Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Christian Hans-Pieter Kat Echtermeyer

Biometrical Identification and Feature Investigation of Salmon

Master's thesis in Computer Science
Supervisor: Theoharis Theoharis
June 2020

NTNU
Norwegian University of Science and Technology

Christian Hans-Pieter Kat Echtermeyer

Biometrical Identification and Feature Investigation of Salmon

Master's thesis in Computer Science
Supervisor: Theoharis Theoharis
June 2020

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU

Kunnskap for en bedre verden

Acknowledgements

I would not have been able to perform this research without key individuals, which have helped me throughout this project. First and foremost, i would like to thank my professor and supervisor, Theoharis Theoharis, for providing facilities and valuable guidance throughout this project. A big thanks Aleksander Børresen Eilertsen, Jonatan Sjølund Dyrstad, and Morten Steen Bondø from the SINTEF SalmID project, which provided the dataset for this research. I would like to thank Helena Koch Haugane, Vidar Melstveit, and Marcus Nickelsen from the NTNU School of Entrepreneurship for their initial ideas and work on this subject. Lastly, thanks to Bart Iver van Blokland and Michael Gimle for their ideas and suggestions.

Preface

The basis for this research stemmed from my passion for computer vision, artificial intelligence, and love for innovation. The combination of these led me to join a tech startup as a co-founder and sole developer, which focused on developing computer vision technology for use in the Norwegian salmon industry. Initially focusing on biometrical identification of salmon, the startup pivoted over to different ideas. As my field was no longer as relevant, I also decided to move on. Nevertheless, the idea of biometrically identifying salmon still fascinated me. Additionally, the lack of detailed and descriptive work on this subject led me to pursue this academically.

Abstract

Spots form unique patterns on many different species in the animal kingdom and have been used in combination with computer vision to identify individuals. This work creates an identification pipeline for salmon based on spot patterns located on the head of the fish. The pipeline primarily consists of three parts: extracting biometrical features, converting these into numerical feature vectors, and the matching algorithm itself.

The features are extracted using three different neural networks created previously. A unique coordinate system was constructed utilizing the eye and a novel anchor point: the pectoral fin. Location, shape, and size are extracted from the spots and saved in a database for matching. An algorithm was created, achieving similar results to the current state of the art, using less and possibly more reliable biometrical features. An experiment was conducted matching the largest number of unique salmon to date, containing 1000 unique identities and achieving a correct match percentage of 96.83%, setting a new baseline. A confidence rating was established, allowing for a customizable trade-off between precision and recall. The work also contributes a database containing biometrical features and trained neural networks for future work.

Sammendrag

Prikker danner unike mønster på mange forskjellige arter i dyreriket og har blitt brukt i kombinasjon med datasyn til å identifisere individer. I denne oppgaven beskrives prosessen av å skape en identifikasjonspipeline for laksegjenkjenning basert på prikkemønster. Pipelinen består hovedsaklig av tre deler: ekstrahering av biometriske trekk, konvertering av disse til numeriske beskrivelsesvektorer, og gjenkjenningsalgoritmen.

Trekkene blir ekstrahert ved bruk av tre forskjellige nevrale nettverk fra tidligere arbeid. Et unikt koordinatsystem ble konstruert ved hjelp av øye og et nytt ankerpunkt: brystfinnen til fisken. Lokasjon, form og størrelse av prikkene blir ekstrahert og lagret i en database for matching. En algoritme ble konstruert som oppnår lignende resultater som nåværende state-of-the-art, men ved bruk av mindre og kanskje mer pålitelige biometriske trekk. Et eksperiment ble gjennomført for å matche den største mengden unike laks til dags dato, 1000 unike identiteter, som oppnådde 96.83% true positive og setter dermed en ny grunnlinje. Et sikkerhetsvurderingssystem ble opprettet som åpner for en avveining mellom precision og recall. Dette arbeidet tilfører også en database med biometrisk data og trente nevrale nettverk for fremtidig arbeid.

Contents

List of Figures

Acronyms and Terms

Acronyms

AI Artificial Intelligence

OpenCV Open Source Computer Vision Library

HOG Histogram of Oriented Gradients

TP True Positive

FP False Positive

FN False Negative

TN True Negative

MS Match Score

MSD Match Score Difference

PMSD Proportional Match Score Difference

Chapter 1

Introduction

1.1 Motivation and Background

The Atlantic Salmon (*Salmo salar*) is a valuable Norwegian resource with an export value of 51 billion NOK in 2018 [1]. Like many other industries, the farming of salmon is regulated and requires health inspections and close monitoring of the fish. Today this is a manual process, inspection of fish is done by workers who take a sample of fish out of the net and inspect for injuries, lice, and other health criteria. Inspection is time-consuming and requires resources to conduct. For example, the industry standard of measuring the number of sea lice (*Lepeophtheirus salmonis*) on salmon is taking a sample of 10-25 fish out of the net and counting manually [2]. However, the number of fish in a single farm net is around 200 000. For every net inspected, only a small fraction of the fish are sampled. The use of high-quality cameras has become standard in most fish farms, mainly due to the need to monitor feeding manually. Monitoring is primarily done to avoid wasting costly food for the fish and avoid too much feed getting into the local environment around the fish farms. The hardware technology to monitor the fish at all times is already readily available. This opened up many commercial possibilities for the use of computer vision within the industry, already there are claims and patents concerning everything from biomass estimations to automatic lice counting [[3], [4], [5], [6]] as well as some scientific papers on the subjects [[7], [8]].

Another issue is that of monitoring the stress level of fish. It is a known fact that stressed fish do not put on as much weight as non stressed fish [9], [10], the effects on salmon in fish farms have also been looked at specifically [11]. Stressed fish show external signs [12], such

as changed behavior and discoloring. Physical signs can, in theory, be detected by humans and computers. These tasks are not necessarily reliant on knowing the identity of the fish in question, as reasonable estimations can be done using statistical models. However, all these tasks would benefit from knowing the identity of the fish being analyzed to give a more accurate picture of the reality.

Although statistical models can be a replacement for identification, there are even more compelling cases where there are good reasons for individual tracking. Individual tracking could allow for accurately following the health and condition of an individual over time. Recent studies have found that there is substantial additive genetic variation in resistance to the sea lice [13]. Tracking which salmon are genetically less predisposed to lice in real conditions could be done on a large scale with individual tracking. In general, this would enable a whole new world of data gathering, allowing for optimization. Thousands of fish could be tracked throughout their life cycle and investigated through computer vision. This would allow for an increase in scientific knowledge surrounding the Atlantic Salmon and provide the basis for commercial optimization.

1.2 Objective

This thesis sets out to use numerical algorithms to identify salmon by using spots on the head of the fish. The main focus of the work is to provide a detailed approach to creating a biometrical identification pipeline and investigate previously unused features that can potentially also improve approaches by other researchers.

The extraction of spots has been completed in the Fall Project linked to this master thesis.

1.3 Acronyms and Terms

Acronyms and terms used in this report are described after the list of figures in : Acronyms and Terms.

1.4 Structure of the Report

This report is structured as follows: Chapter 2 contains related work, theory, and choice of technology. Chapter 3 details the methods used during this project. Chapter 4 details the results of the detection pipeline and the investigation of the matching algorithm. Lastly chapter 5 discusses details and results. Conclusions and further work are presented in chapter 6.

Chapter 2

Background and Related Work

2.1 Overview

This chapter considers related work and general theory. Parts of this chapter are taken from the fall project on this subject by the author [14].

2.2 Fall Project by the Author

In the Fall Project leading up to this thesis, initial research was conducted [14] to establish the foundation for the biometric data extraction. This project created three neural networks that utilized thousands of hand-labeled images to extract the desired features for use in biometric matching. The full project is available online [14].

The goal was to identify important biometrical features and points and create a pipeline for extracting these features for later use in biometrical matching.

Spots on the skin of the salmon were used as the basis for biometrical identification. Specifically, spots on the head of the fish as these are on the most rigid part of the fish. Therefore they do not experience as much deformation during swimming. Additionally, the eye and pectoral fin were identified as interesting anchor points to be used to create a unique coordinate system. This enables the system to account for scaling and rotation.

The features were extracted using three separate neural networks: one for extracting the location of the eye and the fin in the image, one for extracting the bounding box of the head,

and lastly, one for segmenting spots inside the extracted bounding box. These neural networks were trained using hand-labeled images requiring significant labeling work.

The results of the networks were satisfactory and could potentially be used for biometrical identification. Images of the results from the eye-fin, head and segmentation network can be found in Figures [2.1, 2.2, 2.3] respectively. Figure 2.1 shows the results from the eye-fin network, which finds the location of the eye and fin to be used as a basis for the unique coordinate system. Figure 2.2 shows the results from the head network, which extracts the head region of the fish, which is later passed on to the segmentation network for identifying spots. Some results from the segmentation network can be seen in Figure 2.3, which classifies every pixel in the head region extracted as either spot or not spot.

Figure 2.1: An example image showing the results of the eye-fin network. This network predicts the location of the eye and fin of a fish.
Arrows: The arrows point towards the interest points in the first image for easy spotting.
red: eye prediction.
blue: pectoral fin prediction.

Figure 2.2: An example image showing the results of the head network. This network predicts the location of the head of a fish.

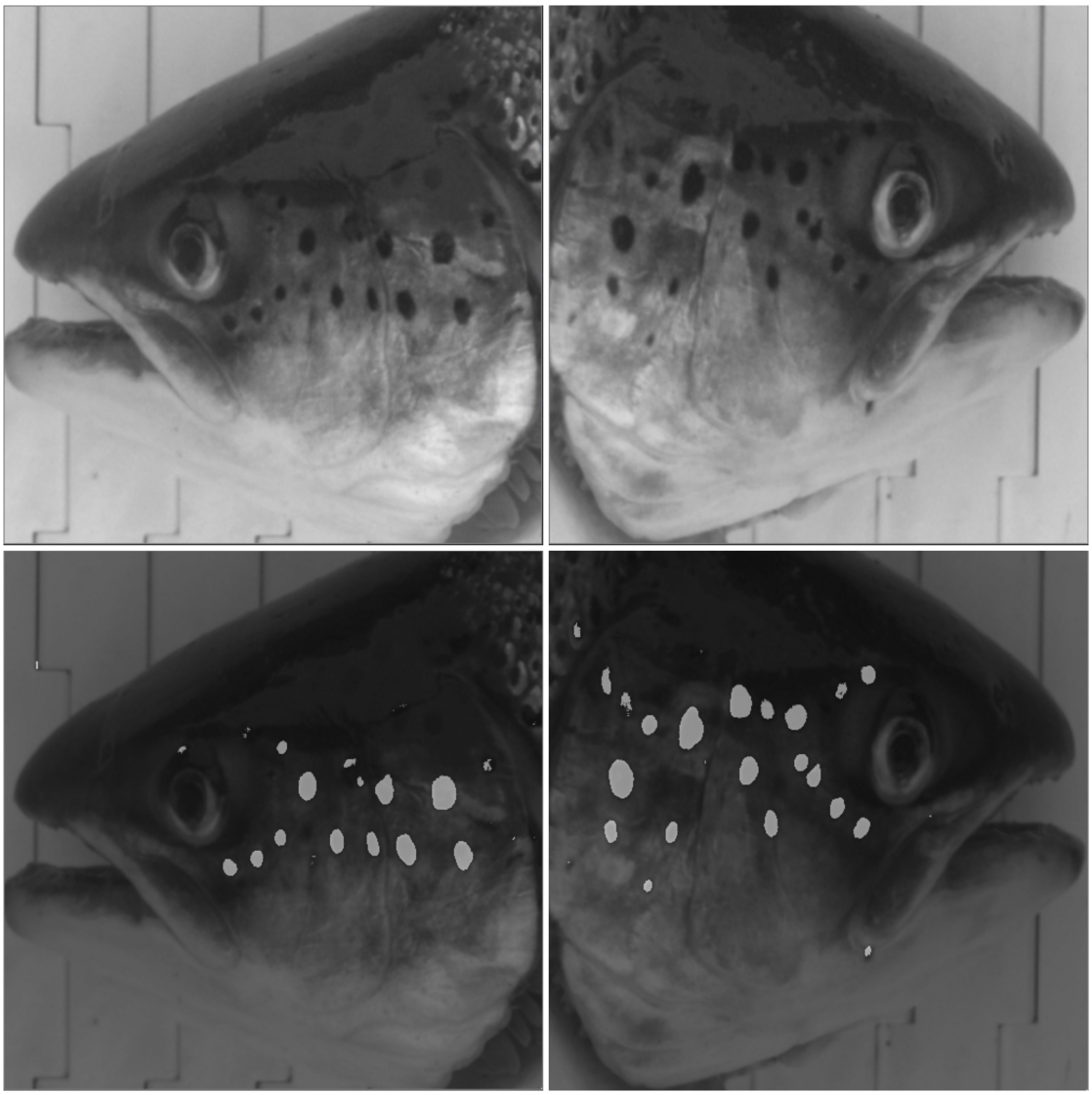


Figure 2.3: An example image showing the results of the segmentation network. This network segments the spots on the head of the fish. Original image on top and overlaid results on the bottom.

2.3 Related Work

2.3.1 Salmon Identification

A study was done by Eilertsen et al. [15] to explore the possibility of identifying salmon without physical contact using the pattern of spots on the skin of the salmon. An additional objective was to explore whether the pattern on the skin is unique enough for identification.

The researchers created a dataset using three separate cameras with slightly different angles

and locations. This rig enabled the researchers to model slight movement and different views of the same fish, allowing for the testing of re-identification and cross-validation. Images of fish on a conveyor belt were collected and labeled.

To extract the skin pattern on the fish, the researchers first attempted global thresholding. After that, they tried using a deep learning approach, which yielded better results. The eye was also identified and used as an anchor point for a coordinate system. The second point used to establish a coordinate system was the shortest point from the eye to the end of the skull (see Figure 2.4). The deep learning method and network, as well as the method for finding the eye, are not mentioned.

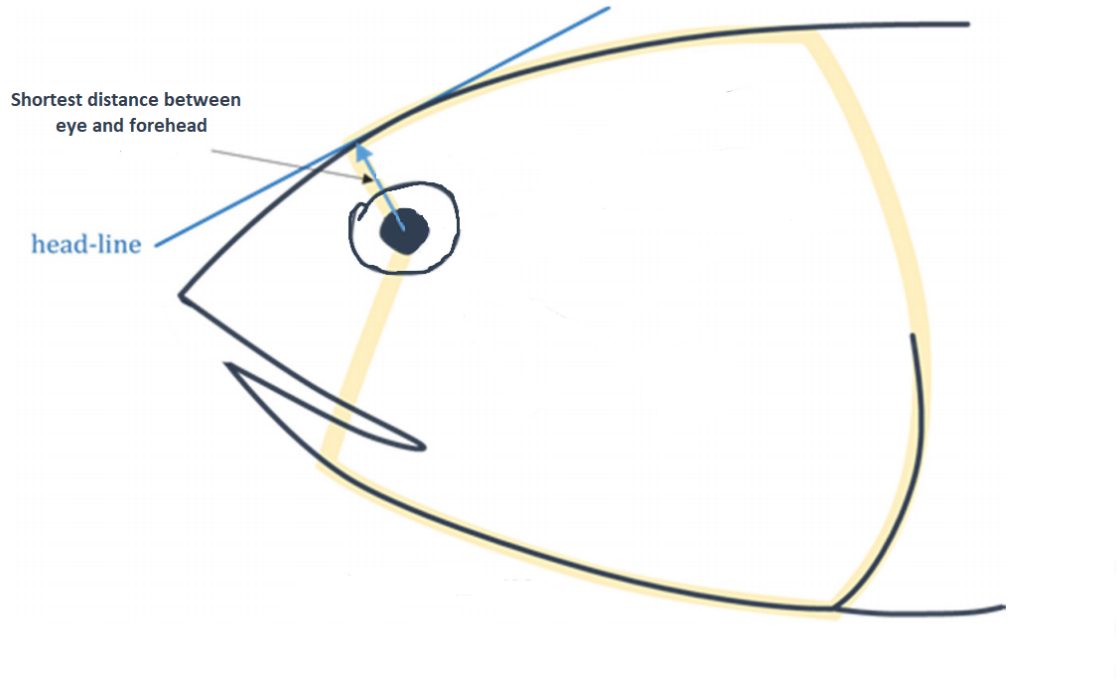


Figure 2.4: An illustration by Eilertsen et al [15], edited to show how the researchers created a coordinate system. The two anchor points are the eye and the shortest distance from eye to the end of the skull.

Using the extracted features, the researchers proposed two methods for creating identities.

First was a polar coordinate approach using the eye as a center and the head line as the 0-degree line. This meant that each spot could be mapped and compared to others. Both location and size of the spots were used to create identity data.

The second method is the star constellation approach, used in a similar task to recognize patterns on whale sharks by Arzoumanian et al. [16]. Originally this algorithm was used in

astronomy to match star patterns, hence the name star constellation approach [17].

> "It (constellation algorithm) has been demonstrated to be reliable even when the two lists of coordinates have as few as 25% of their points in common."
> - Arzoumanian et al. [16]

The algorithm creates constellations from a list of spots, which can then be compared to find matches.

The polar algorithm did not appear to produce promising results. Using the star constellation algorithm on the extracted spots, the researchers managed to achieve a 99% accuracy on a dataset consisting of 361 fish. This is concluded as promising for the skin pattern being unique. The small size of the dataset is mentioned as a weakness.

2.3.2 Whale Shark Identification

In a study done by Arzoumanian et al. [16], researchers set out to create a computer-aided identification system for identifying whale sharks. Many encounters were photographed, but since whale sharks are not easily identifiable by humans, a numerical approach was desired. Therefore the researchers set out to create a pipeline to extract the features from the skin and create an identity from these.

The data consisted of photographs taken of encounters with whale sharks. Most of the data was collected by the researchers themselves, but some of it was contributions from other sources.

Feature extraction was done by an image processing pipeline, yet it is slightly unclear if any humans are involved at any point or whether it was a fully automatic process. First, the image is rotated by analyzing the horizontal line that outlines the vertebrate of the fish. A region of interest is extracted behind the gills of the whale shark (see Figure 2.5). This image is then processed to remove as much noise as possible before a blob detection algorithm is run to extract the spots on the skin.

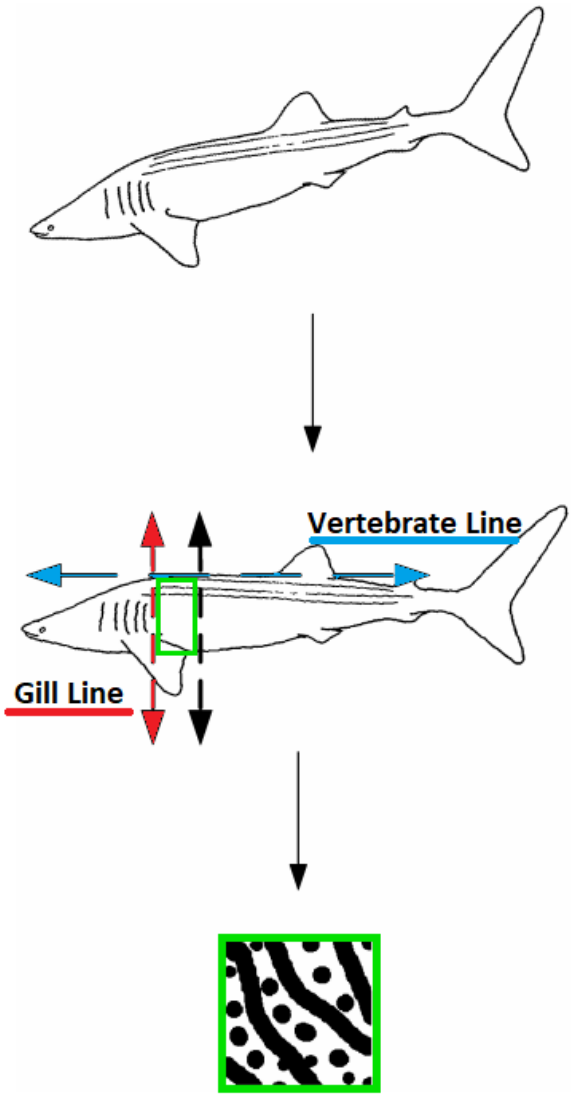


Figure 2.5: An edited illustration from the paper [16] showing the region to be extracted.
Blue Line: Vertebrate line.
Red Line: Gill Line.
Green Box: Region of Interest

The biometric matching was done using a modified star constellation algorithm, which was initially described by Groth [17]. The researchers modified the algorithm to be more robust towards the deformation of the whale sharks' body.

Due to the fact that an identity label has to be made by a human, the results rely on correct human labeling. Twenty-seven images of previously identified pairs of encounters the algorithm resulted in 21 strong matches and four moderate matches. The two remaining images were unable to be matched (none were incorrectly classified). Simulations also suggest that a successful match can be made for viewing perspectives up to 30 degrees. At the time of

publishing, 111 pairs of previously unknown whale sharks were matched by the algorithm, of which 96 had strong scores.

2.4 Salmon Skin Pattern

The skin on the salmon is proposed to be unique by Eilertsen et al. [15]. This reasoning is based on the fact that there have previously been multiple studies on identifying animals based on their skin pattern, some examples of these are for whale sharks [16] and African penguins [18]. More instances of pattern matching and general use of biometrics in the animal kingdom is explored by a survey study [19].

2.5 Useful Computer Vision Methods

2.5.1 Image Moments

Overview

Image moments, as described by Hu [20], are weighted averages of image pixel intensities used to describe the spatial distribution of a set of points. Several different kinds of image moments can be used in computer vision, each with progressively more invariance.

These image moments are used to capture spatial information of blobs. A blob is an object within an image which a corner or edge detector has separated from other objects.

Raw Moments

Raw image moments, sometimes called spatial image moments contain information about the location and the shape of a blob. The formula for raw image moments can be written as:

$$m_{ij} = \sum_x \sum_y x^i y^j I(x,y)$$

where m is the resulting matrix, and I is a single channel binary image. I(x,y) is the intensity of the pixel position (x,y), which can be either 0 or 1 in the case of a binary image. Lastly, i

and j are integers that index the order of the moment being calculated. In the case of OpenCV moments [21] the following values (up to the third-order) are calculated for raw image moments: m00, m10, m01, m20, m11, m02, m30, m21, m12, m03.

Centroids

The center of mass of a blob can be found using the raw moments calculated in 2.5.1. This is essentially the blob location on an image. The centroid location $(\bar{x}, \bar{y})$ is given by the following formula.

$$\bar{x} = \frac{m_{10}}{m_{00}}$$
$$\bar{y} = \frac{m_{01}}{m_{00}}$$

Central Moments

Central moments is an image moment that is translation invariant. This means that the values provided by this formula will be the same for a shape no matter where it is found in the image. This is achieved simply by subtracting the centroid from the raw moment formula. The new formula can be written as follows:

$$mu_{ij} = \sum_{x}\sum_{y} (x - \bar{x})^{i} \left(y - \bar{y}\right)^{j} I(x, y)$$

The OpenCV libray [21] provides support for calculating the third order central moments: mu20, mu11, mu02, mu30, mu21, mu12, mu03.

Normalized Central Moments

Normalized central moments apply a normalization to the central moments calculated above 2.5.1. This provides invariance to scale. This means that if the same shape appears at different scales, the normalized central moments will give the same value.

$$\mathtt{nu}_{ji} = \frac{\mathtt{mu}_{ji}}{\mathtt{m}_{00}^{(i+j)/2+1}}.$$

The OpenCV libray [21] provides support for calculating the third order normalized central moments: nu20, nu11, nu02, nu30, nu21, nu12, nu03.

Hu Moments

Finally, Hu moments provide additional invariance. Hu moments are invariant to translation, scale, rotation, and reflection. There are seven moments calculated, each based on various combinations of normalized central image moments. An in-depth analysis of the properties and theory behind Hu moments can be found in a research paper by Zhihu Huang and Jinsong Leng [22].

$$hu[0] = nu_{20} + nu_{02}$$
$$hu[1] = (nu_{20} - nu_{02})^2 + 4nu_{11}^2$$
$$hu[2] = (nu_{30} - 3nu_{12})^2 + (3nu_{21} - nu_{03})^2$$
$$hu[3] = (nu_{30} + nu_{12})^2 + (nu_{21} + nu_{03})^2$$
$$hu[4] = (nu_{30} - 3nu_{12})(nu_{30} + nu_{12})[(nu_{30} + nu_{12})^2 - 3(nu_{21} + nu_{03})^2] + (3nu_{21} - nu_{03})(nu_{21} + nu_{03})[3(nu_{30} + nu_{12})^2 - (nu_{21} + nu_{03})^2]$$
$$hu[5] = (nu_{20} - nu_{02})[(nu_{30} + nu_{12})^2 - (nu_{21} + nu_{03})^2] + 4nu_{11}(nu_{30} + nu_{12})(nu_{21} + nu_{03})$$
$$hu[6] = (3nu_{21} - nu_{03})(nu_{21} + nu_{03})[3(nu_{30} + nu_{12})^2 - (nu_{21} + nu_{03})^2] - (nu_{30} - 3nu_{12})(nu_{21} + nu_{03})[3(nu_{30} + nu_{12})^2 - (nu_{21} + nu_{03})^2]$$

2.5.2 Binary Morphological Operations

In this section some binary morphological operations [23] are detailed. The following notation will be used: A is the binary image on which the operations are performed. B is the kernel (also called structuring element) that is used during the operation. E is euclidean space or an integer grid (such as an image).

Erosion

Erosion is the morphological operation of passing a kernel over an image to remove material from a binary image.

$$A \ominus B = \{z \in E | B_z \subseteq A\}$$

Dilation

Dilation is the morphological operation of passing a kernel over an image to add material to an image.

$$A \oplus B = \bigcup_{b \in B} A_b$$

Opening

An opening operation is a compound operation consisting of an erosion operation 2.5.2 followed by a dilation operation 2.5.2. This removes areas smaller than the kernel but leaves larger areas relatively unchanged.

$$A \oplus B = (A \ominus B) \oplus B$$

Comparison

A comparison of the different operations on an image can be found in Fig. 2.6.

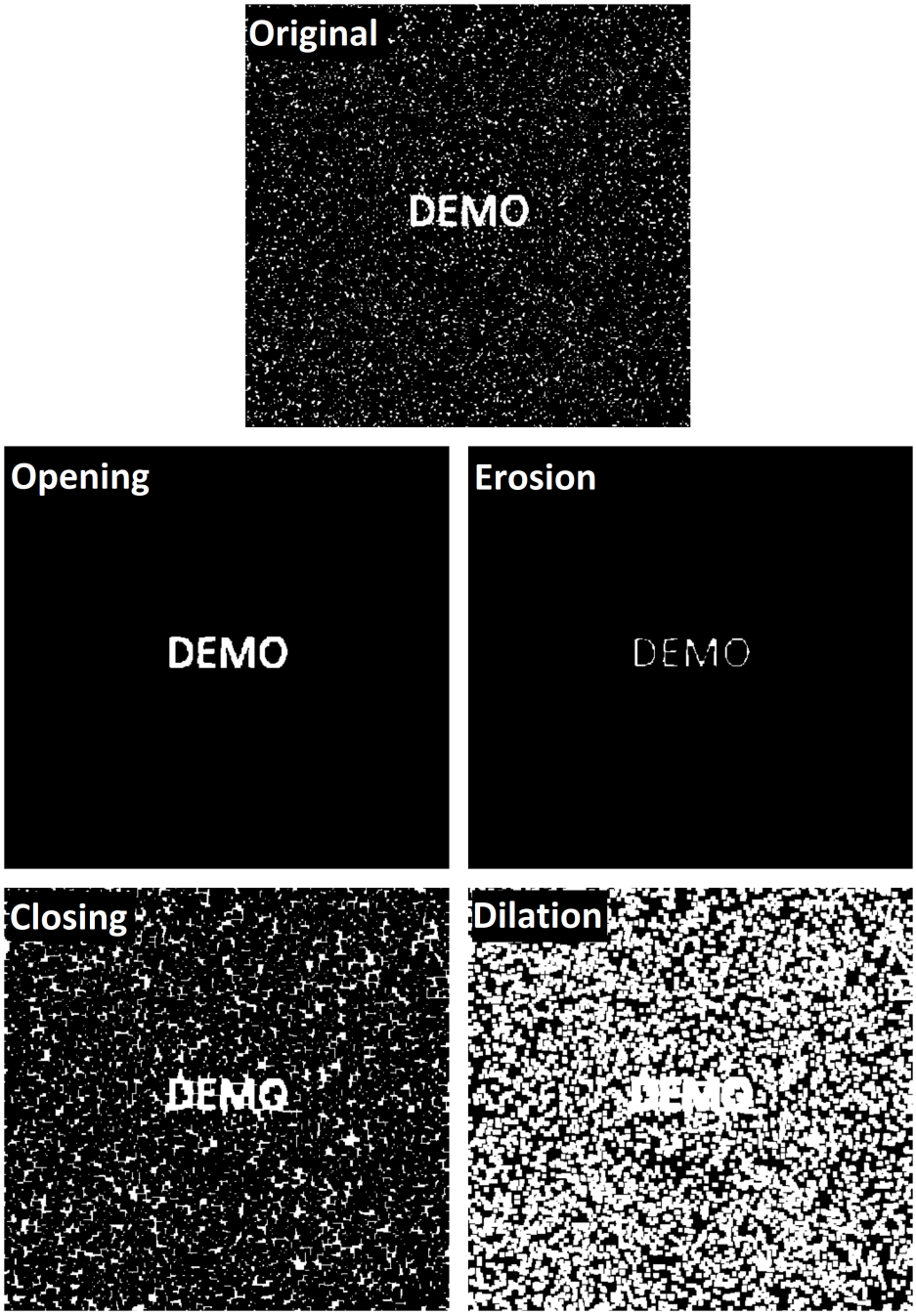


Figure 2.6: A comparison of the different morphological operations on an image. The original image can be seen on top and results of the operations on the image below.

2.5.3 Histograms of Oriented Gradients

Histograms of Oriented Gradients, also known as HOGs, were first proposed by Dalal and Triggs [24] for human detection. They have been widely used in computer vision for object detection, initially for detecting humans but more recently for detecting vehicle orientation [25] and even landmine detection in radar data [26].

For this thesis, the focus will be on the bucket and voting system utilized by the HOG algorithm. Essentially after calculating the gradients in an image, the resulting data consists of vectors with different orientations and magnitudes.

Once the orientations and magnitudes have been calculated, a bucket system is set up. The number of buckets can be changed. The orientation of the calculated vectors decides which bucket receives a vote, and the vote's strength is decided by the magnitude of the same vector. The number of buckets is usually set to 9, which means that a score is saved for every 20 degrees up to 180. An illustration of this can be found in Fig. 2.7.

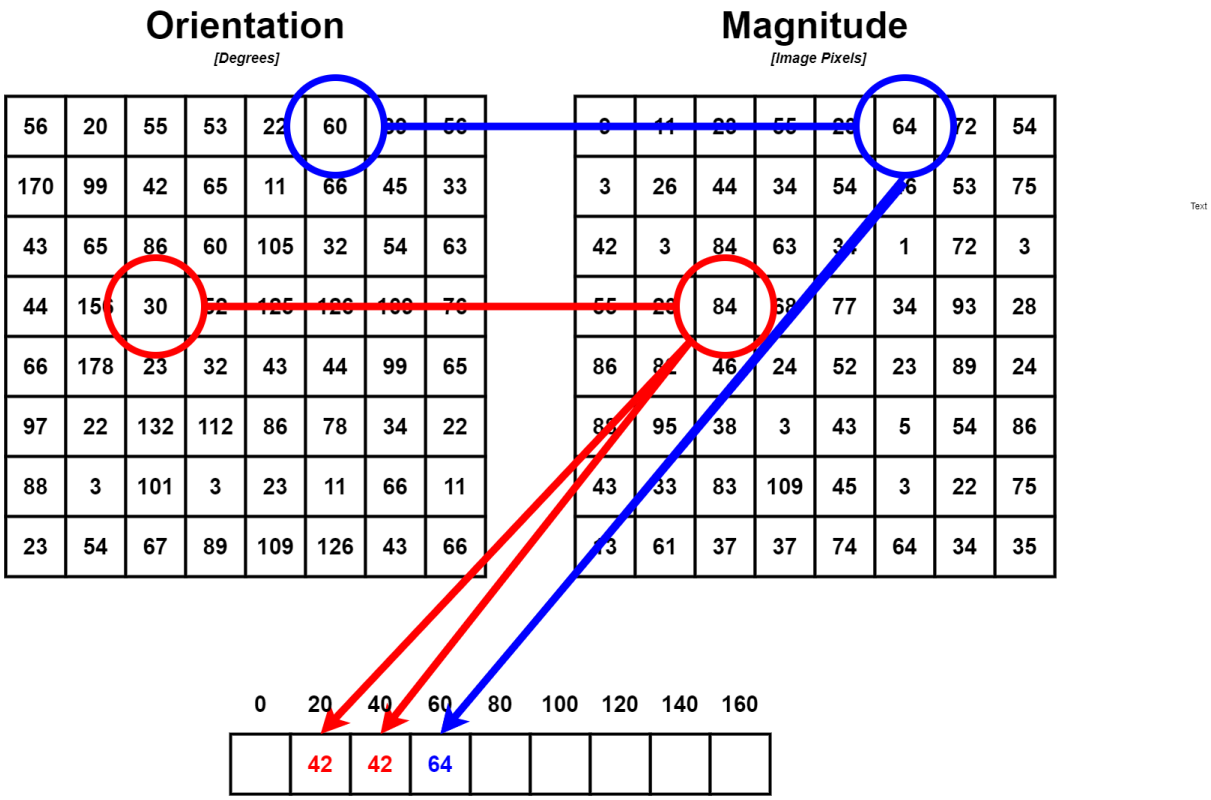


Figure 2.7: Figure illustrating the bucket voting system of HOG. Blue hits a bucket exactly and therefore the entire magnitude votes for the 60 bucket. The red orientation is split between two buckets, therefore the vote is divided.

2.6 Dataset

The original data for this research was generously provided by SINTEF and Eilertsen et al. [15]. The dataset consists of images that are 5472 pixels wide and 2000 pixels high.

A camera array consisting of three cameras took pictures of fish on a conveyor belt. This gives one identity from three slightly different perspectives. The images are marked with identity

and camera number. Though the dataset is not representative of underwater conditions, it provides an excellent baseline to create biometric algorithms. An example identity can be seen in Fig. 2.8.

Figure 2.8: An example identity from the SINTEF dataset. Notice how this is the same fish taken with three different cameras and angles.

These images have been processed by AI networks created in the Fall Project [14] to extract important biometrical features. These features create the basis for the data used in this research.

Throughout the thesis, the identity consisting of up to three images will be referred to as a fish's identity. Image will hereafter refer to a single image in an identity. This means that each identity contains up to three images, while each image only has one identity.

Chapter 3

Methods

3.1 Overview

In this chapter, the methods used to extract important biometrical information are defined, and the approach to constructing a matching algorithm is detailed.

A general overview of the data captured by the proposed approach as well as approaches discussed in the theory section (2.3.1, 2.3.2) can be seen in Tab. 3.1. A high-level overview of the method for extracting and matching the biometric information can be seen in Fig. 3.1.

Table 3.1: A table showing the blob information captured by the different algorithms.

	Position	Shape	Size	Spatial Relation
SalmID - Eilertsen et al. constellation algorithm [15]	YES	NO	NO	YES
Whale Shark Identification - Arzoumanian et al. [16]	YES	NO	NO	YES
Proposed	YES	YES	YES	NO

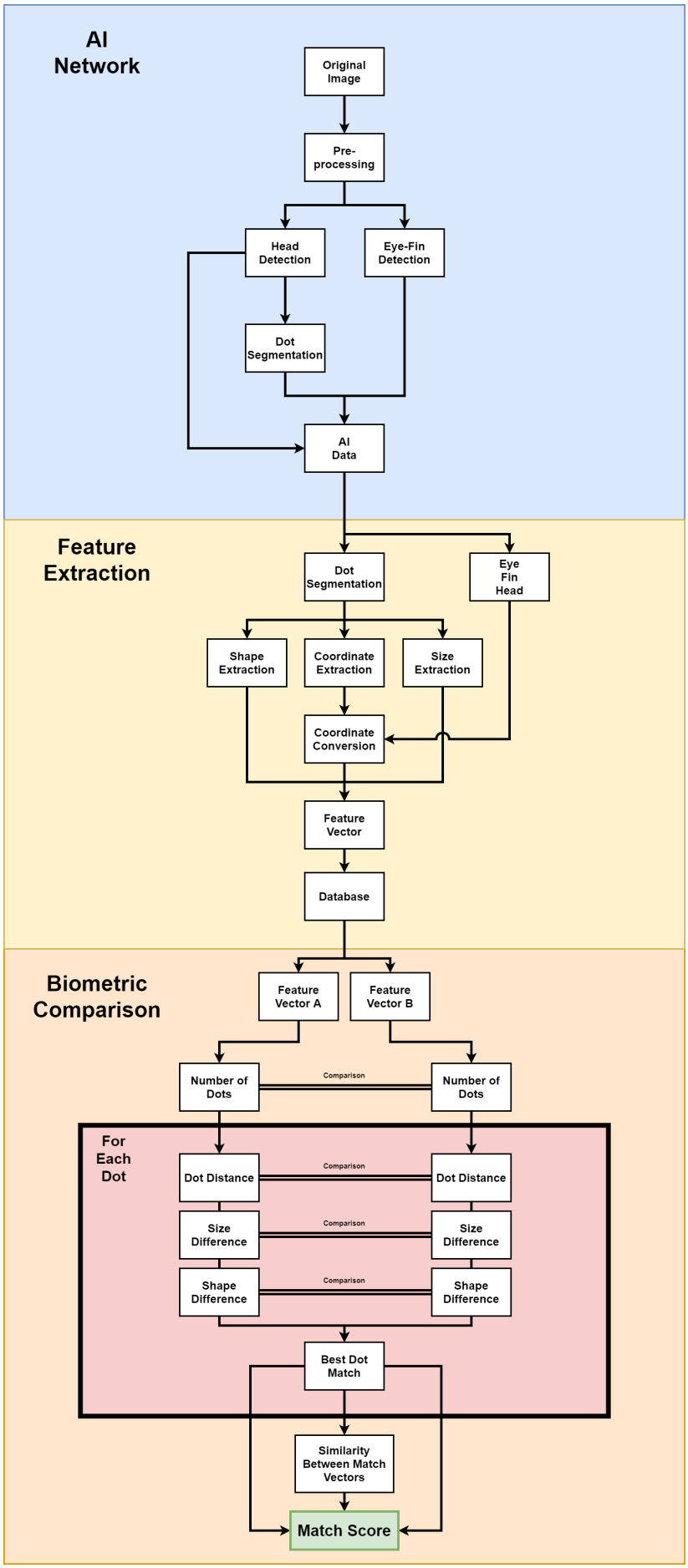


Figure 3.1: A high level overview of the methods described in this chapter. The AI Network part was completed during the fall project 2.2.

3.2 Extracting Spot Segmentation Data

3.2.1 Removing noise

Noise removal from images has long been a problem tackled by the visual computing community. This also means that there are many different approaches and ways to deal with the problem. In most cases, the resulting segmentation data from the AI network contains some noise, see Fig. 3.2.

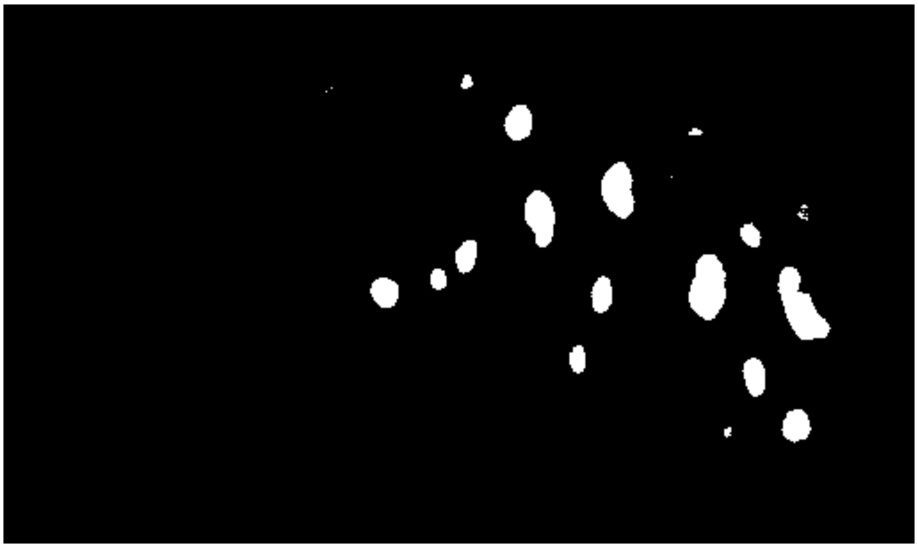


Figure 3.2: An image showing a segmentation before any post processing is done. Notice the small specks of noise.

To remove as much noise as possible, an opening operation performed. See theory section 2.5.2 for more details. This morphological operation allows for noise removal in binary images, such as those returned from the segmentation network. When using an opening operation, a fitting kernel size has to be utilized. A too-small kernel size will result in noise remaining in the image, and a too-large kernel size will remove actual spots from the image. A comparison of different kernel sizes can be found in Fig. 3.3.

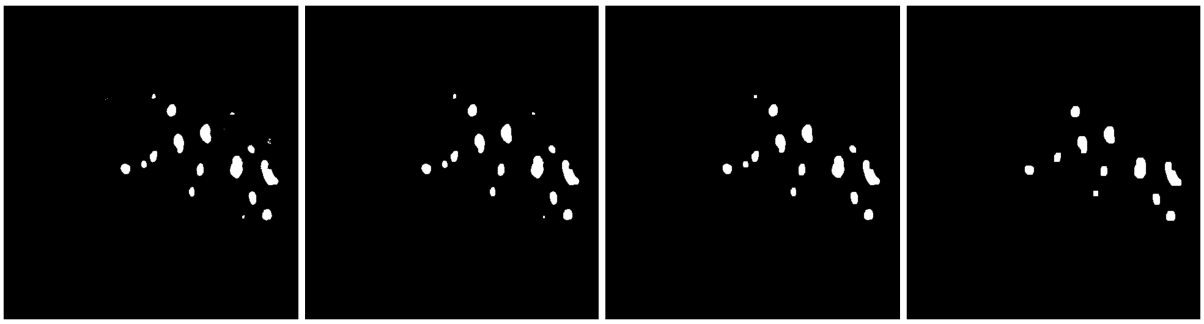


Figure 3.3: An image showing the effect of different kernel sizes on the pattern segmentation from the segmentation network. Left most image is the original image and then 3x3, 5x5 and 8x8 kernel size openings are shown respectively.

After running a few tests, the 3x3 kernel appears to provide the best results as it removes the worst of the noise but simultaneously keeps as much information as possible.

3.3 Extracting Spot Shape and Size

Image moments and hu moments, both originally proposed by Ming-Kuei Hu [20], are an excellent tool for capturing shape information. More details can be found in the theory section 2.5.1.

OpenCV has a built-in tool for image moments and hu moments making this easy to implement. There are seven invariants calculated in each of the image moments; these can be used to compare the spots' shapes. OpenCV allows computation of shapes mu 2.5.1, nu 2.5.1 and hu 2.5.1 values.

For every spot on a fish, the seven mu, nu, and hu moment invariants are then saved in the database 3.11. An example of a comparison of spot shapes using the different image moments can be found in Fig. 3.4.

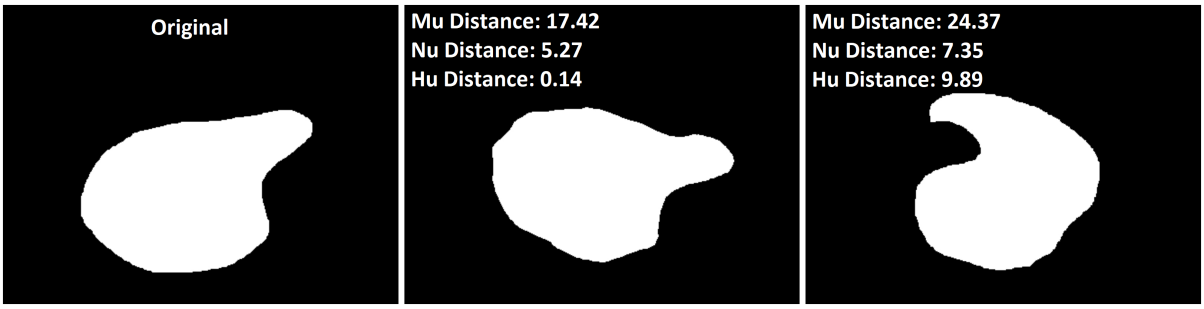


Figure 3.4: An illustration comparing some shapes. The left image is the original. The middle image is a slightly rotated and altered version of the original. Some distortion is very likely to happen on different images of the same spot. Right is the comparison to a completely different shape. Notice although the scales on the different shape metrics are different they all consider the middle shape a better match.

The shape matching is useful due to the large variance in spot shapes on the skin of the fish. Some examples of real spots can be found in Fig. 3.5.

Figure 3.5: Real spots returned from the segmentation network showing the large variance and unique shape of the spots.

3.3.1 Spot Coordinate Extraction

Image moments can be utilized to extract the center of mass of a blob 2.5.1. This results in a single XY coordinate for each blob, which is extracted alongside size and shape information.

3.3.2 Mathematical Representation

The first step to being able to match the spots on the fish is to represent the problem mathematically. After passing the images through the AI-networks, 4 points and an image is returned. The 4 points are the following: Eye coordinate, fin coordinate, XY-minimum of the

head bounding box, and XY-maximum of the head bounding box. The image returned is the segmentation of the spots on the head bounding box.

This segmentation data contains a lot of important features. Mainly the location of the spots but also size and shape.

The resulting data can be found in Figure 3.6. Note that shape information does not translate well to plots and, therefore, is missing in the figure.

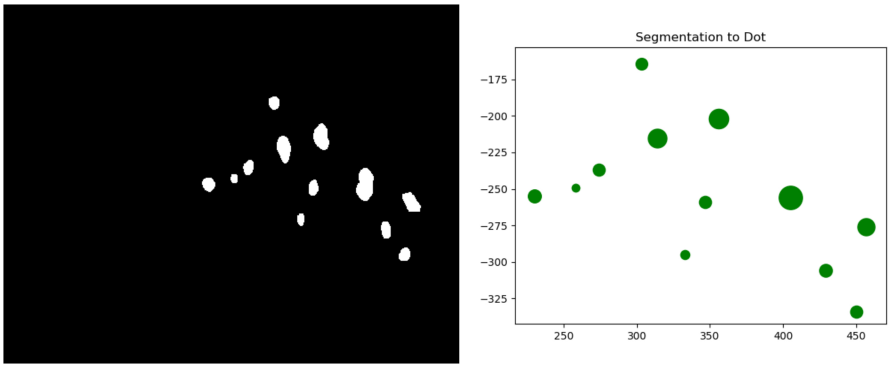


Figure 3.6: A comparison of the original segmentation from the AI-network and its corresponding mathematical representation. Shape data is not captured in plots. Note: Usually image coordinates start in the top left, but here the y-axis is flipped to better compare the images.

3.4 Coordinate Conversion

An essential step to comparing spots numerically is finding an invariant coordinate system for the spots. A unique coordinate system enables the spots to be matched even if the fish is at a different scale (further away or closer to the camera) or a different rotation. Translation invariance is provided by the head bounding box network.

This invariant coordinate system can be achieved in a two-dimensional system by finding two invariant points. In the fall project [14], an artificial intelligence network was created to locate the eye and fin of the fish, which serves as the two anchor points for this coordinate system. These anchor points are used to create two axes, the eye-fin line and its normal will be used with the center point of the eye as the origin.

To be able to convert the spots, a few transformations are necessary. All locations extracted by the artificial intelligence networks are given in image coordinates. The bounding box, eye, and fin are relative to the original image, while the spots are given in coordinates relative to

the bounding box. An illustration of this can be found in Fig. 3.7. A real example taken from the dataset with key points marked can be found in Fig. 3.8.

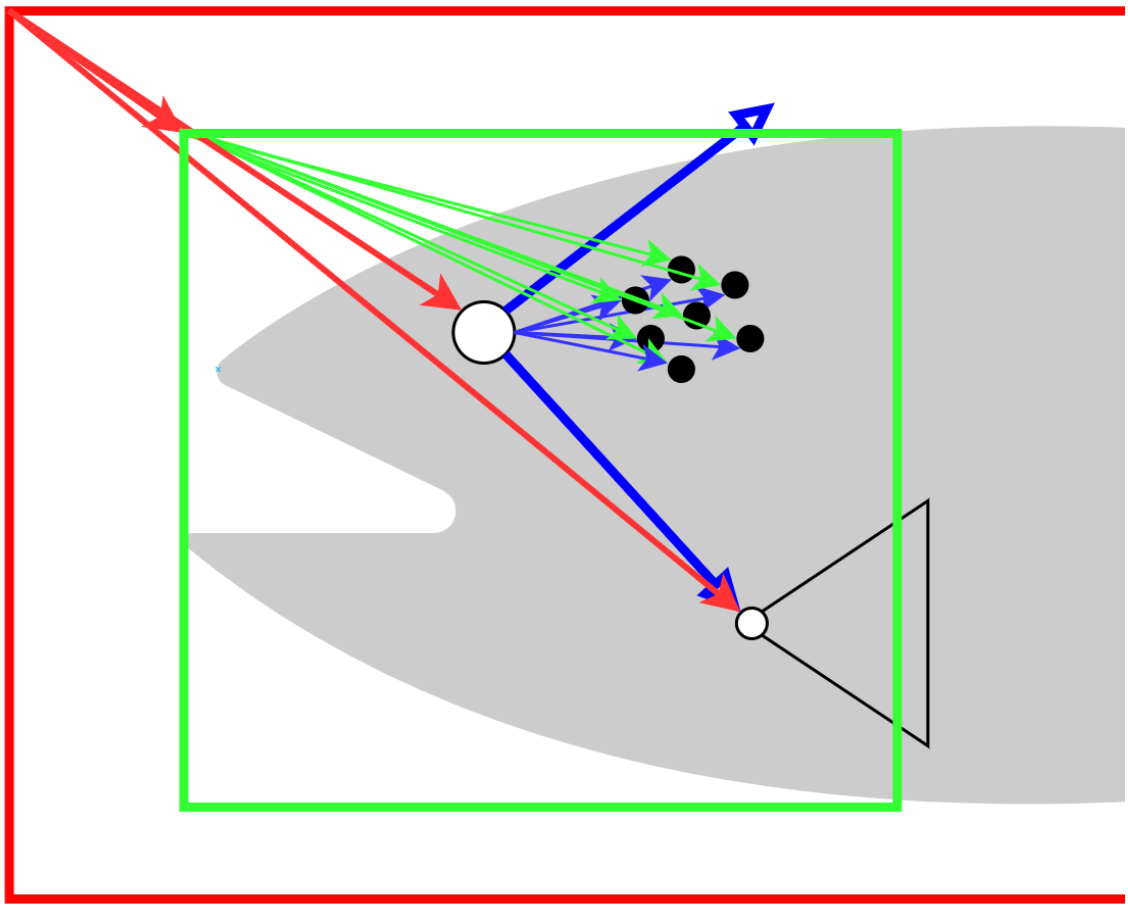


Figure 3.7: Figure showing the different coordinate systems resulting from the image processing. The red box is the original image. The coordinates of the head, eye, and fin are given relative to the original image's pixel values represented by the red vectors. The head extraction is represented by the green box. This image is given to the segmentation network to locate spots. The spots (in black) are therefore given in coordinates relative to the head extraction shown by the green vectors. The desired coordinate system is given by the line drawn from the eye to the fin points and the normal vector of this line. Desired spot coordinates are represented by the blue vectors.

Figure 3.8: An image taken from the dataset illustrating the concept from (Fig. 3.7.)

To convert points to the new coordinate system, three transformations have to be applied: translation, rotation, and scaling. The translation factor can be found by subtracting the eye location *(E)* from the origin of the coordinate system *(O)*. This centers the spots in the new coordinate system. The next step is to perform a rotation since the converted coordinate system is rotated relative to the image coordinate system. This can be done by finding the rotation factor *(r)* by finding the rotational angle between the old x-axis and the new coordinate system x-axis. The last step in the coordinate conversion is correcting the scale. This is done by measuring the magnitude of the eye-fin line and scaling it to a constant value. The difference between this constant value and the magnitude of the eye-fin vector will be the scaling factor *(s)*. These factors are then represented in matrix form 3.1 and multiplied to create the final transformation matrix 3.2.

$$\mathbf{T} = \begin{pmatrix} 1 & 0 & O_x - E_x \\ 0 & 1 & O_y - E_y \\ 0 & 0 & 1 \end{pmatrix} \quad \mathbf{R} = \begin{pmatrix} \cos(r) & -\sin(r) & 0 \\ \sin(r) & \cos(r) & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \mathbf{S} = \begin{pmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.1}$$

$$M = S \cdot R \cdot T \tag{3.2}$$

The result of the coordinate transformation can be seen in Figures [3.9, 3.10].

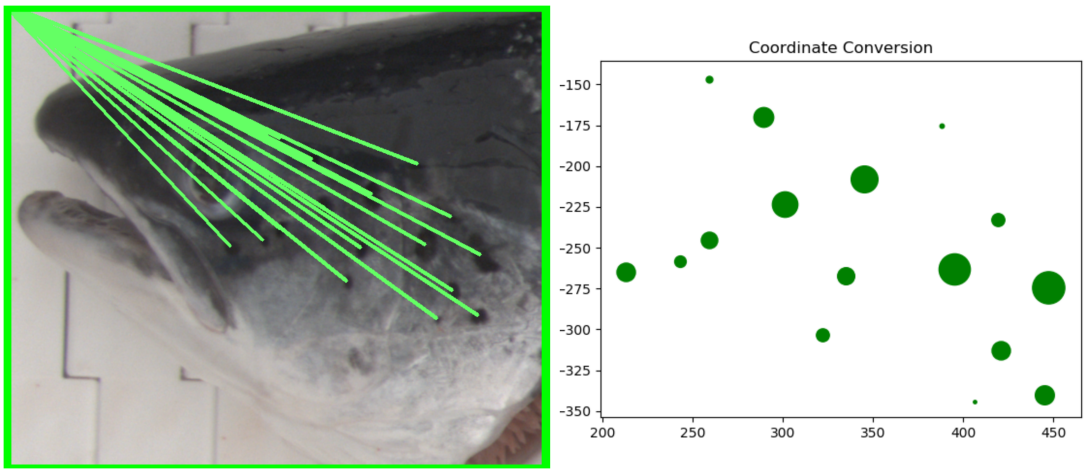


Figure 3.9: A side by side comparison of **original** spot coordinates. Note: Usually image coordinates start in the top left, but here the y-axis is flipped to better compare the images.

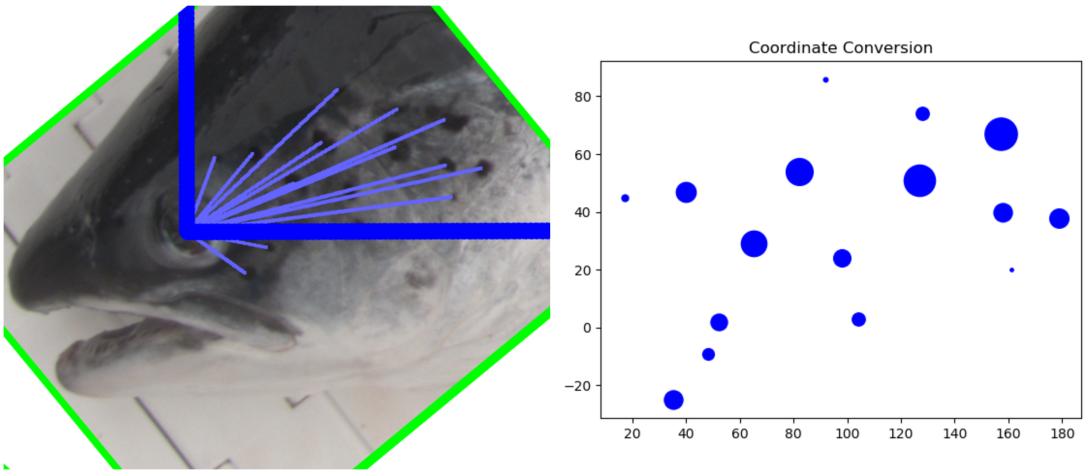


Figure 3.10: A side by side comparison of **converted** spot coordinates. Origin is the centre of the large vectors. Note: The left image is rotated to better compare the images.

3.4.1 Database

The extracted data is saved in a database before matching. This enables rapid extraction and efficient storage. By storing the data in the database, experiments can be performed on numerical data without the need to pass an entire image through the AI network. The data and structure of the database can be seen in Fig. 3.11.

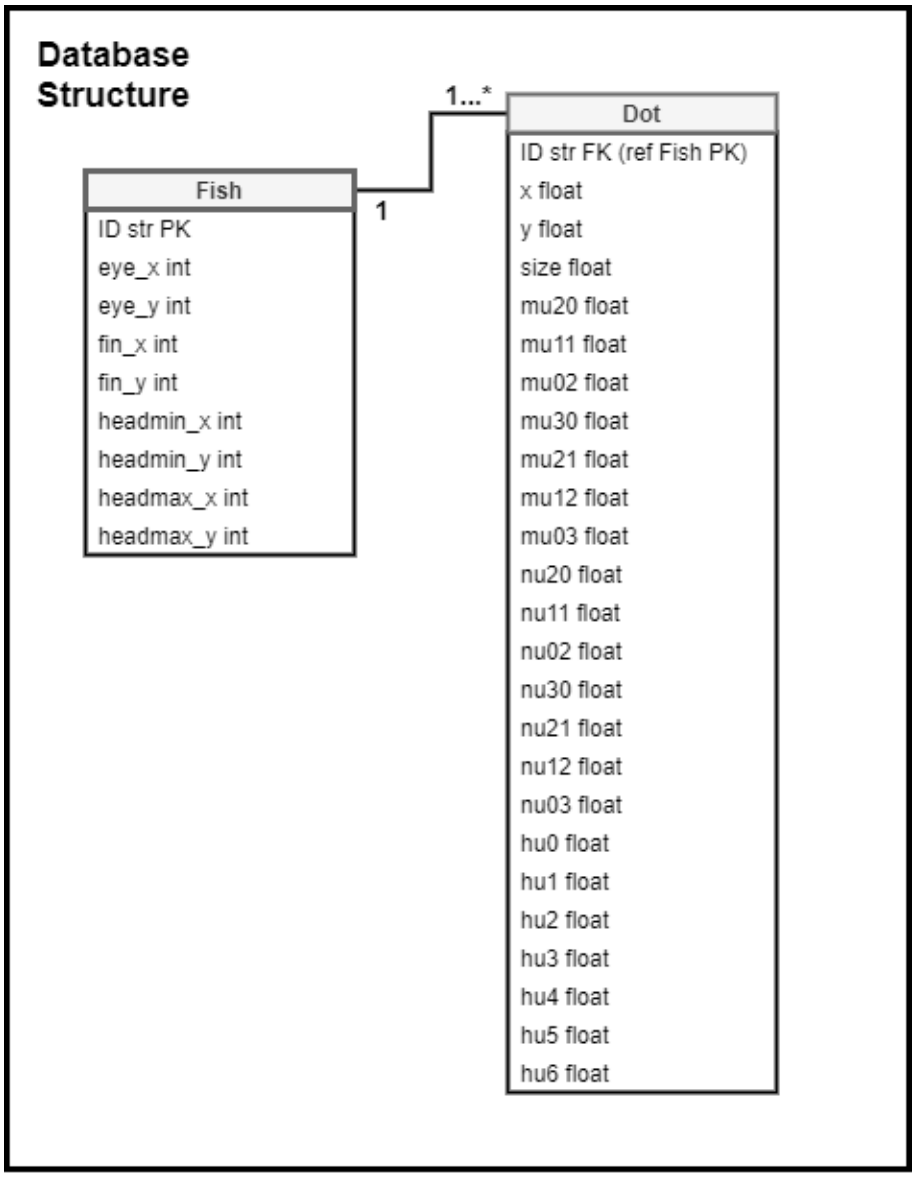


Figure 3.11: An overview of what is contained in the database. The basic relationship is that each fish can have one to many spots but each spot can only belong to one fish.

3.5 Algorithm Construction

3.5.1 Overview

In this section, the process of constructing the matching algorithm is detailed. The algorithm consists of three main components: Number of spots score, spot match score, and vector similarity score.

3.5.2 Identity Distance based on Spot Properties

Number of Spots

Before any spot comparisons are made, a simple comparison of the number of spots is made. The number of spots on an image is compared to its potential match image, and the difference in spots is then squared to punish large differences more severely. Lastly, this value is multiplied by a constant c0. The following score is computed:

$$(abs(a) - abs(b))^2 * c_0$$

where a is the number of spots in identity A, and b is the number of spots in image B. Pseudocode for this operation can be found in Listing 3.1.

Listing 3.1: Spot Matching

```
total_score = 0

missing_spot_score = square(abs(n spots A - n spots B)) * c0

total_score += missing_spot_score
```

Spot Scoring

For the spot scoring part of the algorithm, only the scores between the best matches are saved. The algorithm is based around weighted sums of the distances between the features. The following formula is used.

$$\sum_{i=1}^{a} \forall_{j=1}^{b} max(min((\Delta d_{ij} \cdot c_1 + \Delta S_{ij} \cdot c_2 + \Delta \mu_{ij} \cdot c_3 + \Delta \nu_{ij} \cdot c_4 + + \Delta h_{ij} \cdot c_5) \cdot \frac{c_6}{S_i}, c_7)$$

where a is the number of spots in the identity A and b is the number of spots in the identity B. i and j are spots from their respective identity. $\Delta_{d_{ij}}$ is the distance between spot i and j. $\Delta_{S_{ij}}$ is the difference in size between spot i and j. $\Delta_{\mu_{ij}}$ is the difference in mu value 2.5.1

between spot i and j. $\Delta_{v_{ij}}$ is the difference in nu value 2.5.1 between spot i and j. $\Delta_{h_{ij}}$ is the difference in hu value. The best constant values found during tuning are described in section 3.5.5.

Additionally, the entire score for a spot is weighed based on the size of the spot being matched. The reason for this being that the segmentation network is more accurate in size, shape, and even location on larger spots. Pseudocode for what this looks like can be found in Listing 3.2.

Listing 3.2: Spot Matching

```
for A_spot in identity_A
    best_match = None
    best_score = 99999999
    for B_spot in identity_B

        size_diff = abs(A_spot["size"] - B_spot["size"])

        distance = abs(A_spot["x"] - B_spot["x"]) +
            abs(A_spot["y"] - B_spot["y"])

        hu_diff = abs(A_spot["hu"] - B_spot["hu"])
        mu_diff = abs(A_spot["mu"] - B_spot["mu"])
        nu_diff = abs(A_spot["nu"] - B_spot["nu"])

        score = (size_diff * c1) +
            (distance * c2) +
            (mu_diff * c3) +
            (nu_diff * c4) +
            (hu_diff * c5)

        # The larger the spot the larger the impact on the score
        # (Remember lower score is better)
        score = score * (c6 / A_spot["size"])
```

```
        if score < best_score
            best_score = score
            best_match = B_spot

    # Best score is capped to prevent large outliers
    if best_score > constant
        best_score = constant

    total_score += best_score
```

3.5.3 The Best Spot Match

For each spot, its best match is found based on the above calculations of size, shape, and location. It is important to note that each spot is matched to its lowest score in this algorithm, but this spot is not removed from the possible matches in the future. This essentially means that each spot has one and only one best match, but can be the best match of multiple spots. A figure showing a spot's best matches can be seen in Fig. 3.12

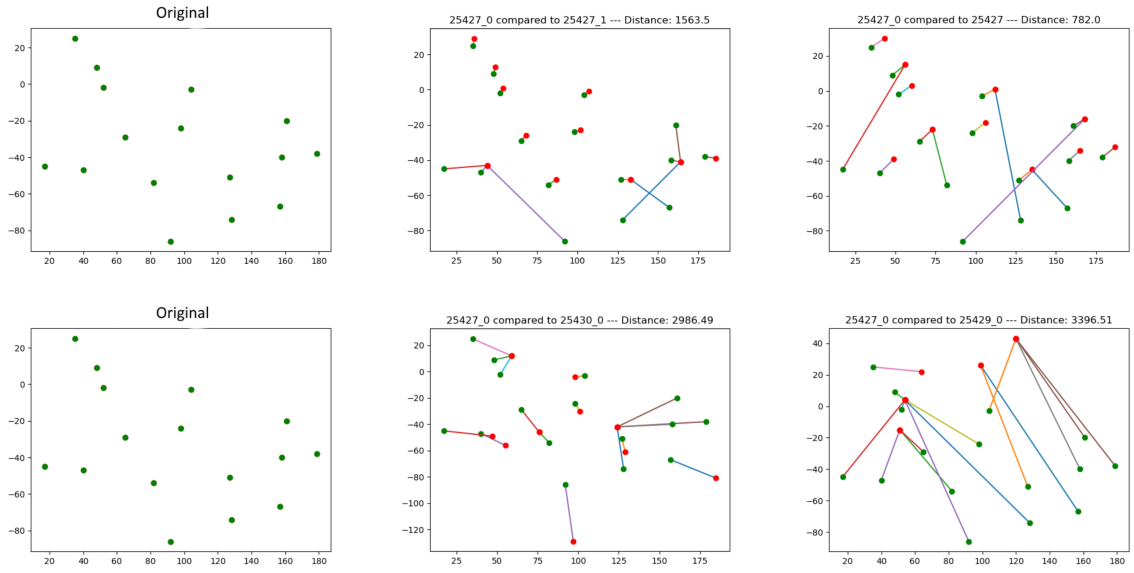


Figure 3.12: The top three images show an identities spots (green) matched to spots from the other two images (red) of the same identity. The bottom three images show the same original identity as above but in this instance matched to two images from different identities. The lines between spots indicate a best match relation.

Note that the best match vectors (line indicating the best match relation between two spots of different images 3.12) have some interesting properties. If the matches are true, they have small magnitudes and similar directions, and in the case that they are false matches, the vectors are large and random. These properties create the basis for the vector similarity part of the algorithm.

3.5.4 Vector Similarity

Once the best spot match has been decided for every spot in the potential match identity, the similarity between the match vectors is examined. This is based on properties noticed in section 3.5.3 where true matches have similar best match vectors, while false matches more random vectors.

Statistical Methods

To exploit the difference in vector properties observed, simple statistical methods were first utilized.

The average of the vector magnitudes was investigated. Based on observations, a true match would have a lower magnitude average than a false match. This did not yield significant results.

The standard deviation of both vector magnitude and angle were investigated as well. Observations showed that vectors in true matches might not necessarily be small, but the vectors were similar in both magnitude and angle. Comparing the standard deviation of the magnitude and angle did improve the match score but was later replaced by the bucket score system 3.5.4.

Bucket Scores

Since the more basic statistical methods yielded little results, a more robust system was attempted. To utilize the vector similarity phenomenon, a bucket system was set up similar to HOG 2.5.3. The bucket system was modified to favor vectors with a small magnitude. This allows for finding instances where similar directional vectors with small magnitudes such as those appearing in true matches. The modification made was essentially only subtracting the magnitude from a constant value, thereby vectors with smaller magnitudes receive

higher scores. An illustration of this can be found in Fig. 3.13.

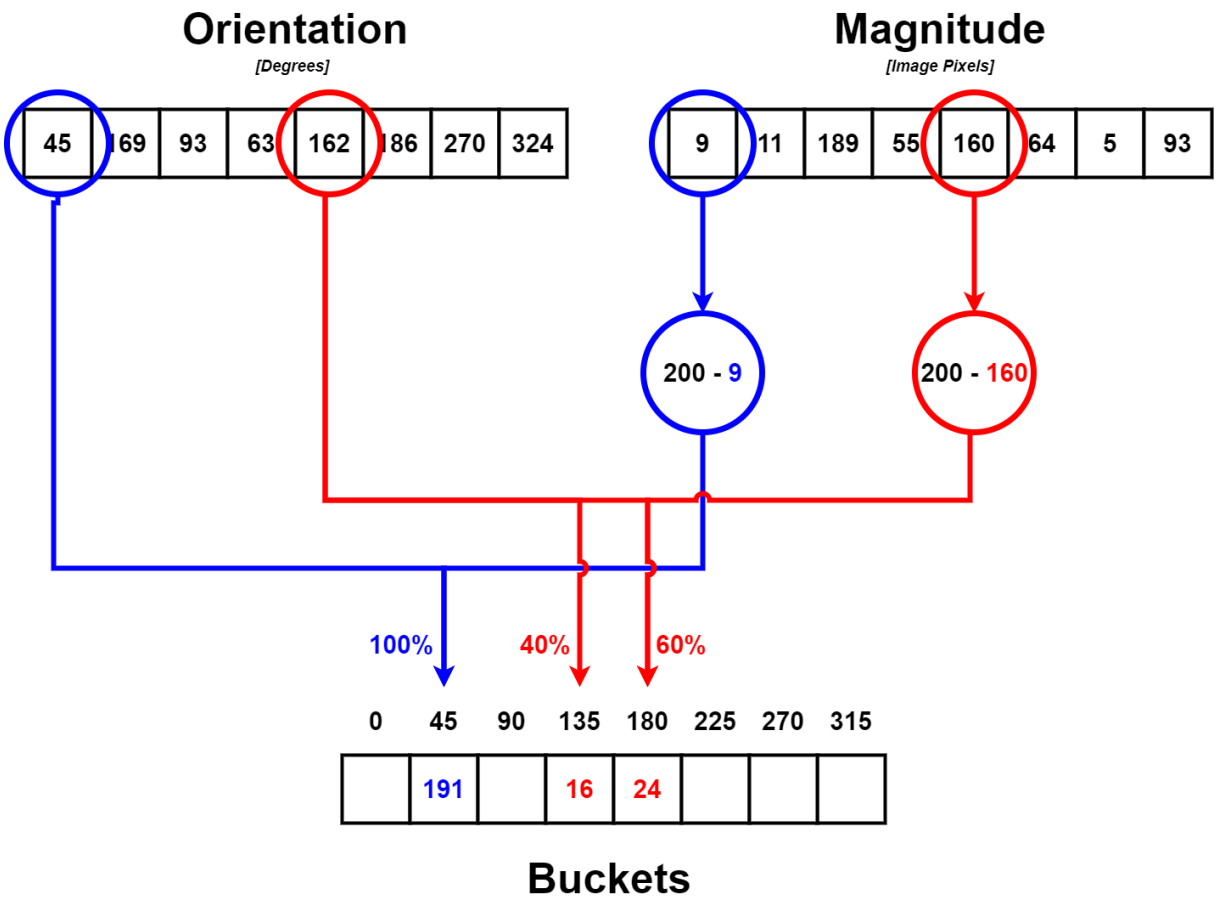


Figure 3.13: An illustration showing the modified HOG algorithm voting on buckets. Note that 360 degrees is equal to 0 degrees.

Empirically it seems that the magnitude does not matter as much as the angle. A modified version that only counts the number of occurrences in a bucket instead of magnitude performs similarly. Pseudocode for the bucket algorithm with constants b1 and b2 can be found in Listing 3.3.

Listing 3.3: Spot Matching

```
reference_vector = [0, 1]
number_of_buckets = b1

# Instatiate array with 0s
bucket_score = zeros(number_of_buckets)
```

```
for vector in vector_list
    angle = get_angle_between(vector, reference_vector)
    magnitude = norm(vector)

    bucket = (angle / (360 / number_of_buckets))
    decimal = bucket % 1

    lower_bucket_score = (1 - decimal) * (b2 - magnitude)
    higher_bucket_score = (decimal) * (b2 - magnitude)

    bucket_score[int(bucket)] += lower_bucket_score
    bucket_score[(int(bucket) + 1) % number_of_buckets] +=
        higher_bucket_score
```

Summary

The three main parts of the algorithm are:

Number of Spots Score: (3.3)

Spot Matching Score: (3.4)

Vector Similarity Score: (3.5)

$$((abs(o) - abs(p))^2 * c_0) \tag{3.3}$$

$$(\sum_{i=1}^{o} \forall_{j=1}^{p} max(min((\Delta d_{ij} \cdot c_1 + \Delta S_{ij} \cdot c_2 + \Delta \mu_{ij} \cdot c_3 + \Delta v_{ij} \cdot c_4) \cdot \frac{c_5}{S_i}), c_7)) \tag{3.4}$$

$$(std(magnitudes(V)) * c_8 - bScore() * c_9) \tag{3.5}$$

All constant values for the algorithm can be found in Table 3.5.5.

3.5.5 Weight Tuning

Algorithm

All the algorithm's components are tuned by multiplying a constant value and finding which constant yields the best match rate. Tuning is very time consuming as multiple runs through the entire dataset have to be performed for finding a single constant value. However, this is essential to the performance of the algorithm. The reason this is so important is that the values from the components come in a large scale variance, components would disproportionately affect the algorithm if they did not contain a constant value as a weight. Some components benefit the algorithm more than others and, therefore, might need to be valued higher. The constant values essentially work as weights for the algorithm components. A figure showing the progression of performance with different constant values can be seen in Fig. 3.14. All constants were tuned sequentially.

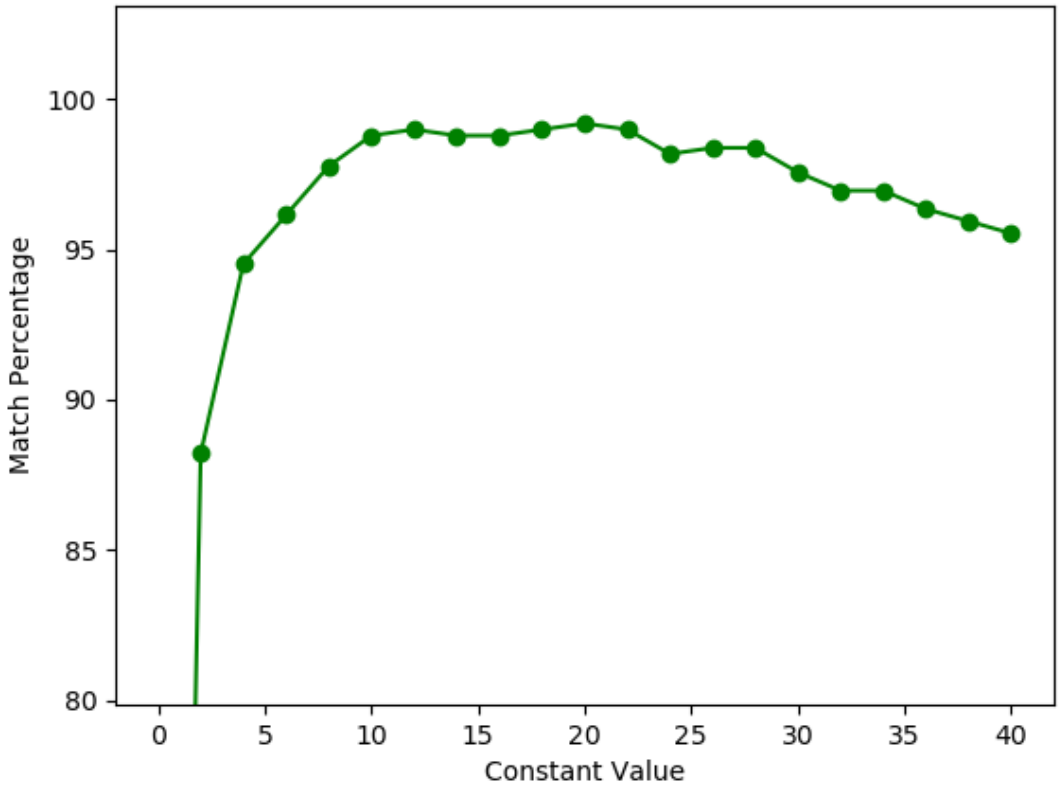


Figure 3.14: Different c1 constant values for an experiment run with 200 identities in the database.

In the case above, the best constant value was 20. Once this value has been selected, the next

constant value can be tuned.

Constants

After much tuning, the constant values that performed the best can be found in Table 3.5.5. Note that the constant values do not indicate how large a role they play in the algorithm; for this, a better comparison is made in 4.4. The constant values are a result of the best performance from tuning 3.5.5.

Table 3.2: Constant Values

Description	**Constant**	**Value**
Number of spots constant	**c0**	4
Distance constant	**c1**	10
Size constant	**c2**	1.5
Mu shape constant	**c3**	0.02
Nu shape constant	**c4**	1000
Hu shape constant	**c5**	500
Size division constant	**c6**	200
Score cap constant	**c7**	420
Number of buckets in bucket scoring	**b1**	8
Magnitude constant in bucket scoring	**b2**	200

Chapter 4

Evaluation

4.1 Overview

In this chapter, the results are presented. Firstly match definitions, and essential metrics are established 4.1.1. Two experiments were conducted in this thesis:

In the first experiment 4.2 a database containing 395 unique identities and 931 images was investigated. Due to this datasets' smaller size, it was also utilized for tuning the constants found in section 3.5.5. A true positive rate of 98.28% was achieved. Additionally this dataset was used to investigate the underlying mechanics of the algorithm 4.4, establish a confidence rating system [4.2.2, 4.3] and investigate imbalances in the dataset [4.4.2, 4.4.1].

The second experiment 4.5 was conducted on the largest database of individuals matched to date: 1000 identities and 2998 images. This experiment used the same weights as experiment 1. A true positive rate of 96.82% was achieved, and a precision-recall plot 4.16 was created using the confidence rating system established.

Lastly, the decline in performance as more identities were added to the database was investigated 4.6.

4.1.1 Match Definitions

True Positive

A match is considered a **true positive (TP)** when an image A is compared to the database, and the best match (image with the lowest distance score), image B, belongs to the **same** identity.

False Positive

A match is considered a **false positive (FP)** when an image A is compared to the database, and the best match, image B, **does not** belong to the same identity.

False Negative

A match is considered a **false negative (FN)** when an image A is compared to the database, and the best match, image B, belongs to the **same** identity, but the confidence rating was too low, and the match was discarded incorrectly.

True Negative

A match is considered **true negative (TN)** when an image A is compared to the database, and the best match, image B, **does not** belong to the same identity, and the confidence rating was too low; therefore the match was discarded correctly.

4.1.2 Precision and Recall

Precision: Eq 4.1.

Recall: Eq 4.2.

$$precision = \frac{TP}{TP + FP} \tag{4.1}$$

$$recall = \frac{TP}{TP + FN} \tag{4.2}$$

4.1.3 Match Score

Match Score is the result of comparing two images using the algorithm 3.5. Since the algorithm measures distance, a lower score is better.

4.1.4 Match Score Difference

Match score difference is used as a confidence measure based on the difference between expected correct and expected incorrect matches. If an identity contains two images, a single image is expected to be correct; the match score difference is defined as the distance between the image with the lowest distance (expected correct) and the second-lowest score (expected incorrect). In the case that an identity contains three images, two images are expected to be correct; the match score difference is then defined to be the distance between the image with the second-lowest score (expected correct) and the third-lowest score (expected incorrect).

4.1.5 Proportional Match Score Difference

Proportional match score difference is a modified version of match score difference 4.1.4 that looks at the proportional difference instead of pure score difference. Proportional match score difference was calculated using the formula below: 4.3.

$$PMSD = \frac{MSD(A,B)}{MS(B)} \tag{4.3}$$

where A is the expected correct image, and B is the expected incorrect image. MSD(A, B) is the match score distance of images A and B. MS(B) is the match score of image B. PMSD is the proportional match score distance.

Proportional match score difference is defined for values 0 up to 1. A higher PMSD indicates higher confidence.

4.2 Experiment 1

4.2.1 Results

Using the algorithm developed (see section 3.5) and with the constant values from section 3.5.5 an experiment was set up to test the matching performance and to get a sense of how the algorithm performs with different numbers of identities. A database containing 395 identities and a total of 931 fish was used. The experiment started with 20 identities matched by the algorithm before progressively adding 20 more for each iteration. The result of this experiment can be seen in Fig. 4.1. The performance on the entire database containing 395 identities and a total of 931 fish was a true positive rate of 98.28%. Of the 931 images, only 16 images were false positives.

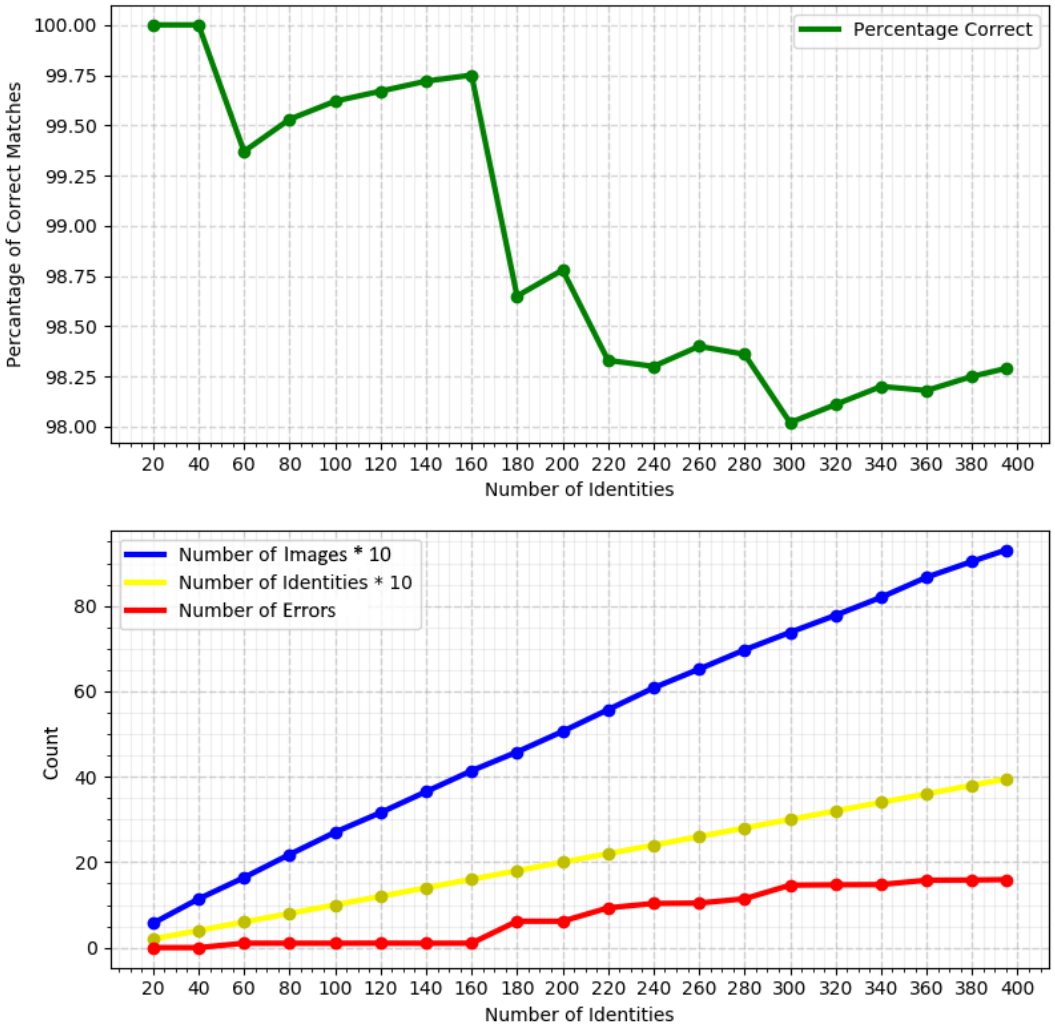


Figure 4.1: Match percentage as progressively more identities and images are added to the database. Notice the scale on images and identities should be scaled by 10, but not the number of mistakes.

4.2.2 Match Comparison

In this subsection, the 20 best matches for an image are presented in bar diagrams. The x-axis shows identity, and the y-axis shows the match score 4.1.3. Note that a lower score is better as it measures the distance from the identity.

in Fig. 4.2, the match was considered a true positive as the image with the lowest score is from the same identity. The distance between the correct images and incorrect images called

match score difference 4.1.4, can be seen as an indicator of how confident the algorithm is. If this gap is large, then the algorithm is essentially more sure of the match. The match score difference is marked in red on the figure 4.2.

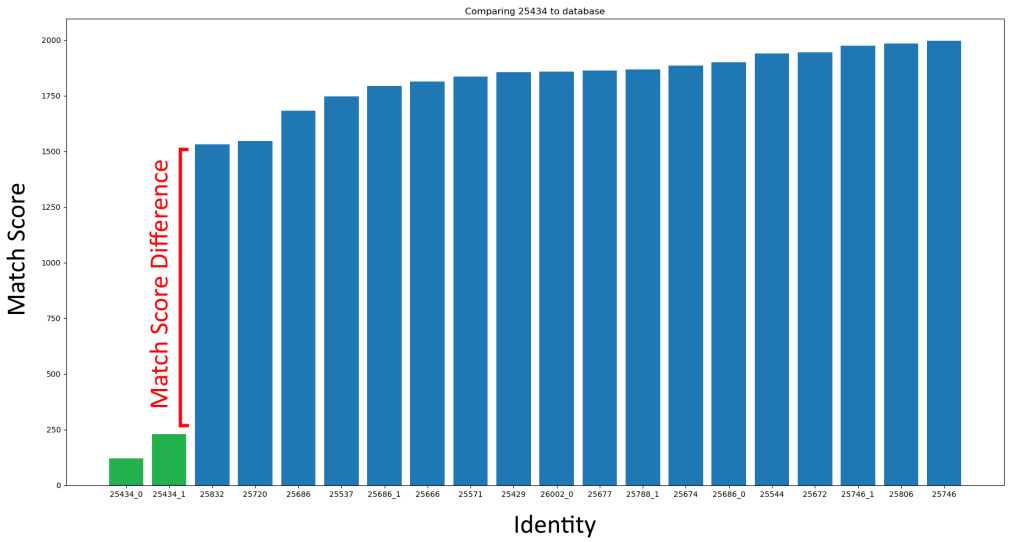


Figure 4.2: Results showing a strong match. Notice the difference in distance between the correct matches and the incorrect matches marked in red called confidence.
Red: Match Score Difference. 4.1.4
Green: Green bars represent the images from the same identity.
Blue: Blue bars represent images from other identities.

in Fig. 4.3 the match was also considered a true positive. However, in this case, the match is weaker than the previous figure given that the match score difference (distance between the second and third match) is lower.

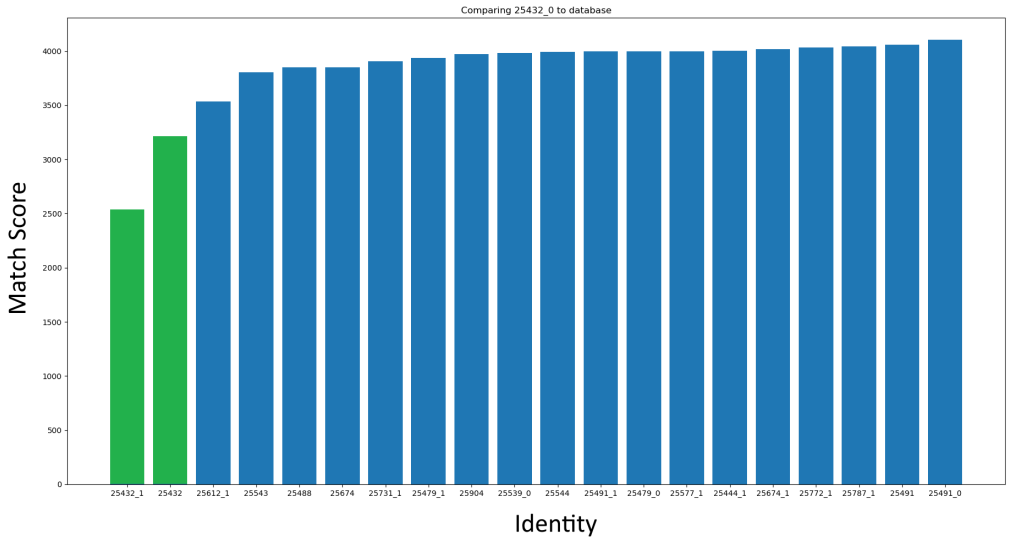


Figure 4.3: Results showing a weaker match. Notice how there is a lower distance between the correct and incorrect matches.
Green: Green bars represent the images from the same identity.
Blue: Blue bars represent images from other identities.

Fig. 4.4 is an example of a false positive image. The actual true matches are close to being correct but are not the images with the lowest scores. Interestingly the match score difference (distance between second- and third-best match) is fairly low.

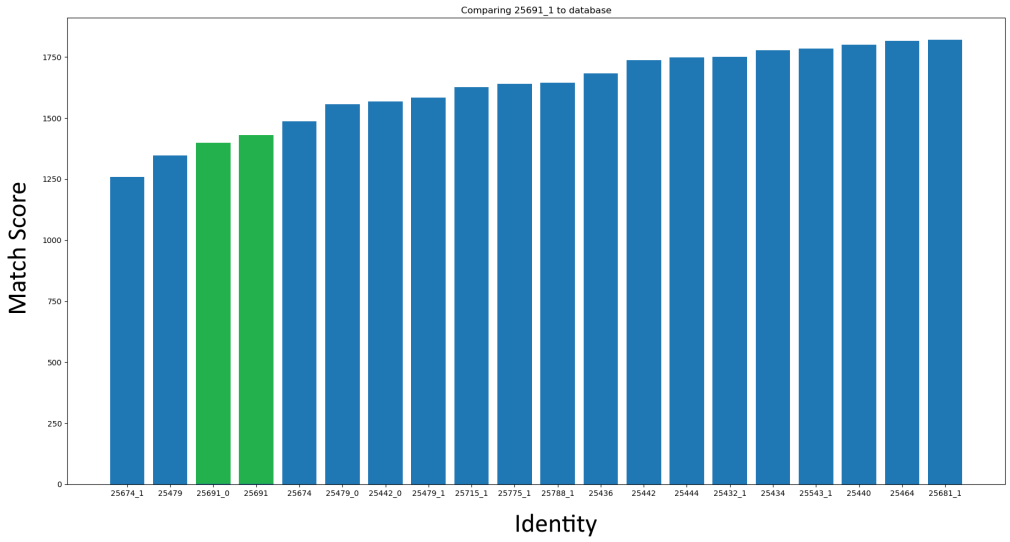


Figure 4.4: One of the mismatched identities showing that the actual correct matches were very close.
Green: Green bars represent the images from the same identity. **Blue**: Blue bars represent images from other identities.

Another example of a false positive image can be seen in Fig. 4.5. In this case, the correct images were nowhere within the 20 best-scored images. However, it is interesting to note the homogeneous scores and low match score difference.

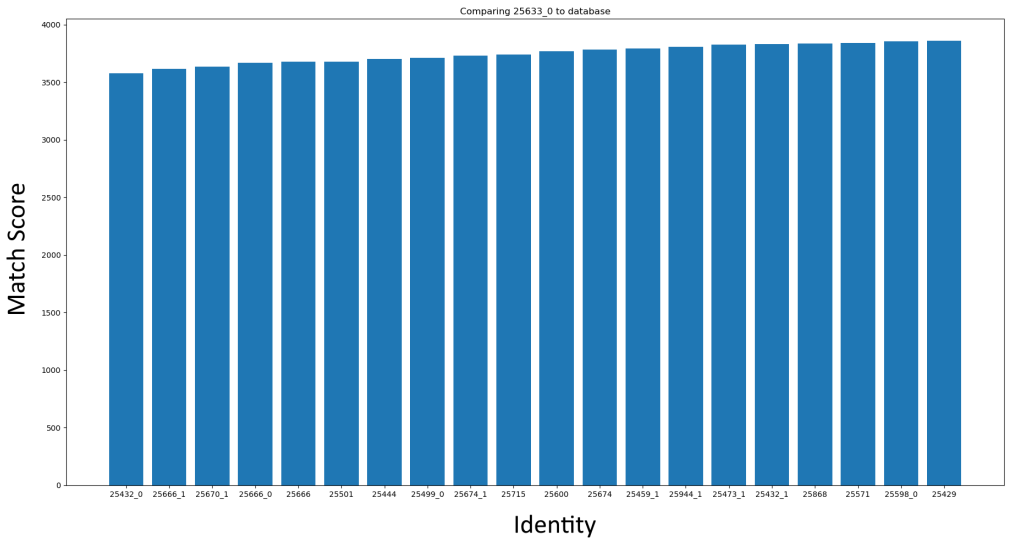


Figure 4.5: One of the mismatched identities showing that no correct identities were close. Interestingly all the scores are very similar.
Blue: Blue bars represent images from other identities.

More graphs can be found in Appendix A.1.

4.3 Confidence Rating

Based on the observations in the match comparison section 4.2.2, a confidence rating system was created based on the match score difference 4.1.4. Each match receives a confidence rating based on the distance between the expected match and the expected non-match (match score distance). An illustration containing match score difference can be seen in Fig. 4.2.

The match score difference is the basis for the confidence rating. A graph showing the confidence rating based on match score difference can be seen in Fig. 4.6.

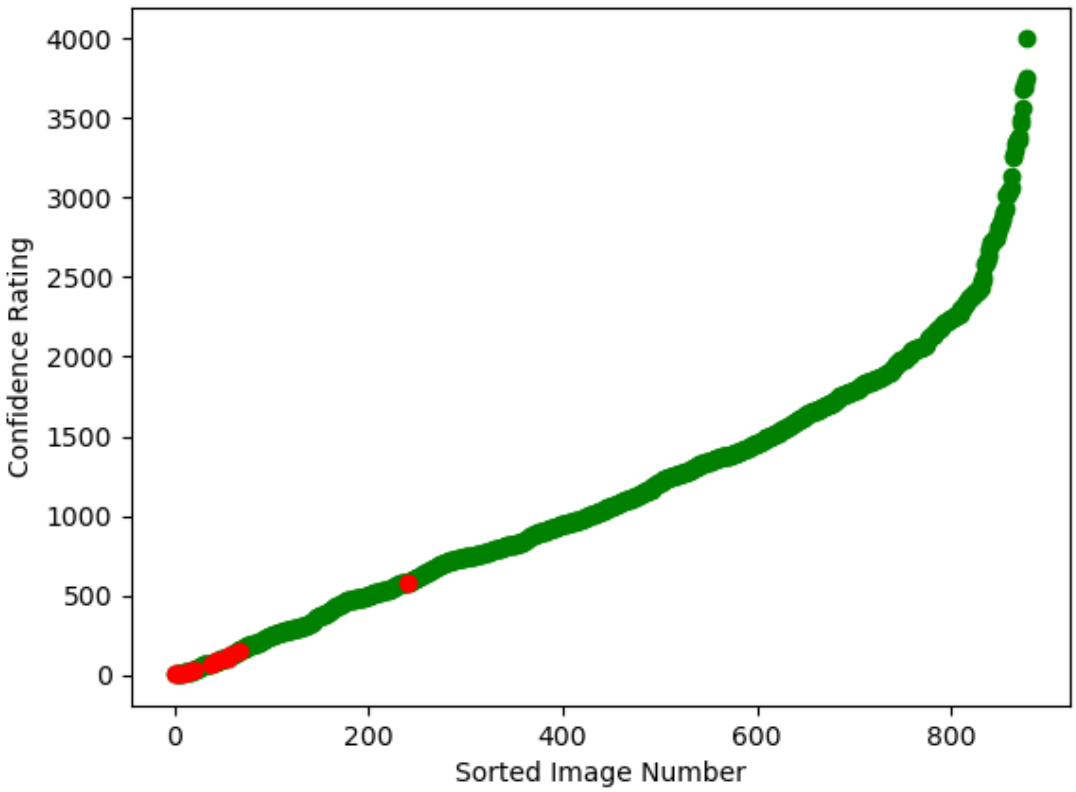


Figure 4.6: A graph showing match score difference as confidence rating. A higher rating is better.
Green Dot: Correctly matched image.
Red Dot: Incorrectly matched image.

An issue with using the match score difference as a confidence rating is that the match score is affected by the number of spots. Images with many spots naturally receive higher scores across the board due to more spot comparison scores. Therefore, an alternate and possibly better way to display confidence rating is to take the difference and divide it by the first expected incorrect match (either the second- or third-best match depending on the number of images in the identity). This metric looks at the proportional difference between match scores and is defined as the proportional match score difference (PMSD) 4.3. The resulting graph can be seen in Fig. 4.7.

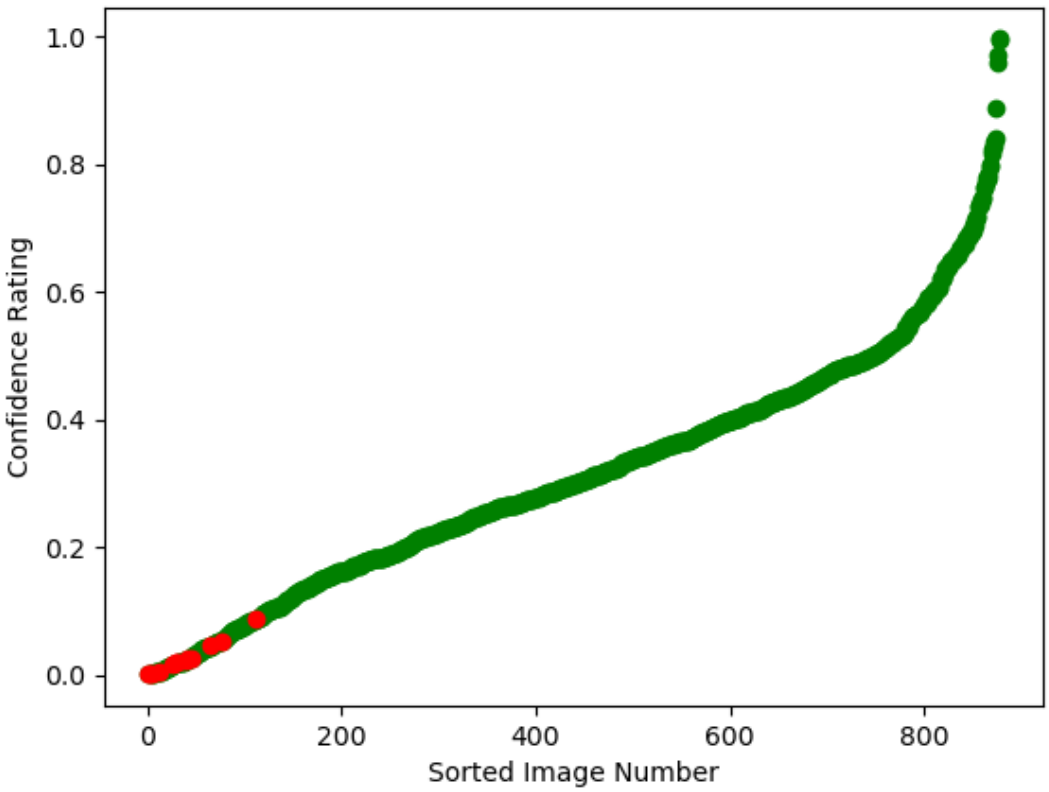


Figure 4.7: A graph showing proportional match difference as confidence rating. A higher rating is better. Red dots are incorrectly matched images.
Green Dot: Correctly matched image.
Red Dot: Incorrectly matched image.

The graph that looks at the PMSD seems like the better choice based on the first mistake happening at a lower confidence than the pure match score difference-based approach.

The confidence rating can provide a customizable cut-off enabling low certainty matches to be discarded. A high cut-off would allow for a 100% correct match rate, although this would discard all low confidence matches, even the correct ones. A lower rating would discard the worst mistakes but keep some incorrect matches.

A cut-off rate can be set just above the first mistake achieving a 100% match rate but discarding 12.8% of all images due to low confidence. This is illustrated in Figure 4.8.

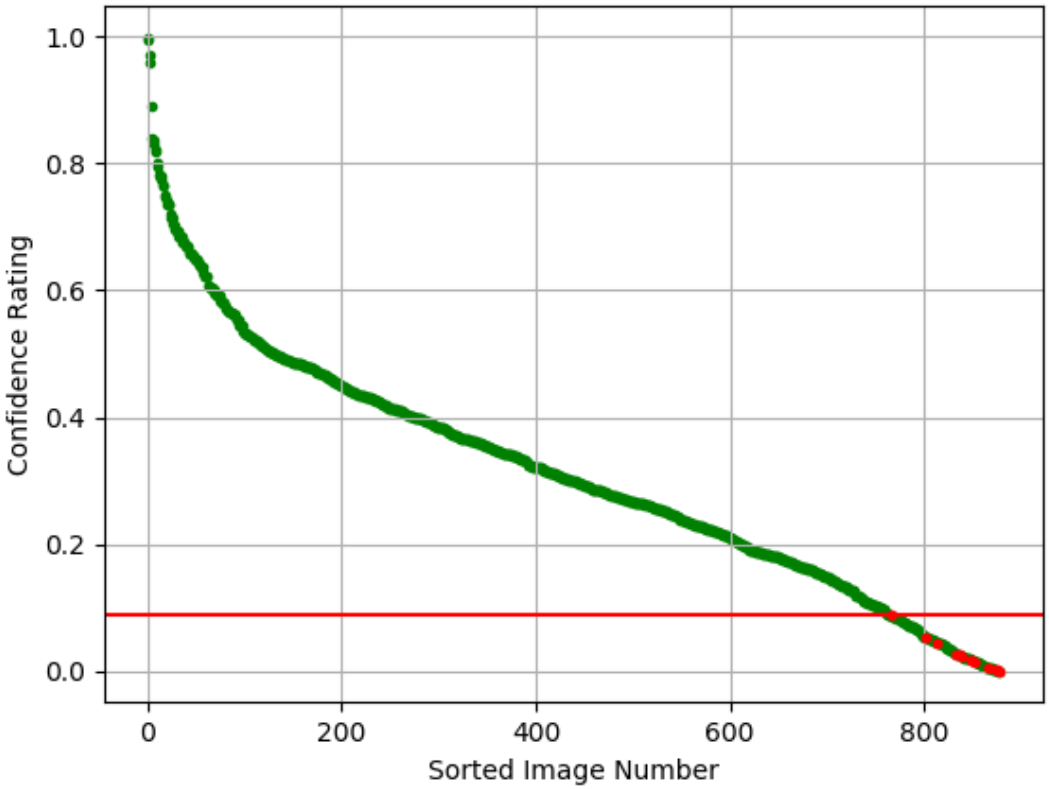


Figure 4.8: A cut-off set at a confidence rating just above the first mistake of the small experiment. The algorithm now achieves 100% correct match rate but discards 12.8% of all images due to low confidence.
Red Line: cut-off line.

The introduction of a cut-off rate allowed for false negatives and true negatives. By using the definitions for false negative and true negative from section 4.1.1, the precision and recall scores were evaluated [4.1, 4.2] for the full dataset of 395 identities. The results can be seen in Figure 4.9. Every data point after the first represents a mistake and, therefore, a change in the precision-recall score.

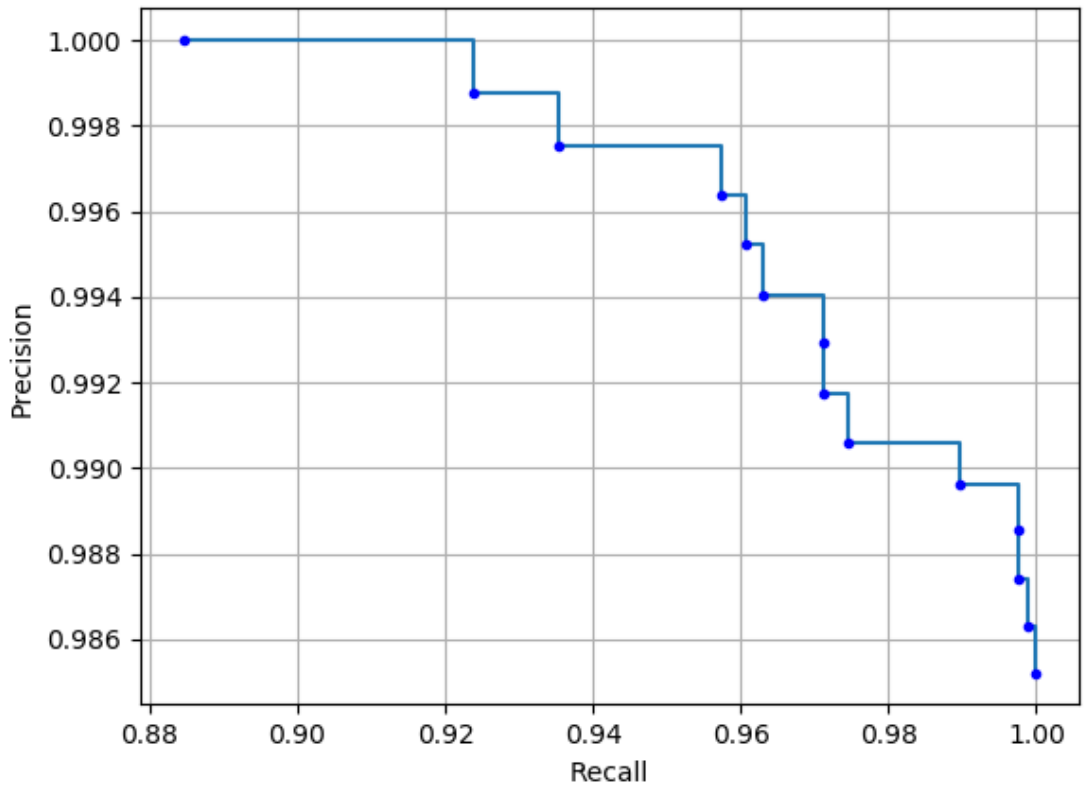


Figure 4.9: Precision-recall curve from experiment 1. Note: the scale on precision is a lot higher than on recall.

4.4 Feature Contributions

Investigation of underlying mechanics was conducted to highlight the individual performance of the components. The algorithm was broken down into its components. These components were then used to match the database individually and in smaller groups. The result of this can be seen in Fig. 4.10.

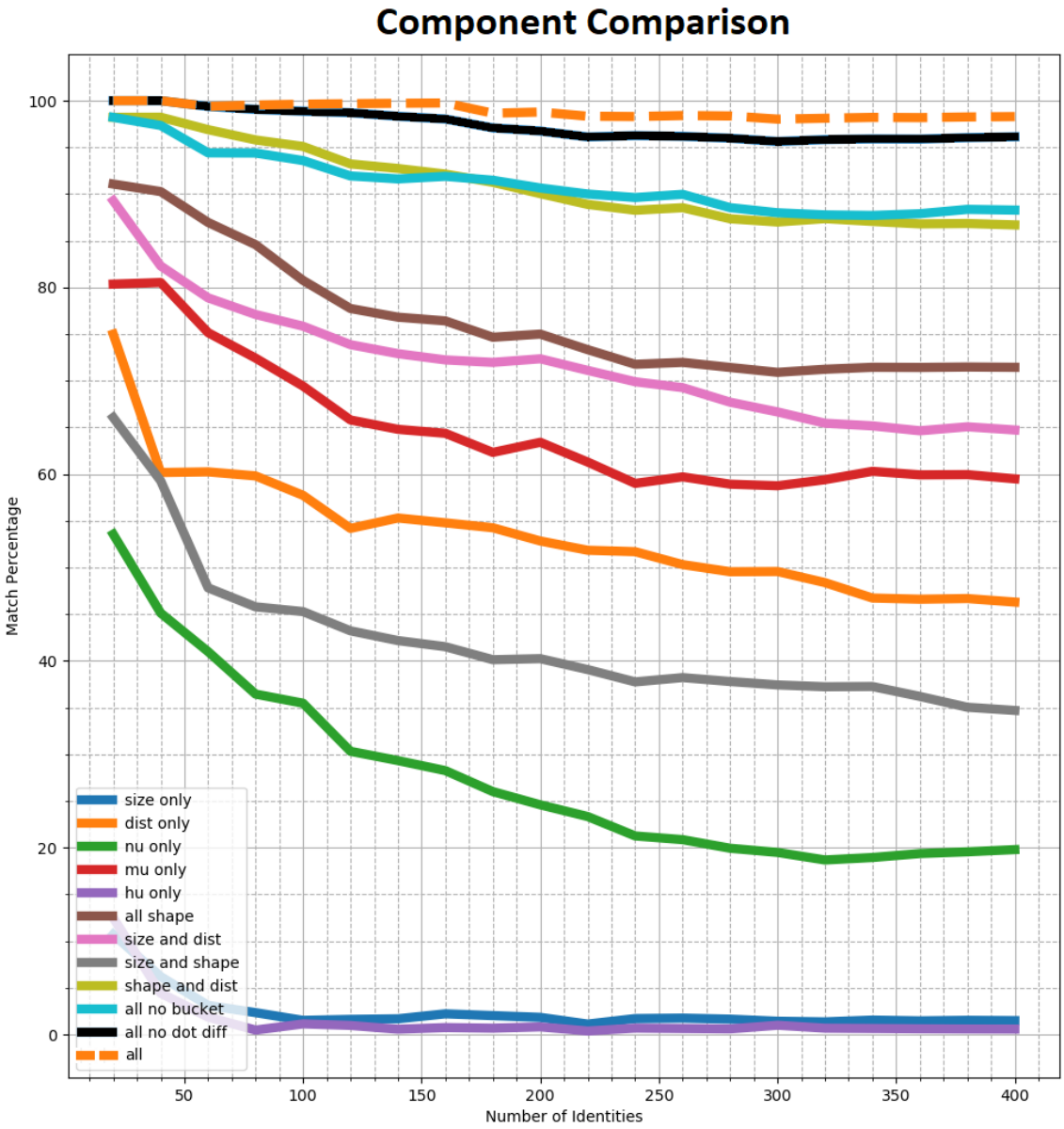


Figure 4.10: A graph showing the match score of the components of the algorithm. All components are in their weighted form 3.5.5. Nu, mu, and hu are all variants of the image moment scores (shape) 2.5.1. Dist refers to matching based on a spots' location, a simple (x, y) distance. All shape is the combined score of the nu, mu and hu scores. All no bucket is the combined score of size, distance, and all shape but without spot difference and bucket score 3.5.4. All no spot diff is the combined score of all, including bucket score, but without a simple comparison of the number of spots 3.5.2. All is the complete algorithm.

The graph above 4.10 clearly demonstrates the power of the additional features. The feature with the highest contribution on its own is the mu shape, even higher than the location contribution. Only looking at the mu shape of the spots gives a correct match percentage of 60% for 395 identities. Looking at the combined shape score (mu, nu, and hu) achieves a match score of 70% on its own. Although it should be noted that these scores are largely a reflection

of the algorithm, it certainly shows that there is a large potential in using both shape and size in addition to location when matching spots.

Of all shape matching scores, the mu score 2.5.1 is the highest. The mu moments, also known as the central image moments, are simply translation invariant. This differs from the additional scale invariance of the nu moments and the even more complex hu moments, which are also invariant to scale, rotation, and reflection. This shows that the coordinate conversion 3.4 performed on the spots is working as intended as additional invariants only cause false matches. It can be assumed that the distance scoring strength is in part due to this normalized coordinate system. Intuitively it makes sense that hu moments would cause many false matches as it allows for rotated matches, which the coordinate conversion seeks to eliminate. Additionally, hu moments allow for reflected matches, which is not advantageous for spot matching.

Interestingly size is generally a bad matching feature. This can be seen as it is one of the weakest features in the figure 4.10, additionally when combined with another score such as size and distance or size and shape it performs worse than distance or shape respectively. Nevertheless, this feature still improves performance in the long run. Improvement can be seen when looking at shape and distance compared to 'all no bucket' (see figure 4.10, 'all no bucket' is essentially shape, dist, and size). In the beginning, 'all no bucket' performs worse, but over time the addition of the size features slightly outperforms the lack of this feature.

4.4.1 Double vs Triple Identities

Identities in the database consist of either two images or three images. As identities consisting of two images have one less comparison image, they may be more likely to be matched incorrectly. The number of identities consisting of double and triple images, as progressively more identities are added to the experiment, can be seen in Fig. 4.11. Additionally, the number of double and triple image identities in the mistakes can be found in Fig. 4.12.

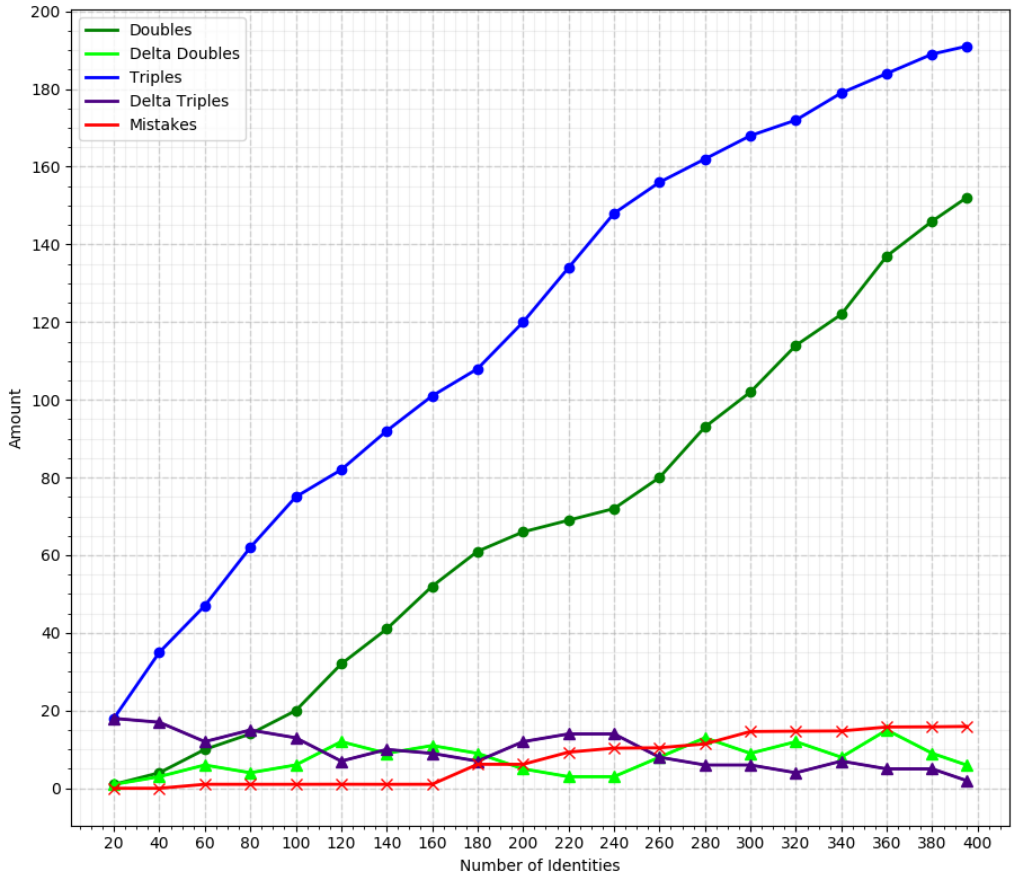


Figure 4.11: A graph showing the number of triple and double identities compared to the number of mistakes as progressively more identities are added to the experiment. Additionally the change in number of double and triple identities is plotted as Delta Double and Delta Triple.

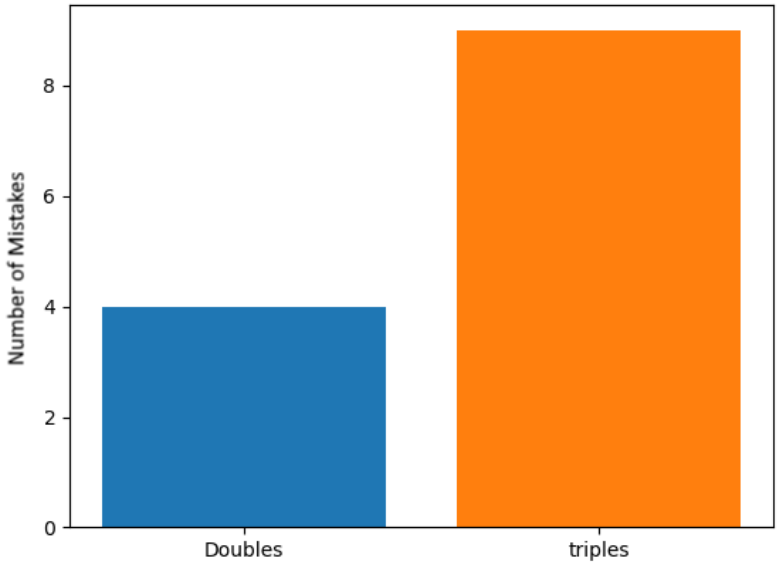


Figure 4.12: A graph showing the number of mistakes by identity type. A double identity only contains two images from that identity. A triple identity contains three images. More mistakes were made on identities containing three images.

The data points to no correlation between the number of images in an identity, and its likelihood to be mismatched. However, it is hard to draw conclusions from a small number of mistakes.

4.4.2 Cameras

Similar to the possibility that the number of images in an identity could affect the outcome, the camera the image was taken from could affect the result. In Figure 4.13, the number of mistakes from their respective cameras were plotted.

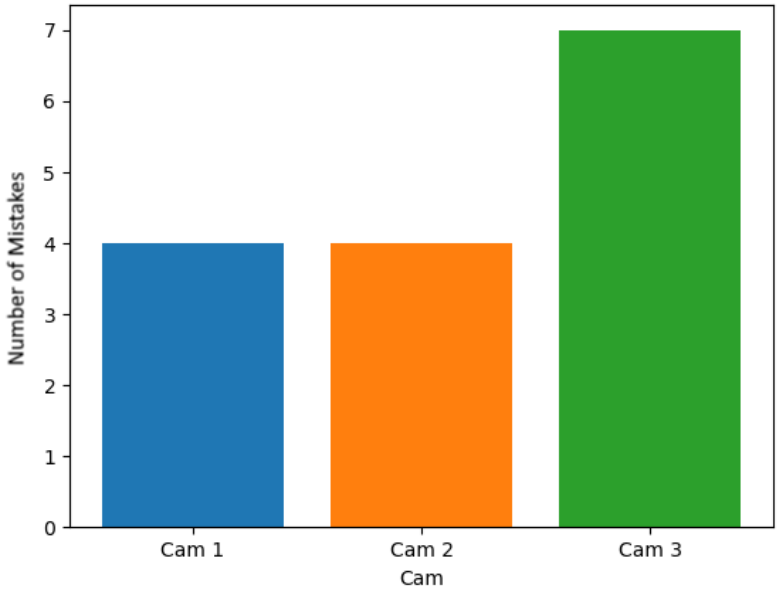


Figure 4.13: The number of mistakes from each camera.

Once again, there does not seem to be a correlation. It is possible that camera 3 is slightly harder to match than the rest. Camera 3 is at a medium distance from the fish, and therefore, in theory, it should not be the hardest to match. Therefore given the small amount of data due to few mistakes, it is hard to draw a conclusion.

4.5 Experiment 2

The second experiment was conducted using a larger dataset containing 1000 identities and 2998 images. A true positive rate 4.1.1 of 96.83% was achieved. A total of 94 images were matched to an incorrect identity; seven of these incorrectly matched images were unable to be matched due to the segmentation network incorrectly returning a blank image (incorrectly returned 0 spots). 2904 images were correctly matched to their true identity.

Using the same confidence rating system as was used in the smaller experiment 4.3, the confidence was logged for each match. In general, the confidence rate of the larger dataset is lower than the rate of the smaller experiment. The confidence rating as difference score can be seen in Figure 4.14 and the confidence rating as proportional difference can be seen in Figure 4.15.

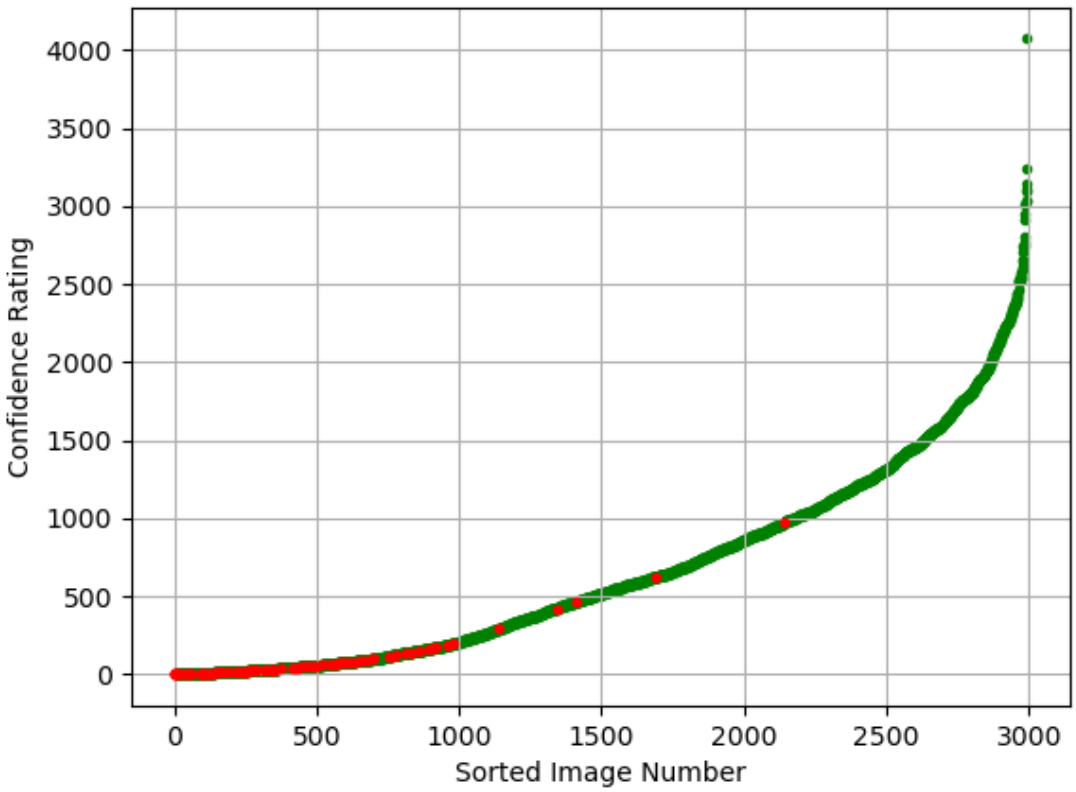


Figure 4.14: Confidence rating given by match score difference.
Green Dot: Correctly matched image.
Red Dot: Incorrectly matched image.

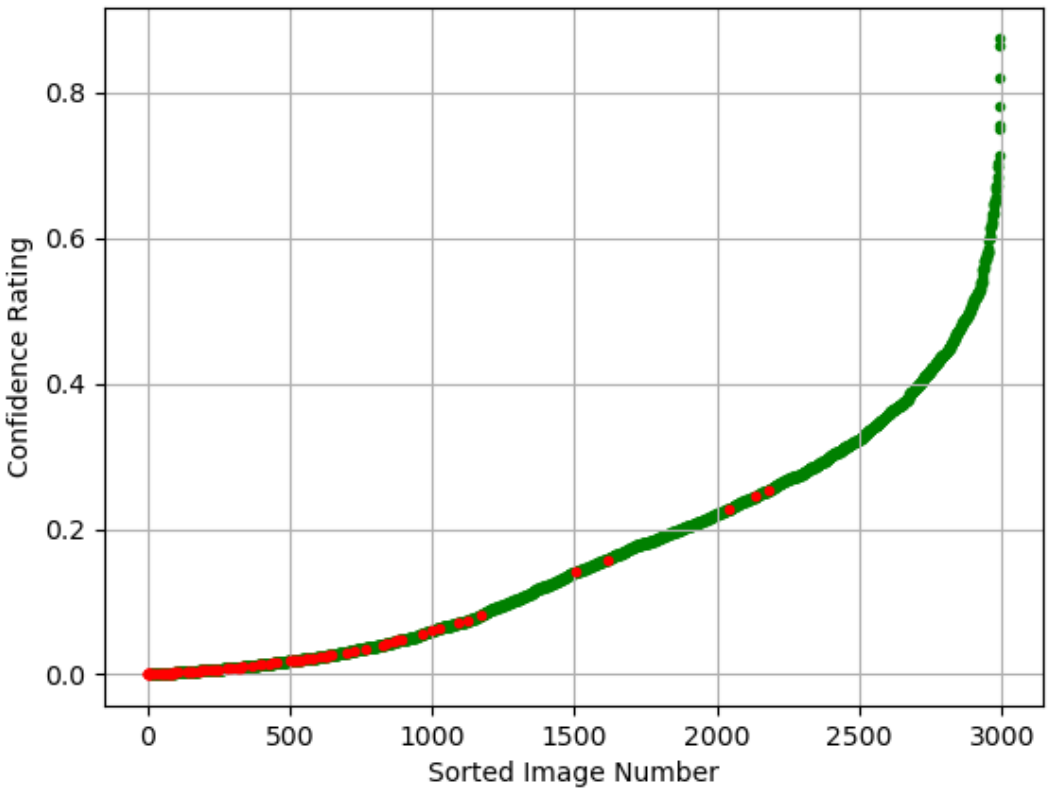


Figure 4.15: Confidence rating given by proportional match score difference.
Green Dot: Correctly matched image.
Red Dot: Incorrectly matched image.

The precision-recall curve was also calculated using the same definitions as experiment 1 [4.2]. The results can be seen in Figure 4.16. Every data point after the first represents a mistake and, therefore, a change in the precision-recall score.

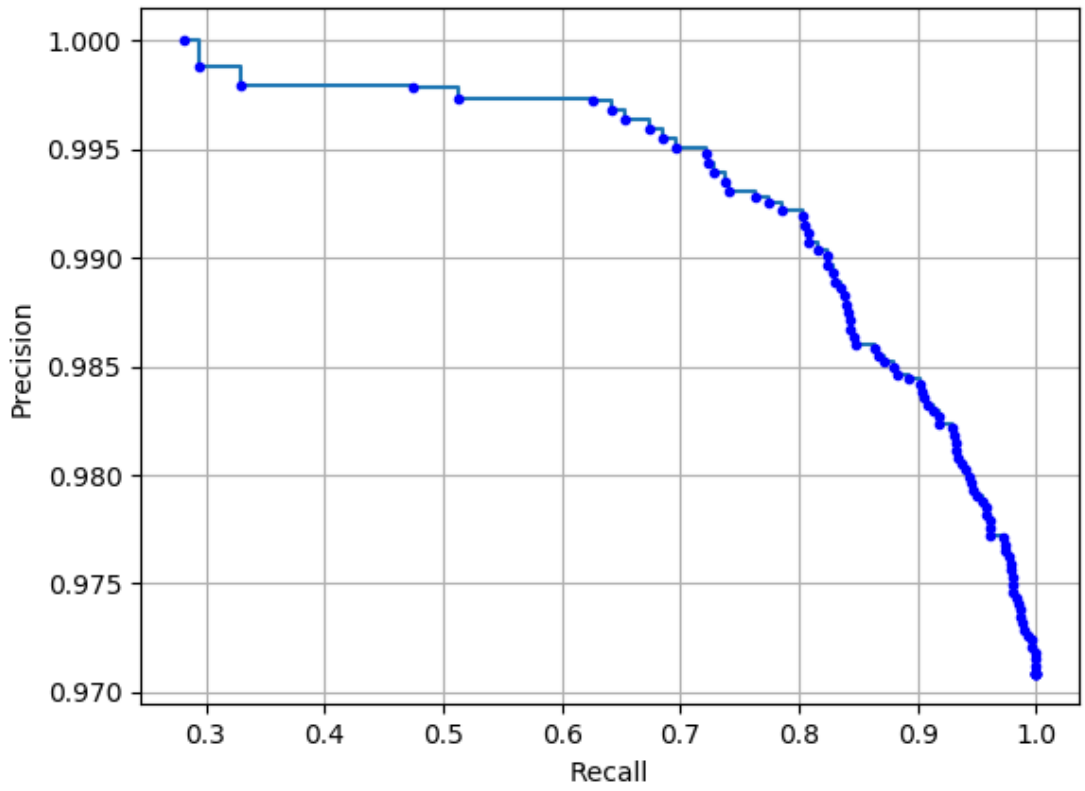


Figure 4.16: Precision-recall curve from experiment 2. Note: the scale on precision is a lot higher than on recall.

4.6 Decline of Match Score With More Identities

There is a noticeable decline in performance as more identities are added. This is intuitive as more comparisons lead to more possibilities for mistakes. A graph plotting the decline as more identities are added can be seen in Figure 4.17. The trend appears to have roughly linear characteristics.

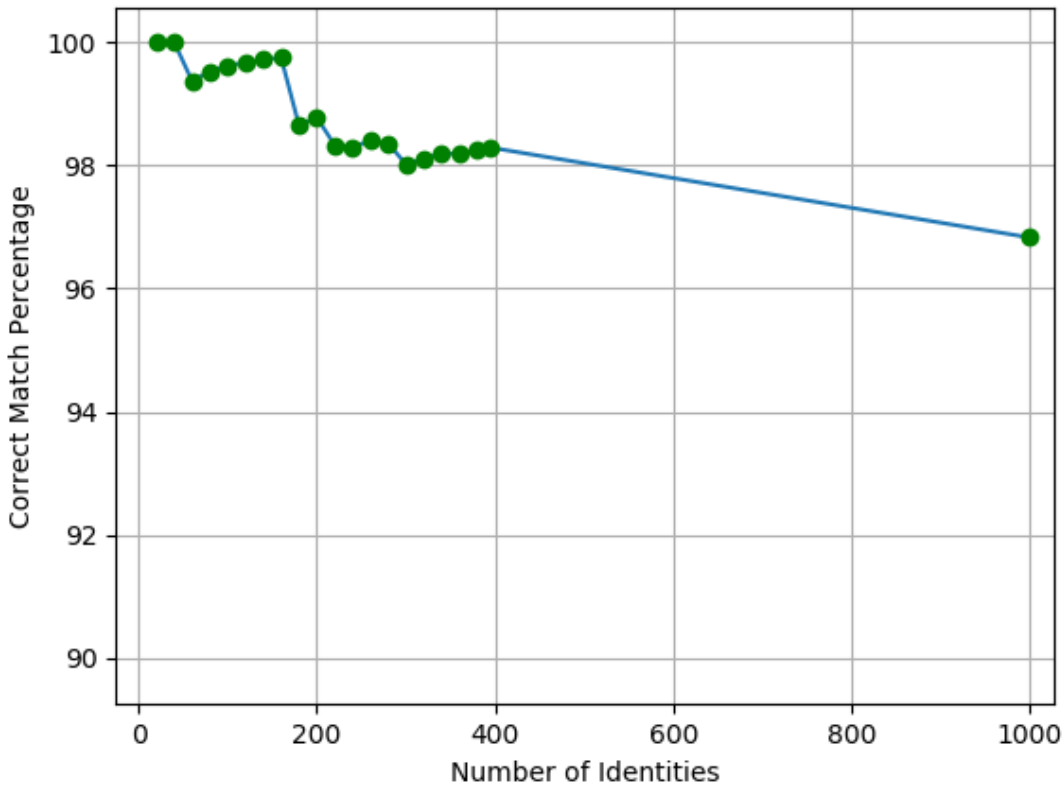


Figure 4.17: A graph showing the decline in correct match percentage as more identities are added to the database. The first 20 data points are taken from experiment 1 4.2, the final data point is taken from experiment 2 4.5.

Chapter 5

Discussion

5.1 AI Network

As the algorithm developed attempts to find the best spot match for every spot, there are some cases that it does not handle well. Mainly the case of split spots, this is where the segmentation network in one image gives one connected spot and in another two split spots. When converting the segmentation to numerical values, this causes two smaller spots to be registered instead of a larger one. An example of what this segmentation looks like can be found in Fig. 5.1.

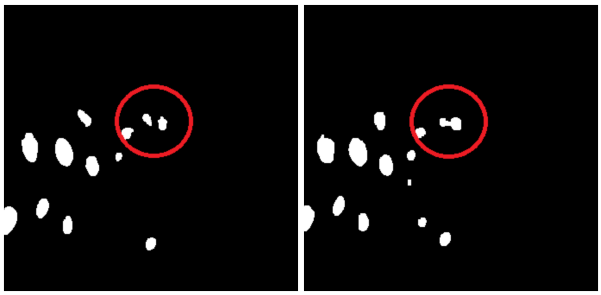


Figure 5.1: The same spot is in one image connected and in another split into two.

This appears to have a significant impact on the match score of the images as the spot in question will no longer be correctly matched. Additional punishment is added as the number of spots is no longer the same. This means that the more accurate the AI network for extracting the spot segmentation is, the better the result for matching will be.

5.2 Feature Choice

The main difference in approaches between the proposed features in this research and the approach used by Eilertsen et al. [15] is the choice of features. This approach uses the spots on the head while Eilertsen et al. achieved the best results with an approach that used all the spots on the fish, including the body.

Eilertsen et al. also had an approach using only the spots on the head of the fish; this did not yield satisfactory results. The eye was used in addition to the shortest distance between the eye and forehead to create a polar coordinate system. A comparison of coordinate features can be found in Fig. 5.2.

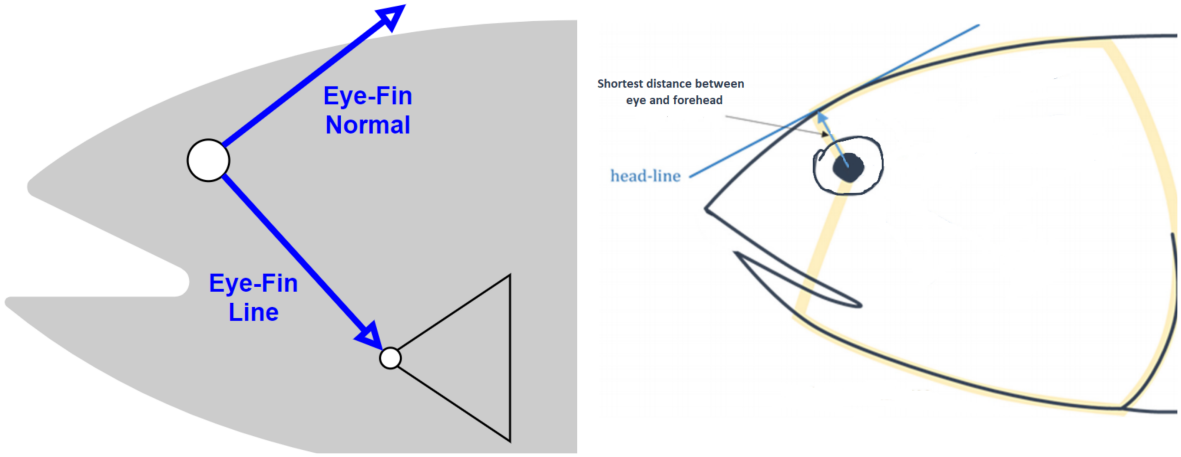


Figure 5.2: A comparison of the two head based approaches.

In the data used by this research and Eilertsen et al., the fish are presented on a conveyor belt with a perfect sideways profile. This is not indicative of real-world use for biometric identification of salmon as this would take place underwater with live salmon swimming. A possible issue with using the whole body of the fish is the deformation of the body when swimming. Fish bend their body to create thrust with their tails, which essentially means a significant non-linear transformation of the spots on the skin of the fish. This is potentially solved using only the head for identifying features as the head of the fish is bony and, therefore, rigid. This means that the head of the fish suffers mostly linear transformations of spots, which can be corrected for using the eye-fin coordinate system.

5.3 Number of Spots

Some fish have very few spots on their head, and since this is the basis for the biometric comparisons, it would be intuitive to think that fish with few spots are difficult to match. The mistaken identities vs. the number of spots from experiment 1 4.2 were plotted against the entire database to investigate this hypothesis. The results can be seen in Fig. 5.3.

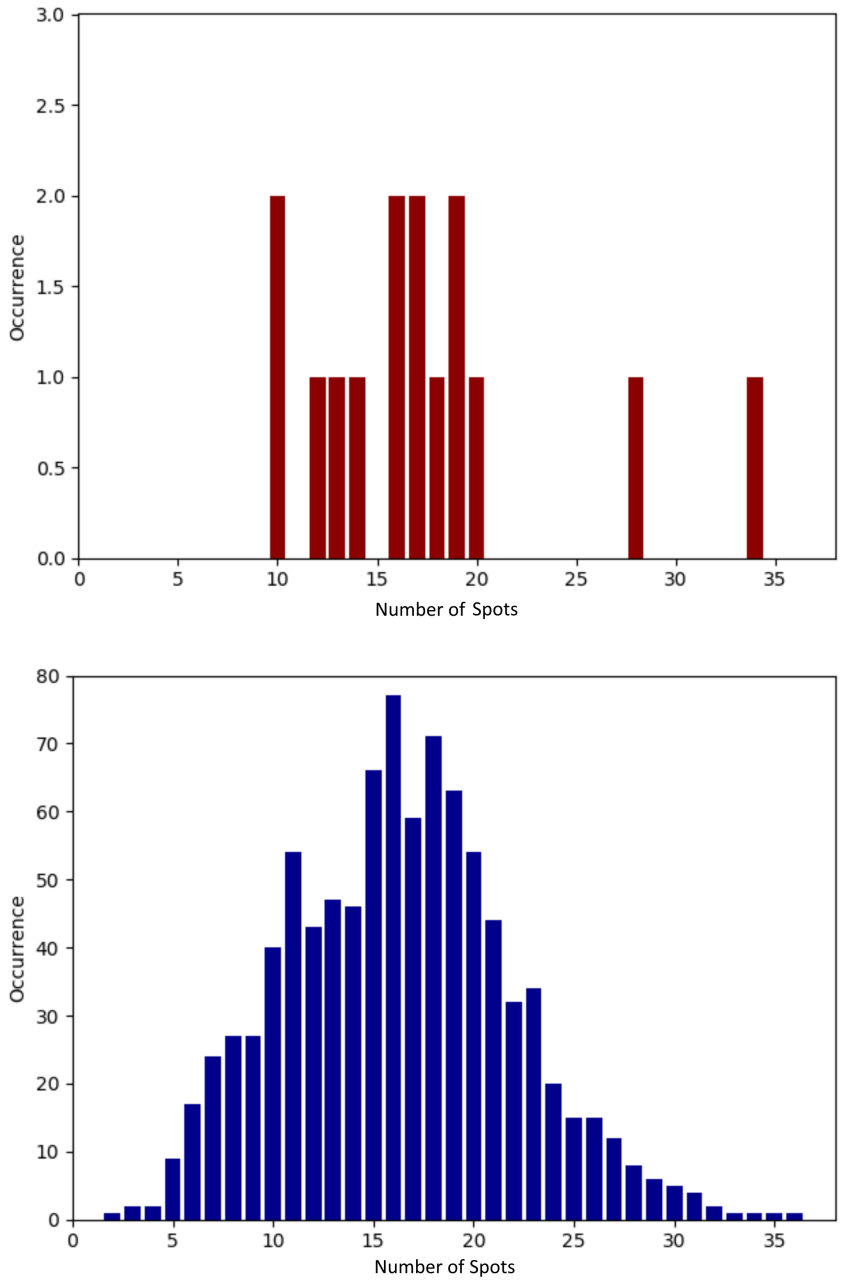


Figure 5.3: The top graph shows the number of spots in the mistaken identities. The bottom graph shows the number of spots for every fish in the database.

The results from the figure show that the biometric algorithm is slightly more mistaken towards larger numbers of spots if anything. Interestingly that means that a fish with as low as two spots was matched correctly, showing that even with little biometric information, it is possible to match based on location, size, shape, and the number of spots.

Another intuition that the graph shows is that mistakes happen on similar fish. Essentially, it makes sense that 16 spot fish would be mistaken for another 16 spot fish than a two-spot fish, and since there are many fish with around 16 spots, it is more likely to make a mistake. Additionally, the two-spot fish might be the only two-spot fish, therefore a natural match.

5.4 Image Moments

Although image moments are, in this case, used in a controlled environment due to the unique coordinate system, they have the potential to be used in many more cases. There are progressively more invariant image moments that can be used to match shapes and sizes in controlled and uncontrolled environments.

Due to the nature of the coordinate system, mu moments are best suited to matching in this environment, as explored in 4.4. However, the progressively more invariant moments, such as the nu 2.5.1 and hu moments 2.5.1 could be beneficial even to completely unregulated coordinate systems. Perhaps this could even be worked into the constellation algorithm, such as the one used by Eilertsen et al. [15].

Chapter 6

Conclusions and further work

6.1 Conclusion

This research has explored biometrical features for identifying salmon and conducted experiments utilizing novel spot properties. A novel biometrical feature, the pectoral fin, is also proposed as the new standard for constructing invariant coordinate systems on fish.

An experiment was conducted with a database consisting of 931 images and 395 identities. This experiment achieved a true positive rate of 98.29%, including an in-depth analysis of the algorithm mechanics.

Another more extensive experiment was conducted with a database consisting of 2998 images and 1000 identities, the largest biometrical matching of salmon to date. This experiment achieved a true positive rate of 96.83%.

A confidence rating was established, allowing for a trade-off between precision and recall.

The smaller database results are similar to the state of the art but require less and possibly more reliable biometrical information. The results on the larger database may serve as the new baseline for salmon identification. This was achieved by utilizing image moments to extract the size, shape, and location properties of the spots. Results show that the spots' shape may be the single best identifier for the spots on salmon due to their significant shape variance. Image moments provide progressively more invariant alternatives, which can be used in many different matching environments enabling them to be beneficial to most approaches at spot matching.

6.2 Further Work

6.2.1 Relational Component to Algorithm

One of the main issues with the current approach is its lack of relational data between the spots. Each spot is matched to others as an individual with no care of the relations between them. Although the algorithm currently does not consider this, it is possible to add.

One attempt was made using Delauney Triangulation, which has previously been used in things like fingerprint biometrics. This approach was loosely based on the paper on fingerprint biometrics by Wang et al. [27]. For this approach, a Delauney triangle mesh is first calculated. This is easily done by simply feeding the spot coordinates into the scipy-spatial library [28], which contains support for Delauney meshes. Each triangle in the mesh provides two invariants. Invar1 is the shortest side divided by the longest side. Invar2 is the middle longest side divided by the longest side. Lastly, invar3 is the largest of the angles. A visual representation of what Delauney meshes look like can be seen in Fig. 6.1.

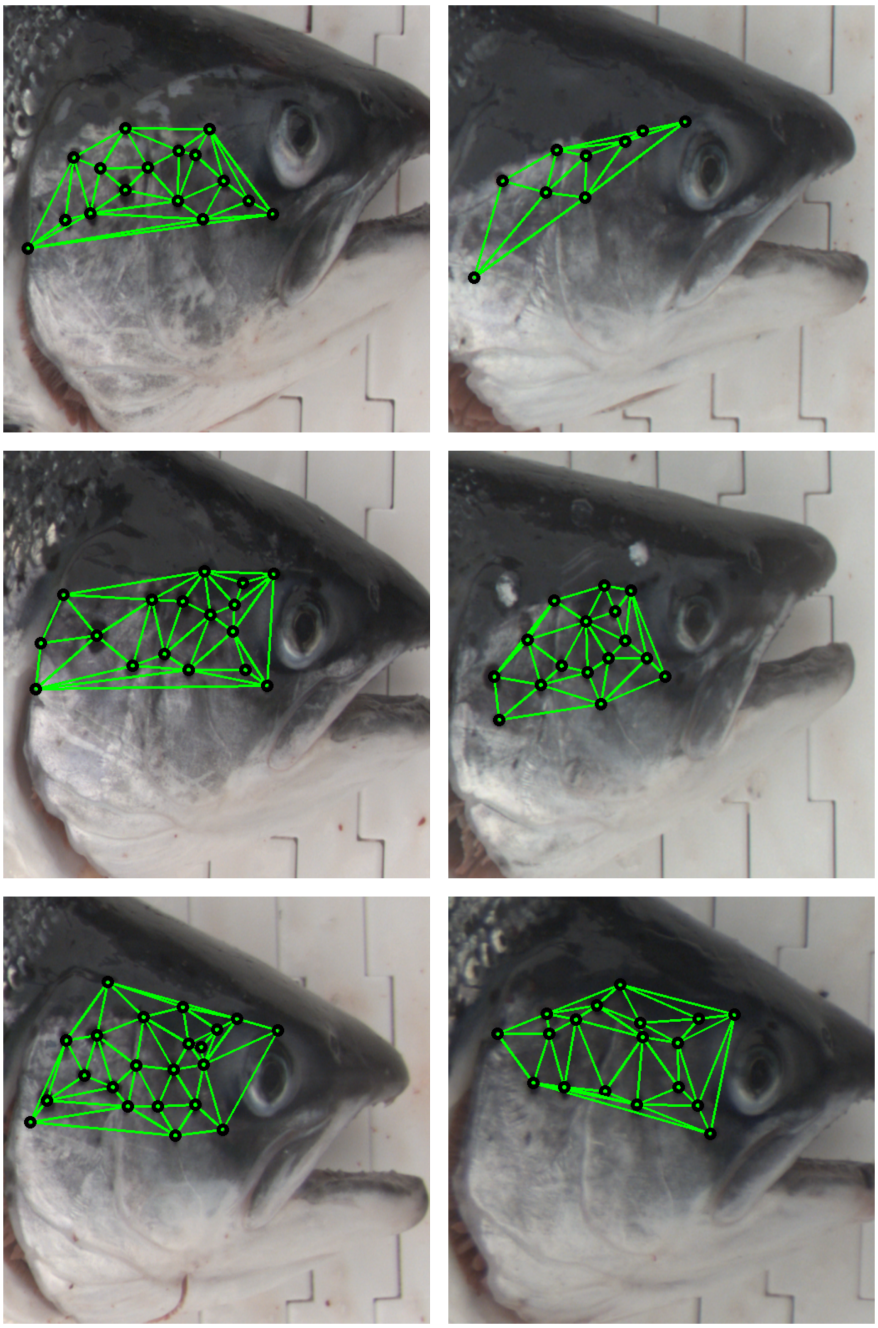


Figure 6.1: A visual example of what delauney triangulations look like using the spot data. Delauney meshes are overlayed their corresponding identity. Green vectors represent sides of a delauney triangle. Black circles represent spot locations.

Delauney triangulation is invariant to rotation and scale, which similar to hu shape is not necessarily advantageous in a controlled environment provided by the coordinate conversion. Although some experimentation was done using Delauney triangulation, a relational component to the algorithm was not achieved in this project. An approach similar to Delauney triangulation, the star constellation approach by Arzoumanian and Eilertsen [[16],

[15]], or an entirely different spatial relation approach would undoubtedly improve the algorithm.

6.2.2 Better AI networks

As mentioned in section 5.1, the AI networks are an essential aspect of the matching as they are the foundation for the data extraction. Anecdotally a small error in the eye-fin or head network does not seem to affect results significantly. On the other hand, the segmentation network seems to have a more substantial impact on the results. During the second experiment 4.5, seven images encountered issues when passed through the segmentation network. This resulted in blank segmentation images; therefore, no spots could be identified.

A possible solution to the neural network issues, such as those encountered during this thesis, is to improve the network performance. One way to do this is to label more data, which is effective but time-consuming.

6.3 Tuning

Tuning was done sequentially, and since there are many different weights, it is nearly impossible to find the optimum values. Perhaps a different approach to tuning would be more efficient. Nevertheless, tuning all weights to their optimum would yield better matching results.

6.3.1 Spot Splitting

Another improvement that can be made is the case of splitting spots. In some instances, a spot might get split into two spots while remaining whole on others. This poses a problem as the two spots from the split only match the original spot in location, but not size and shape. Improvements could be made to the segmentation network, the post-processing of the segmentation or the algorithm to handle the splitting of spots better.

6.3.2 Larger Data set

The largest experiment conducted in this thesis contained 1000 identities. Although more identities were available for testing on this could not be achieved in the short time frame

due to the large amounts of computational power needed when cross-referencing an entire database. Larger datasets can be made available for future use 6.4.

6.3.3 Optimization

The implementation of the comparison algorithm suffers from bad optimization in its current state. Ideally, a single feature vector could be created from the many spots on a fish. One issue that remains unsolved in this thesis is a good way to combine these spots into a single feature vector given the variance in the number of spots on different fish.

6.4 Data and Further Use

The biometrical feature database 3.11 and trained neural networks (from fall project by the author [14]) will be provided at request.

Bibliography

[1] NorgesSjømatråd. https://nokkeltall.seafood.no/, 2018.

[2] Luseprosjektet. https://lusedata.no/for-naeringen/veileder-for-telling-av-lakselus/, http://lusedata.no/wp-content/uploads/2012/06/20130705-Veileder-telling-av-lakselus.pdf, 2018.

[3] Bryton Shang. https://www.aquabyte.no/index.html, 2019.

[4] VisuaLice. https://www.eurostars-eureka.eu/project/id/4721, https://www.era-learn.eu/network-information/networks/eurostars/eurostars-cut-off-2/automated-sea-lice-detection-by-computer-vision, 2012.

[5] Cristina Tanase and Cornelis Teunissen. Light unit for counting sea lice, January 4 2018. US Patent App. 15/545,342.

[6] Peter Jans and Evert Gijtenbeek. A method for automatic sea lice monitoring in salmon aquaculture, October 25 2018. US Patent App. 15/767,888.

[7] Toni A Beddow, Lindsay G Ross, and John A Marchant. Predicting salmon biomass remotely using a digital stereo-imaging technique. *Aquaculture*, 146(3-4):189–203, 1996.

[8] Magnus Reiersen. Deep visual domain adaptation:-from synthetic data to the real world. Master's thesis, NTNU, 2018.

[9] Angelico Madaro, Rolf E Olsen, Tore S Kristiansen, Lars OE Ebbesson, Tom O Nilsen, Gert Flik, and Marnix Gorissen. Stress in atlantic salmon: response to unpredictable chronic stress. *Journal of Experimental Biology*, 218(16):2538–2550, 2015.

[10] BioMar. Håndteringsdødelighet.

[11] SINTEF. Stressed-out salmon get sick. 2017.

[12] CB Schreck, BL Olla, and MW Davis. Behavioral responses to stress. *Fish stress and health in aquaculture*, 62:145–170, 1997.

[13] Muhammed Suleamn Muhammed. Genetic variation in susceptibility of atlantic salmon (salmo salar l.) to salmon lice, lepeophtherius salmonis, from field and challenge tests. Master's thesis, Norwegian University of Life Sciences, Ås, 2018.

[14] Christian Hans-Pieter Kat Echtermeyer. Feature detection pipeline for biometrical identification of salmon, doi: 10.13140/rg.2.2.19897.60006. 2019.

[15] Aleksander Børresen Eilertsen, Jonatan Sjølund Dyrstad, and Morten Steen Bondø. Identifikasjon av lakseindivider—biometri fase 1 (salmid). 2017.

[16] Z Arzoumanian, J Holmberg, and B Norman. An astronomical pattern-matching algorithm for computer-aided identification of whale sharks rhincodon typus. *Journal of Applied Ecology*, 42(6):999–1011, 2005.

[17] Edward J Groth. A pattern-matching algorithm for two-dimensional coordinate lists. *The astronomical journal*, 91:1244–1248, 1986.

[18] Richard B Sherley, Tilo Burghardt, Peter J Barham, Neill Campbell, and Innes C Cuthill. Spotting the difference: towards fully-automated population monitoring of african penguins spheniscus demersus. *Endangered Species Research*, 11(2):101–111, 2010.

[19] Santosh Kumar and Sanjay Kumar Singh. Visual animal biometrics: survey. *IET Biometrics*, 6(3):139–156, 2016.

[20] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *IRE transactions on information theory*, 8(2):179–187, 1962.

[21] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[22] Zhihu Huang and Jinsong Leng. Analysis of hu's moment invariants on image scaling and rotation. In *2010 2nd International Conference on Computer Engineering and Technology*, volume 7, pages V7–476. IEEE, 2010.

[23] Jean Serra. *Image analysis and mathematical morphology*. Academic Press, Inc., 1983.

[24] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[25] Paul E Rybski, Daniel Huber, Daniel D Morris, and Regis Hoffman. Visual classification of coarse vehicle orientation using histogram of oriented gradients features. In *2010 IEEE Intelligent vehicles symposium*, pages 921–928. IEEE, 2010.

[26] Peter A Torrione, Kenneth D Morton, Rayn Sakaguchi, and Leslie M Collins. Histograms of oriented gradients for landmine detection in ground-penetrating radar data. *IEEE transactions on geoscience and remote sensing*, 52(3):1539–1550, 2013.

[27] Chengfeng Wang and Marina L Gavrilova. Delaunay triangulation algorithm for fingerprint matching. In *2006 3rd International Symposium on Voronoi Diagrams in Science and Engineering*, pages 208–216. IEEE, 2006.

[28] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1. 0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

Appendix A

A.1 Match Comparisons

A.1.1 Correct Matches

A few more examples of correct matches.

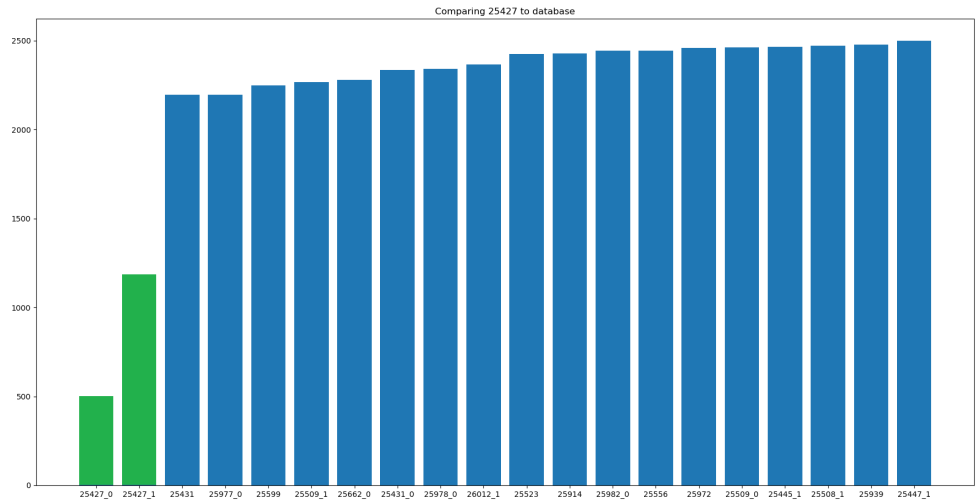


Figure A.1: The 20 best matches for a true positive identity. The x-axis shows identities, and the y-axis shows match score distance.

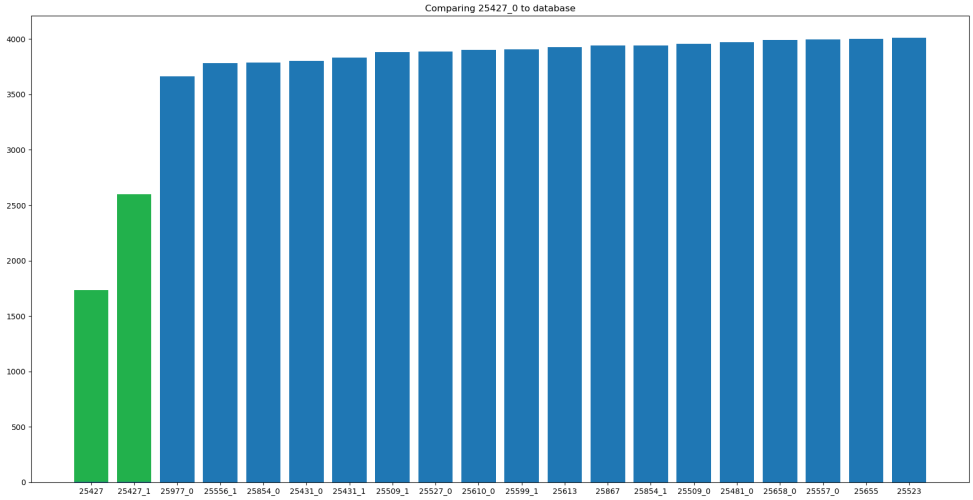


Figure A.2: The 20 best matches for a true positive identity. The x-axis shows identities, and the y-axis shows match score distance.

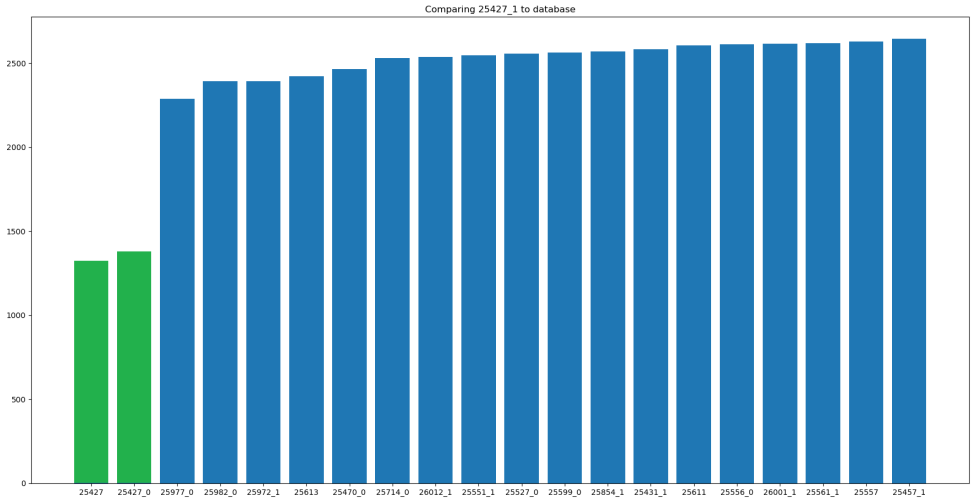


Figure A.3: The 20 best matches for a true positive identity. The x-axis shows identities, and the y-axis shows match score distance.

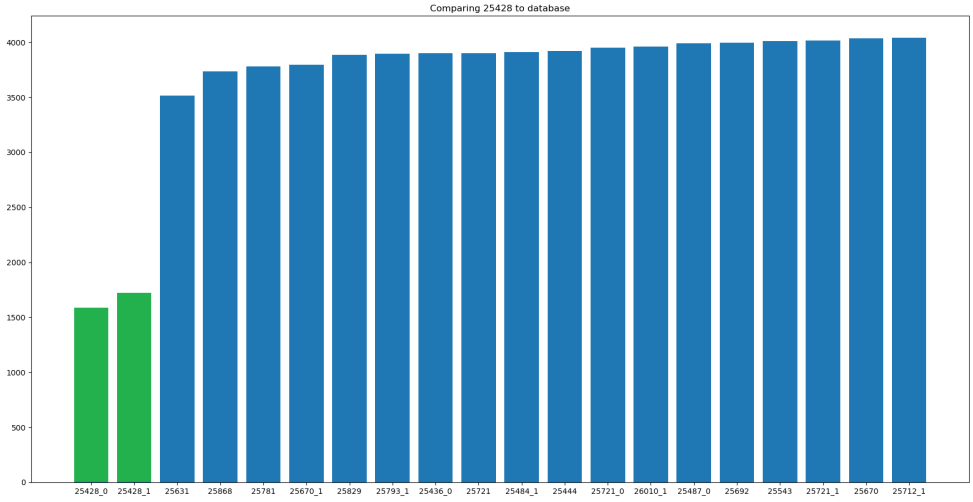


Figure A.4: The 20 best matches for a true positive identity. The x-axis shows identities, and the y-axis shows match score distance.

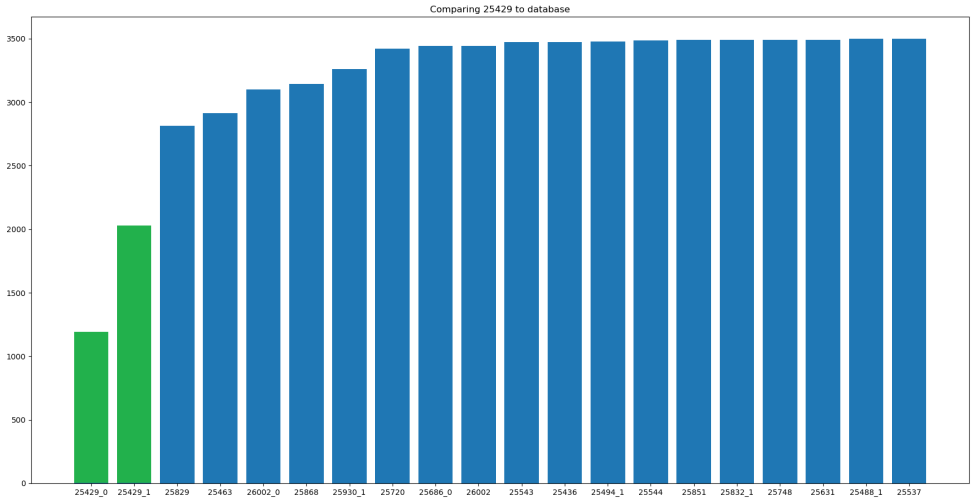


Figure A.5: The 20 best matches for a true positive identity. The x-axis shows identities, and the y-axis shows match score distance.

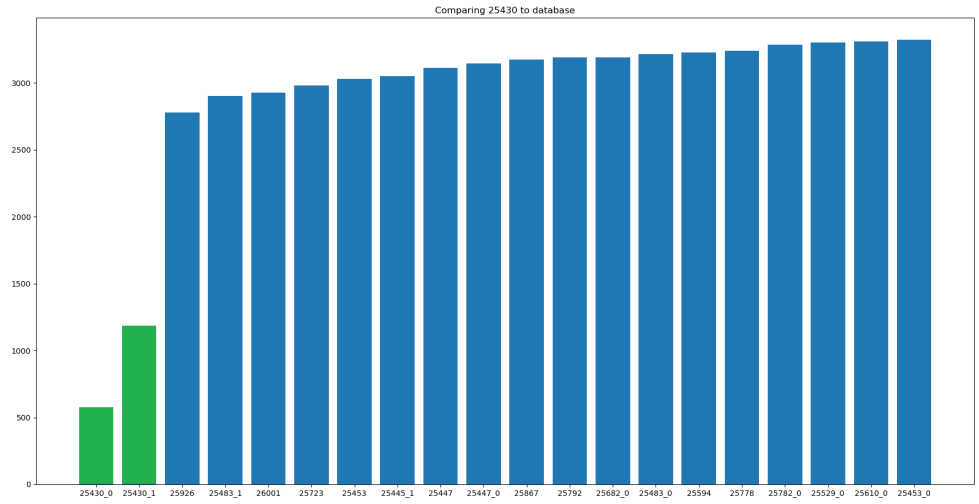


Figure A.6: The 20 best matches for a true positive identity. The x-axis shows identities, and the y-axis shows match score distance.

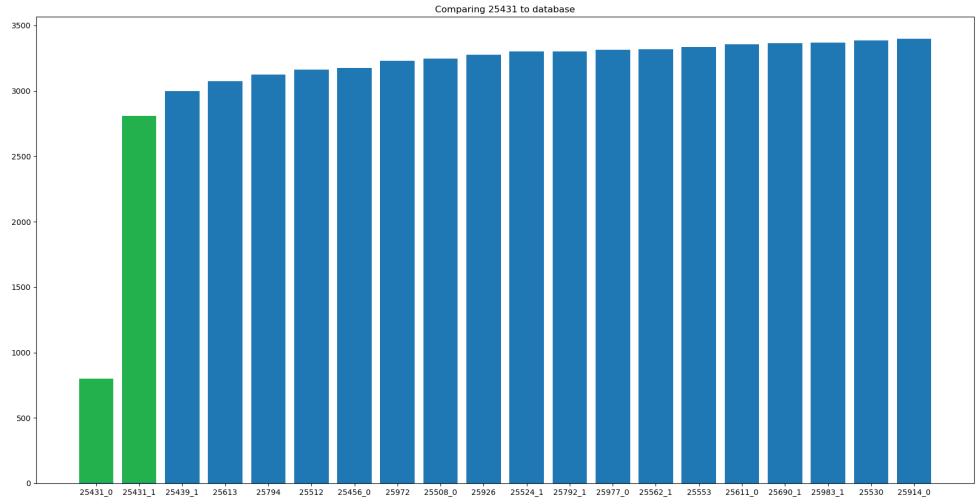


Figure A.7: The 20 best matches for a true positive identity. The x-axis shows identities, and the y-axis shows match score distance.

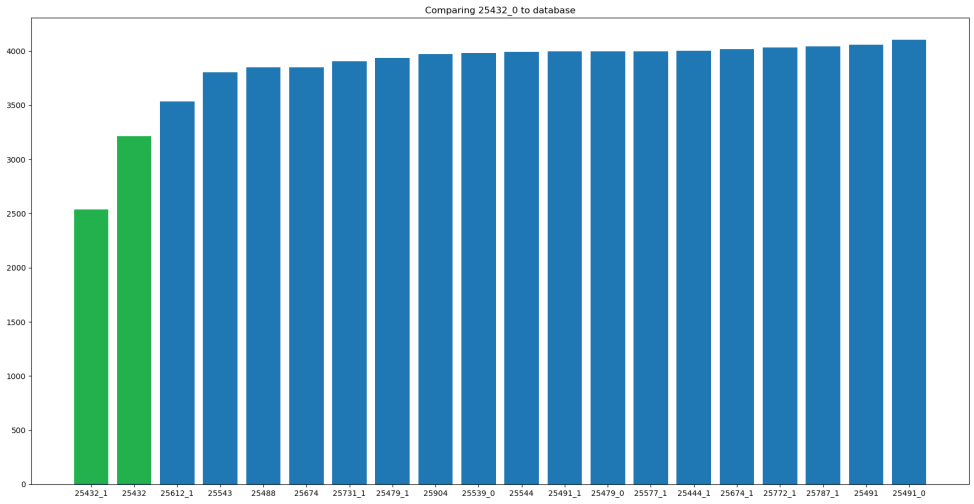


Figure A.8: The 20 best matches for a true positive identity. The x-axis shows identities, and the y-axis shows match score distance.

A.1.2 Incorrect Matches

All the 16 incorrect matches from the experiment.

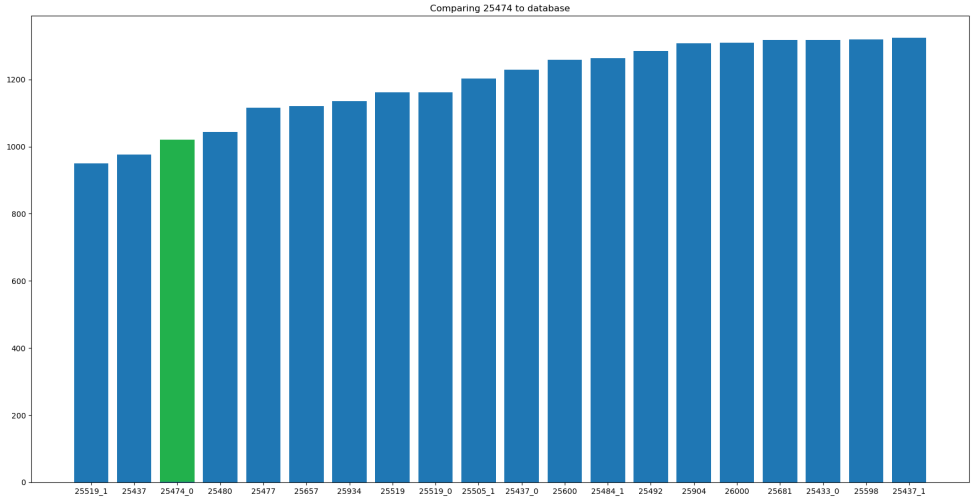


Figure A.9: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

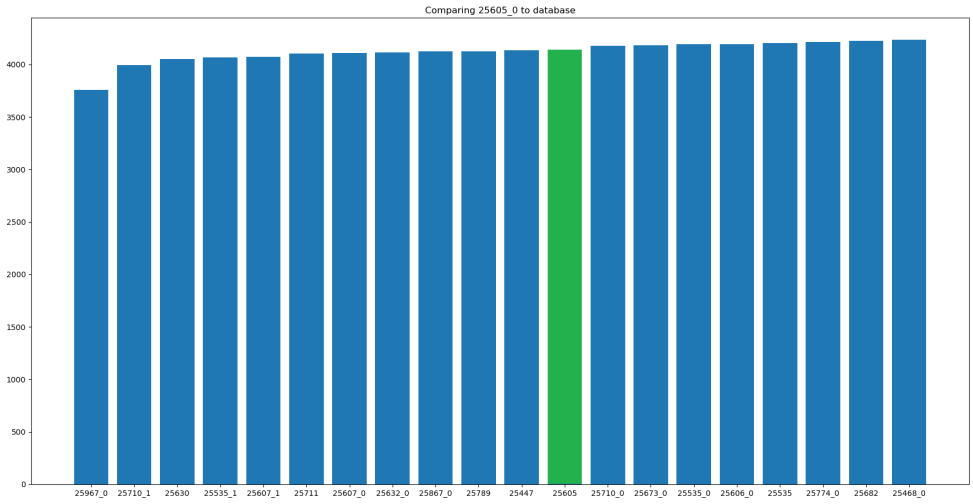


Figure A.10: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

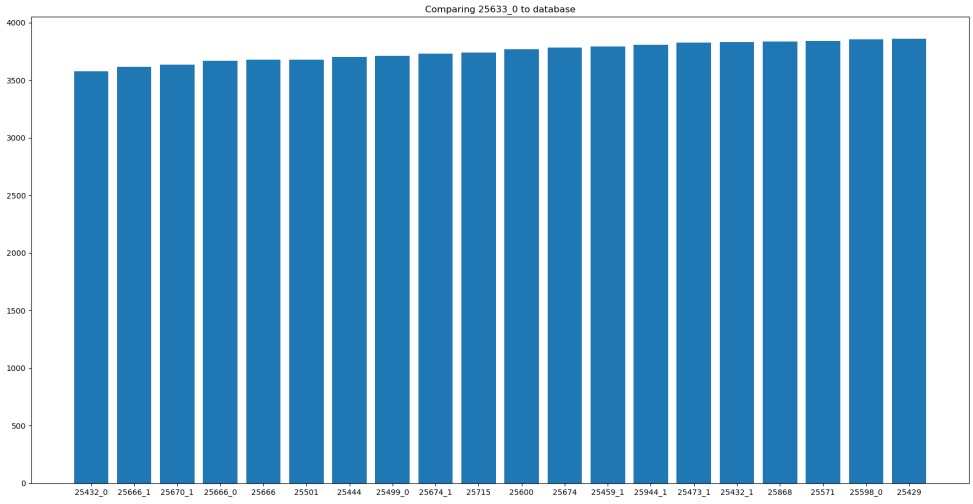


Figure A.11: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

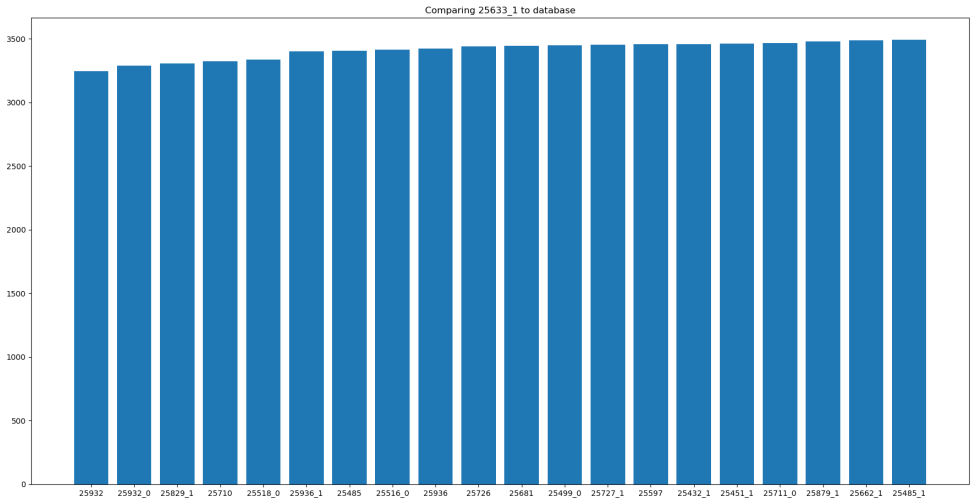


Figure A.12: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

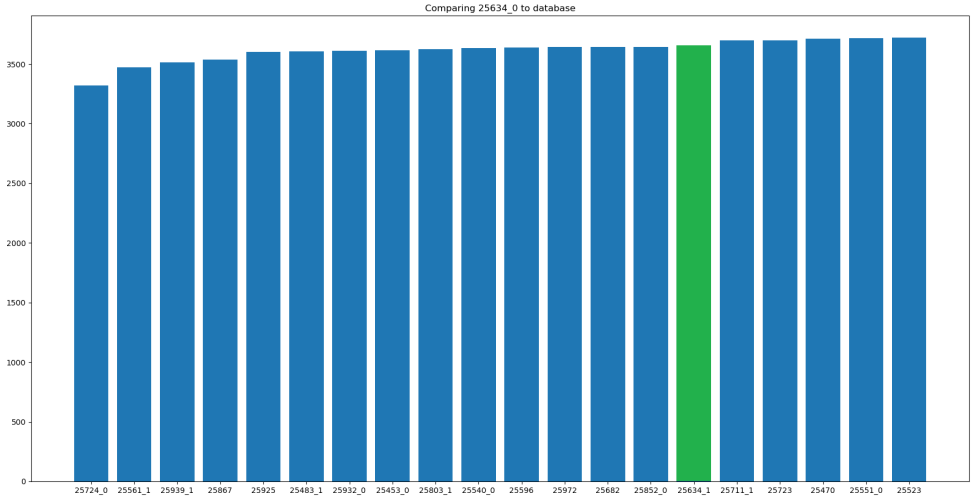


Figure A.13: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

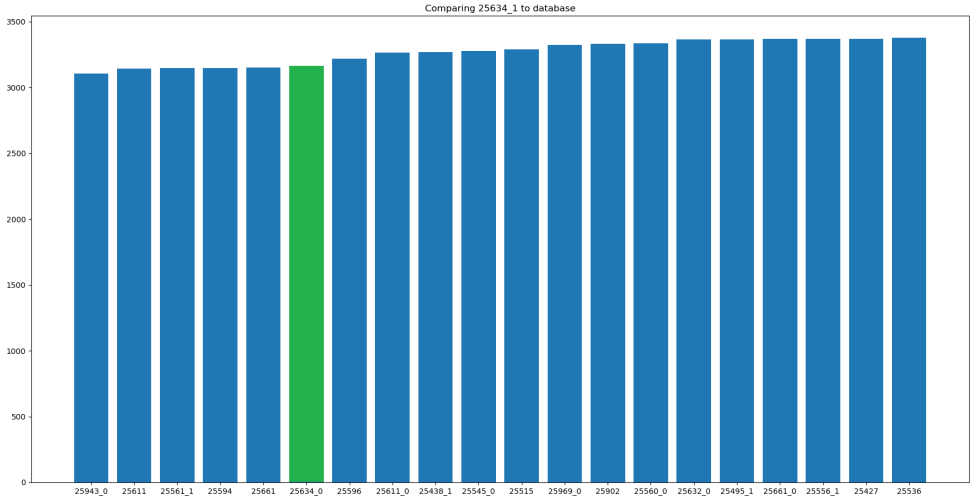


Figure A.14: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

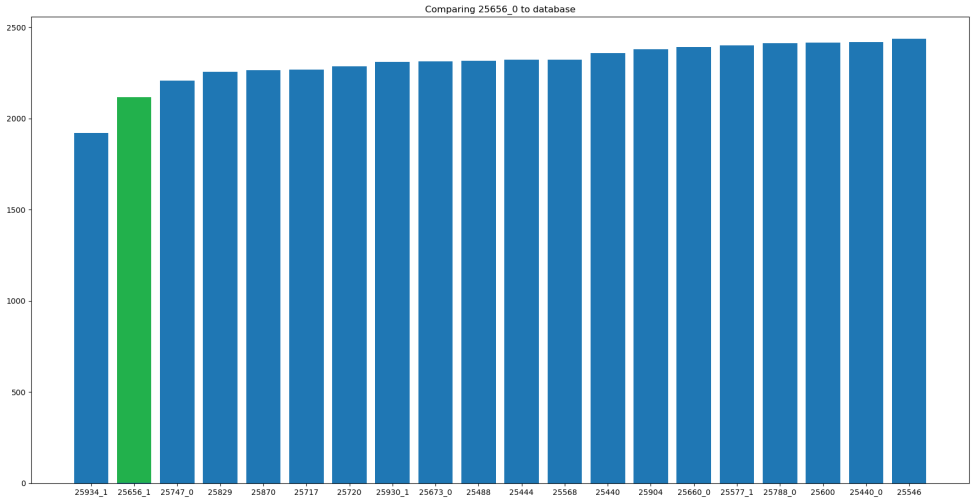


Figure A.15: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

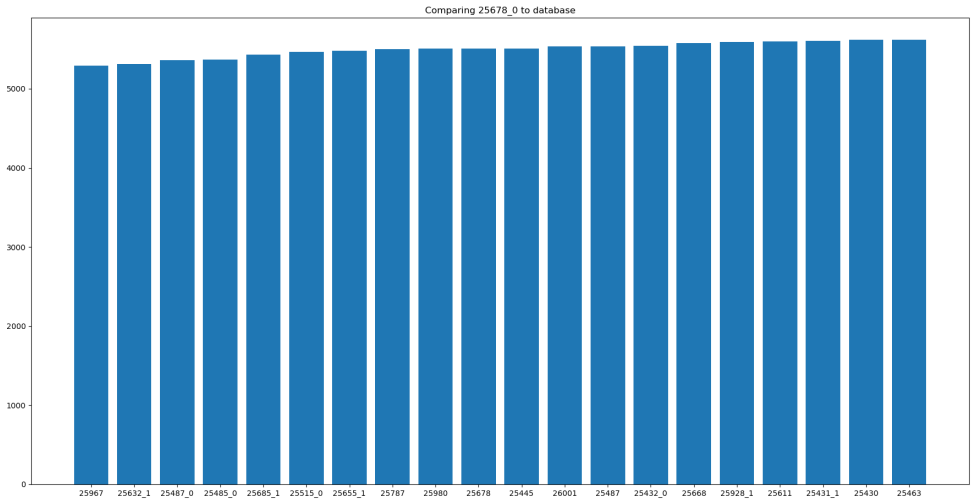


Figure A.16: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

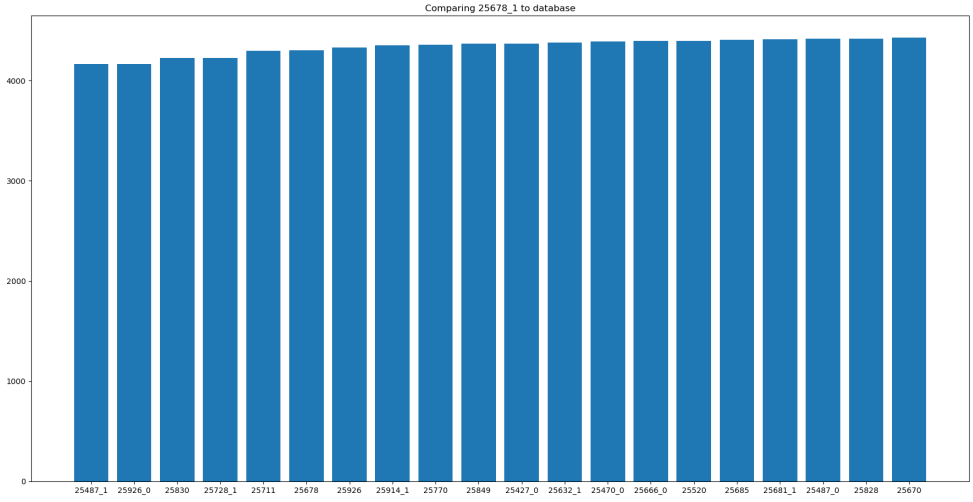


Figure A.17: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

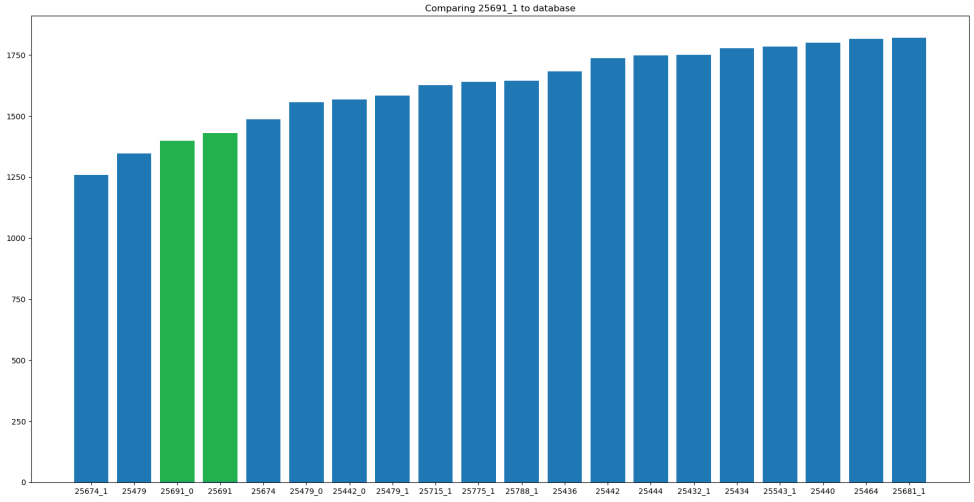


Figure A.18: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

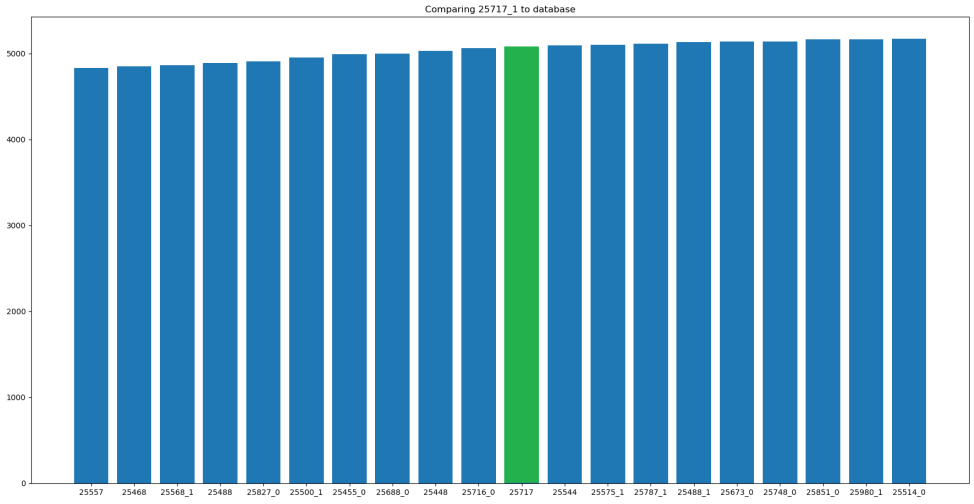


Figure A.19: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

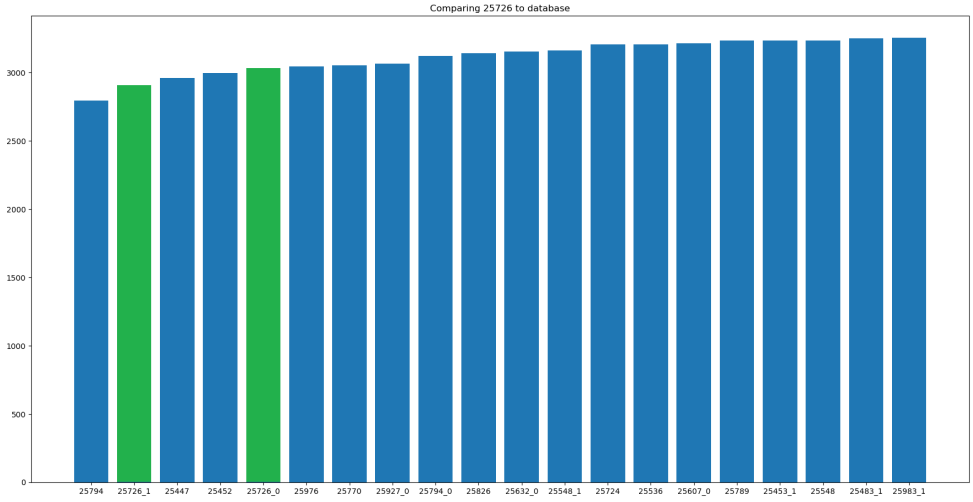


Figure A.20: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

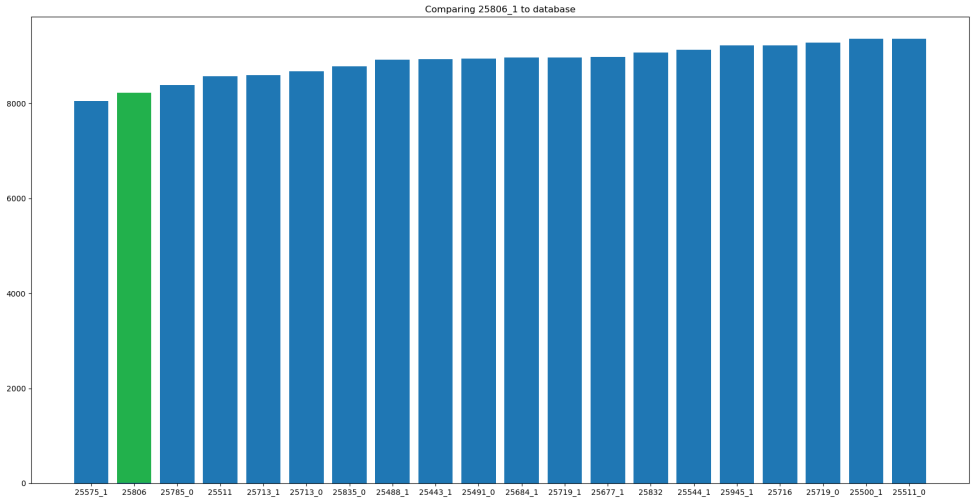


Figure A.21: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

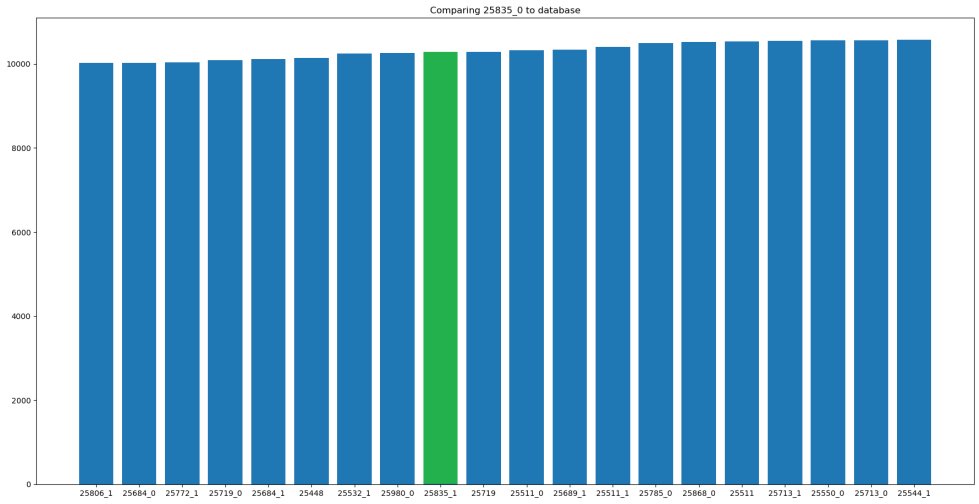


Figure A.22: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

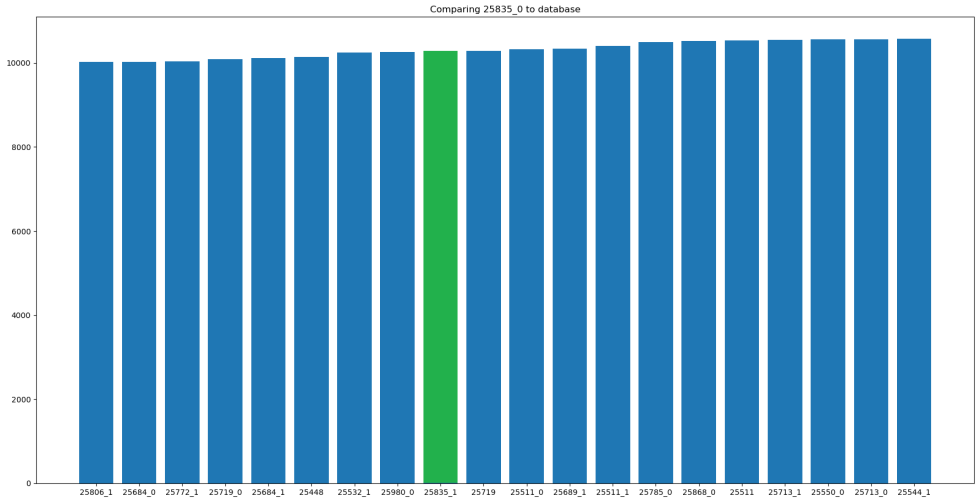


Figure A.23: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.

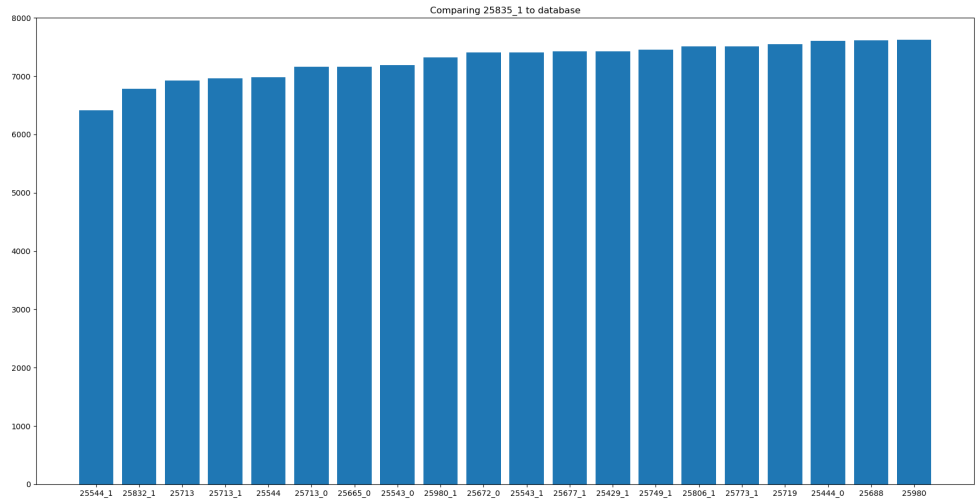
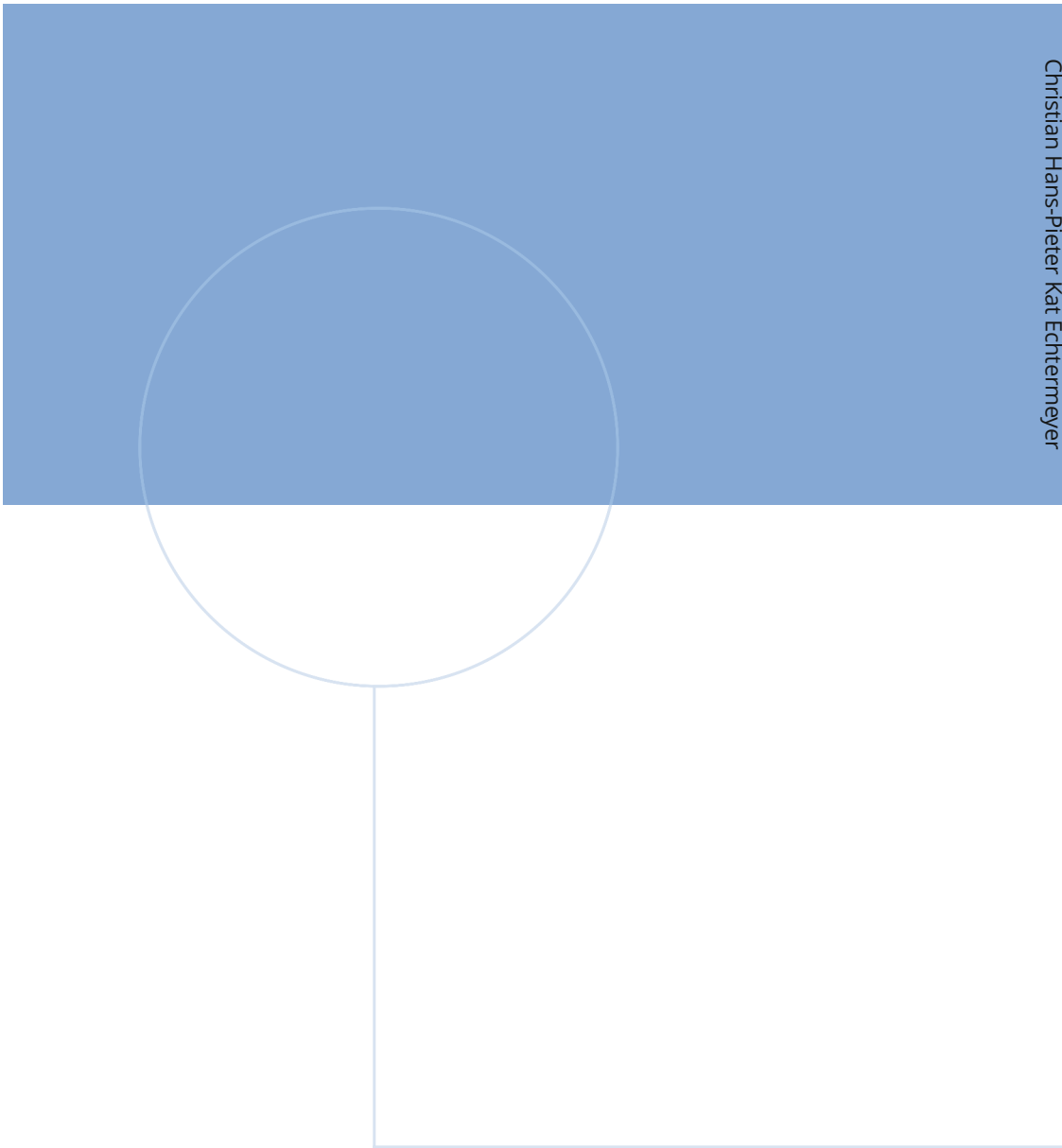


Figure A.24: The 20 best matches for an incorrectly matched identity. The x-axis shows identities, and the y-axis shows match score distance.



Christian Hans-Pieter Kat Echtermeyer

NTNU
Norwegian University of
Science and Technology