, Automatic Cephalometric X-Ray Landmark Detection Challenge*, 2014.

[88] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–511. IEEE, 2001.

[89] P. Vučinić, Ž. Trpovski, and I. Šćepan. Automatic landmarking of cephalograms using active appearance models. *The European Journal of Orthodontics*, 32(3):233–241, 2010.

[90] C.-W. Wang, C.-T. Huang, J.-H. Lee, C.-H. Li, S.-W. Chang, M.-J. Siao, T.-M. Lai, B. Ibragimov, T. Vrtovec, O. Ronneberger, P. Fischer, T. F. Cootes, and C. Lindner. A benchmark for comparison of dental radiography analysis algorithms. *Medical Image Analysis*, 2016.

[91] C.-W. W. Wang, C.-T. Huang, M.-C. Hsieh, C.-H. Li, S.-W. Chang, W.-C. Li, R. Vandaele, R. Maree, S. Jodogne, P. Geurts, C. Chen, G. Zheng, C.-W. Chu, H. Mirzaalian, G. Hamarneh, T. Vrtovec, and B. Ibragimov. Evaluation and comparison of anatomical landmark detection methods for cephalometric x-ray images: A grand challenge. *IEEE Transactions on Medical Imaging*, 34(9):1890–1900, 2015.

[92] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.

[93] H. Yang and I. Patras. Sieving regression forest votes for facial feature detection in the wild. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1936–1943. IEEE, 2013.

[94] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. *Exploring artificial intelligence in the new millennium*, 8:236–239, 2003.

[95] W. Yue, D. Yin, C. Li, G. Wang, and T. Xu. Automated 2-d cephalometric analysis on x-ray images by a model-based approach. *IEEE Transactions on Biomedical Engineering*, 53(8):1615–1623, 2006.

[96] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4694–4702, 2015.

[97] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2929, 2016.

[98] B. Zitova and J. Flusser. Image registration methods: a survey. *Image and Vision Computing*, 21(11):977–1000, 2003.