About Attrasoft ImageFinder

Attrasoft ImageFinder looks at a jpg image(s) or gif image(s) and locates similar images**. The ImageFinder can be used for:**

- **Image Verification (1:1 Matching);**
- **Image Identification (1:N Matching); and**
- **Image Search or Retrieval (1:N Matching).**

Note: Other image formats can be accommodated via customization**.**

Software Requirements

Software Requirements:

(1) Windows .Net Framework 1.1.
(2) J# .Net Redistributable.
(3) Internet Explorer.

(1) To get the latest version .Net Framework 1.1, use Internet Explorer, then click

"Tools\Windows Update".

(2) To get J# .Net Redistributable, either get it from this Microsoft site directly:

http://www.microsoft.com/downloads/details.aspx?familyid=E3CF70A9-84CA-4FEA-9E7D-7D674D2C7CA1&displaylang=en

or get it from this CD with the following path: CD:\vjredist.exe. Please install it by double clicking it.

Install the Software

1. If you have not done so, go to Internet Explorer, then click "Tools\Windows Update" to download Windows .Net Framework 1.1.

2. Click CD:\vjredist.exe to install Microsoft J# .Net Redistributable.

3. Click CD:\setup.exe to install the **ImageFinder.**

Information and Support

Attrasoft ImageFinder
Attrasoft
P. O. Box 13051
Savannah, GA. 31406
USA

http://attrasoft.com

imagefinder@attrasoft.com (Mail Subject: Attrasoft)

Phone: (912) 484-1717; (912) 897-1717

Statement of Copyright Restriction

Attrasoft has copyrighted the Attrasoft program you have purchased, and your rights of ownership are subject to the limitations and restrictions imposed by the copyright laws outlined below.

It is against the law to copy, reproduce or transmit (including, without limitation, electronic transmission over any network) any part of the program except as permitted by the copyright act of the United States (title 17, United States code). However, you are permitted by law to write the contents of the program into the machine memory of your computer so the program may be executed. You are also permitted by law to make a back-up copy of the program subject to the following restrictions:

- Each back-up copy must be treated in the same way as the original copy purchased from Attrasoft;
- **No copy (original, or back-up) may be used while another copy, (original, or back-up) is in use;**
- If you ever sell or give away the original copy of the program, all back-up copies must also be sold or given to the same person, or destroyed;

No commercial use of this software is permitted. This software is for personal use only.

Table of Contents

1. Introduction

The Attrasoft ImageFinder looks at a sample image or several images and will match all similar images from a directory or a file.

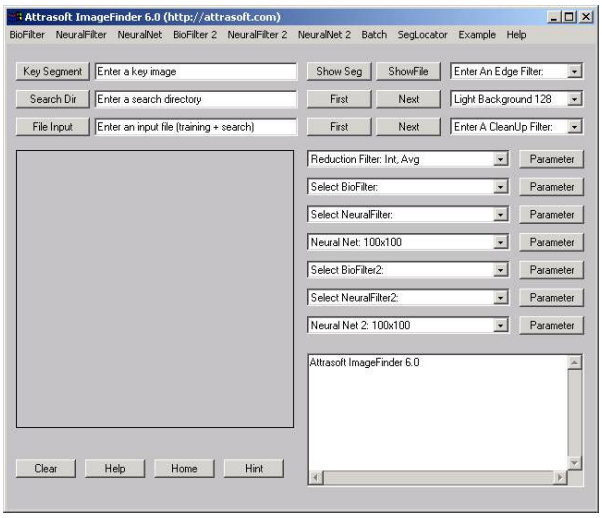


Figure 1.1 ImageFinder.

Attrasoft provides **Universal Image Identification and Retrieval Technology**, which is :

- **Accurate (up to 99.9%);**
- **Flexible** (**Can be used for ANY image type)** Stamps, License Plates, Face, Document images, Product Labels, Infrared, X-ray, color photos, Video Recognition, Fingerprints / Palm prints, Microscopic, Disease Agents, Chemical Forensic Signatures, Medical Imaging, Medication, Dental Images, Trademarks / Logos, Marketing Ads, Industrial Quality Control images, Auto Parts, Cars / Trucks / Ships / Tanks, Animals, Oil vs. Seawater, Crystals**,** …);
- **Scalable** (up to millions of images in a database);
- **Fast** (Has Real-Time Image Recognition Capability for image-based matching: with preprocessed templates, the template matching runs at a speed of 250,000 matches per second); and
- Accommodates user-defined solutions (problems where the solution does not yet exist).

Image Recognition technology will increasingly replace human labor in the form of:

- Creating new products; or
- Improving existing products/processes to increase productivity,

which will translate into saving man-hours, saving money, making more money with fewer resources, doing new things, and/or shorten service turnaround time.

When the **Attrasoft ImageFinder** learns an image(s) and goes to a directory/file to retrieve similar images, it does not deal with keywords. This software learns the **content of an image or several images** directly from the image(s) and retrieves all similar images based on the content. The **Attrasoft ImageFinder** provides the users with **a tool for image matching.**

The central task in any image data management system is to retrieve images that meet some specified constraints. The **Attrasoft ImageFinder** provides users with **a tool for content-based image retrieval.**

Applications are limited only by your imagination:

Biometrics
Content-Based Advertisement
Statistics Collection (Advertisement Statistics, …)
Internet Audio-Visual Search Engine
Satellite Image Recognition (defense)
Cancer Detection (medical)
Fingerprints, Palm Prints, Face Recognition (law enforcement)
Content-base Image Retrieval (digital library)
Space Image Recognition (space exploration)
Object Detection (military)
Face Recognition, Fingerprints, Palm Prints (security locks & systems)
Stamp Recognition (post office)
Trademark Search
Real Time Event Detection
Forensic Identifications

1.1 What will the ImageFinder do?

Attrasoft ImageFinder can match images (jpg or gif) which:

- look like an image (called key-image) or a segment of an image (called key-segment); or
- look like several key-images or key-segments.

Key-images, or key-segments are used to tell the **ImageFinder** what to look for. This is called training. After training, the **ImageFinder** is ready to retrieve all similar images. This includes all:

- Translated segments;
- Rotated segments;
- Scaled segments;
- Rotated and Scaled segments;
- Brighter or Darker segments;
- ...
- All of the above simultaneously.

1.2 Biometrics

Biometrics use human faces, fingerprints, voice, iris, …, to verify or to identify a person.

There are three generally accepted methods for performing human Verification or Identification. These are based on:

a. Something the user KNOWS (such as a password);
b. Something the user POSSESSES (such as a card/badge, called tokens); or
c. Something the user IS (a physical characteristic, or biometric, such as a fingerprint)

However, passwords can be compromised in many ways - they can be forgotten, written down, guessed, stolen, "cracked", or shared. Tokens, like a telephone card or credit card, can be lost, forgotten, stolen, given away, or duplicated.

Biometrics Verification or Identification can be accomplished by measurement of a unique biological or behavioral feature of the user to verify identity through automated means.

To determine if one image sample "matches" another image sample, they must be compared using a unique algorithm. Generally, the result of this comparison is a "score", indicating the degree to which a match exists. This score is then compared to a pre-set threshold to determine whether or not to declare a match. The comparison is performed using the captured image and previously stored images.

There are several types of image matching:

Verification

Verification is a one-to-one (1:1) matching of a single sample set (biometric identifier record) against another. Generally, the first sample is newly captured and the second is the enrolled identifier on file for a particular subject. In a user authentication environment, a score exceeding the threshold would return a 'match', resulting in the authentication of the user. A score below the threshold would return a 'no-match', resulting in the denial of access.

Identification

Identification is a one-to-many (1:N) matching of a single sample set against a database of samples, with no declared identity required. The single image is generally the newly captured sample and the database contains all previously enrolled samples. Scores are generated for each comparison, and an algorithm is used to determine the matching record, if any. Generally, the highest score exceeding the threshold results in identification.

Search or Retrieval

Search is similar to Identification, i.e. 1:N matching; however, the result is a set of possible matching images, not a classification. "Identification" returns a classification, while "Search" returns multiple matched images.

Classification

In the above three cases; only one class of image(s) is compared with a set of existing images. "Classification" is N:1 or N:N matching.

1.3 Software Requirements

Software Requirements:

(1) Windows .Net Framework 1.1.
(2) J# .Net Redistributable.
(3) Internet Explorer.

(1) To get the latest version .Net Framework 1.1, use Internet Explorer, then click:

"Tools\Windows Update".

(2) To get J# .Net Redistributable, either get it from this Microsoft site directly:

http://www.microsoft.com/downloads/details.aspx?familyid=E3CF70A9-84CA-4FEA-9E7D-7D674D2C7CA1&displaylang=en

or get it from this CD with the following path: CD:\vjredist.exe. Please install it by double clicking it.

1.4 Customized Software

There are hundreds of parameters in the software. The fine-tuning of these hundreds of parameters via Customization is Attrasoft's expertise. This software has several examples of customized software:

(1) FaceFinder;
(2) DocumentFinder;
(3) ImageExaminer.

As you will see, customized software has no parameters to adjust.

If you need a customized version of the **ImageFinder**, please contact imagefinder@attrasoft.com.

Customized versions can accommodate:

- Reducing the operation Complexity via Attrasoft tuning the parameters to one specific image type;
- Speed Optimization;
- Internal Structure Optimization
- Graphical User Interface Customization;
- Neural Network Module Customization;

- Memory Optimization (For some problems, the RAM consumption can be reduced by 80%);
- Database Interface;
- New Image Preprocessing Filters;
- Programming Library;
- Specific Symmetries or Combination of Symmetries;
- Fine Tuning of the Neural Parameters;
- Digital Image Database (Combine **ImageFinder** with Database);
- Image Formats other than jpg and gif;
- Internet Image Search Engines;
- Multi-layers of image matching;
- Web Interface (solutions that will provide users with a searchable database using a web interface);
- Other Specific Needs.

1.5 Trails

This User's Guide provides specific guidance for the software. This Guide is organized as follows:

Chapter 2 gives an overview of the software and overview of the chapters.

Part I introduces Customized Software. Part I includes:

- Chapter 3, **FaceFinder;**
- Chapter 4, **DocumentFinder**; and
- Chapter 5, **ImageExaminer**.

Part II introduces the Basic Architecture:

Image Preprocessing
    Edge Filters;
    Threshold Filters; and
    Clean Up Filters.

Normalization
    Reduction Filter.
Feature Space Recognition
    BioFilter;
    NeuralFilter.
Pixel Level (Input Space) Recognition
    ABM Filter.

Part II includes:

- Chapter 6, Image Processing;
- Chapter 7, BioFilter;
- Chapter 8, NeuralFilter;
- Chapter 9, NeuralNet Filter or ABM Filter,
- Chapter 10, Parameters; and
- Chapter 11, Batch.

Part III introduces more topics on ABM Filters. Part II includes:

- Chapter 12, ABM Filter Short-Search;
- Chapter 13, ABM Filter Long-Search;
- Chapter 14, ABM Filter Parameter Advisor;
- Chapter 15, Biometrics;
- Chapter 16, **Segment-Locator**; and
- Chapter 17, Customized **Point-Locator**.

Part IV introduces Extended Architecture. Part IV includes:

- Chapter 18, BioFilter II;
- Chapter 19, Neural Filter II;
- Chapter 20, ABM Filter II.

Part V discusses topics interested to Developers. Part V includes:

- Chapter 21, **ImageFinder Dos Version**; and
- Chapter 22, **API**.

Chapter 23 provides the Reference Manual, which describes all Commands, Filters, and Parameters.

Chapter 24 is the readme.txt file, which quickly describes software requirements, installation, and the image recognition process using the **ImageFinder**.

If you just want to learn what the **ImageFinder** can do, use trail 1. If you just

want to have a quick test of what the **ImageFinder** can do, use trail 2.

Trail 1. Evaluating ImageFinder Technology

This trail includes reading Part 1 (Chapters 3, 4, 5) in any order. You will learn what image recognition can do.

Trail 2. Learning Basics

This trails includes Part III (Chapters 6, 7, 8, 9, 10, and 11). You will learn the basic operations of the **ImageFinder.**

Trail 3. Learning the ImageFinder.

This trails includes Part III, Part IV, and Part V. You will learn more details of operations and the multi-layered approach to image recognition.

2. ImageFinder Overview

2.1 ImageFinder Internal Structures

ImageFinder adopts a layered approach to image recognition. An image recognition application is roughly divided into:

Level 5: User Interface;
Level 4: Data Management;
Level 3: Image-Matching Layer;
Level 2: Scanner, Camera, and their Drivers;
Level 1: PC with Windows.

The reason for the above approach is that the **ImageFinder** library will be integrated into a database application. Stand-alone software is roughly divided into 3 layers:

Level 3: User Interface;
Level 2: Image-Matching Layer;
Level 1: PC with Windows.

For the **ImageFinder**, the Image-Matching Layer is divided into:

Image Preprocessing
Normalization
Feature Recognition
Pixel Recognition
Sub-Pixel Recognition

Each of the sub-layers is further divided into filters:

Image Preprocessing
Edge Filters;
Threshold Filters; and
Clean Up Filters.

Normalization
Reduction Filter.

Feature Recognition
BioFilter;
NeuralFilter.

Pixel Level Recognition
ABM Filter.

Multi-layered Pixel Recognition
BioFilter 2;
NeuralFilter 2;
ABM Filter 2.

…

The **ImageFinder** uses the following GUI (Graphical User Interface) rules:

- The "Choice" button or "Drop Down List" represents all filters and only the filters.
- Parameters for each filter will be in a dialog box opened by clicking the "Parameter" button next to the filter "Choice" button;
- All commands are menu items.

An image will flow through the **ImageFinder** via the above 10 filters; however, in a typical application, the image can also bypass many of the filters.

The combination of the 10 filters will have thousands of free parameters to adjust. The **ImageFinder** has only about 70 parameters available for adjusting by users.

2.2 Image Preprocessing

The image preprocessing sub-layer prepares the image for the **ImageFinder**. The image pre-processing process is not unique; there are many options available. Some are better than others. You should choose an image pre-processing procedure **where the sample objects stand out, otherwise change the options**. If you do not have a good pre-processing setting in the off-the-shelf **ImageFinder**, **a customized filter has to be**

built. Do not make too many things stand out, i.e. as long as the area of interest stands out, the rest should show as little as possible.

Image pre-processing can determine the recognition rates. For many problems like fingerprints, palm prints, …, a special image processing procedure will be required.

The Image pre-processing layer consists of three types of filters:

Edge Filters (Optional);
Threshold Filters (Required); and
Clean Up Filters (Optional).

The **ImageFinder** applies these three filters in the above order.

The Edge Filters attempt to exaggerate the main features a user is looking for. The Threshold Filters attempt to suppress the background. The Clean-Up Filters will smooth the resulting image to reduce recognition error.

If you are not familiar with image processing, please try the following:

Setting 1:

Edge Filters: Sobel 1 (or Sobel 2)
Threshold Filters: Dark Background 128
Clean Up Filter: Medium.

Setting 2:

Edge Filters: None ("Enter An Edge Filter")
Threshold Filters: Light Background 128
Clean Up Filter: None ("Enter A CleanUp Filter")

Beyond that, please use the trial and error approach to exaggerate the main features you are looking for and to suppress the background.

2.3 Normalization

The Normalization sub-layer will prepare the images for the underlying image matching engine. The Attrasoft Image Matching Engine is an internally developed algorithm, which is called "Attrasoft Boltzmann Machine" or ABM. The ABM neural net deployed in the **ImageFinder**, by default, is a 100x100 array of neurons.

While any size of ABM neural net can be used, when coming to a particular application, a decision has to be made. The **ImageFinder** uses 6 different sizes:

- 10,000 neurons,
- 8,100 neurons,
- 6,400 neurons,
- 4,900 neurons, or
- 2,500 neurons.

Later in the multi-layered design, the number of neurons can be much larger. The Reduction Filter will connect the images to various sets of ABM neural networks.

2.4 Feature Recognition

Before the ABM engine processes an image, it will go through a pre-matching step called Feature Space Matching or Feature Matching. The purpose of Feature Matching is to eliminate unmatched images. This Feature Matching sub-layer has two filters:

BioFilter; and
NeuralFilter.

A typical image has a dimension of say 480x640, or 300,000 pixels. A Feature Space of an image is a collection of variables

computed from a given image. The number of variables in a feature space is usually less than 1% of the number of pixels. As a result, the matching in the feature space is much faster than pixel matching.

For example, after FFT, (Fast Fourier Transform), the first few terms will be variables in a feature space. (The Fourier Transform is a way to convert Space-domain data into its frequency components).

The BioFilter will attempt to eliminate 80% of the mismatches, and the Neural Filter will attempt to eliminate 19% more of the mismatches, leaving only 1% for the slower matching.

2.5 ABM Matching Engine

The ABM neural network is the matching engine of the **ImageFinder**. There is one filter in this sub-layer, the NeuralNet filter or ABM Filter. The ABM matching engine is responsible for all of the Attrasoft products. The ABM has been applied to many fields of pattern and image recognition. Each time a customer is served, their feedback polishes this core technology. This core technology has been tested over and over again since 1995.

2.6 Multi-Layered ABM

Once we go beyond one layer of the neural network, there are endless choices. An application will normally dictate the selection of a multi-layer ABM architecture. In this version, an example of multi-layered ABM architecture will be used, which is based on the **ImageExaminer**. Depending on the arrangement, this sub-layer can have many filters. In this version, this sub-layer has 3 additional filters:

BioFilter 2
NeuralFilter 2
ABM Filter 2

2.7 Chapter Overview

Part I. Customized ImageFinder

In this version, the number of parameters is increased; the number of filters is increased; and the number of commands is increased. As a result, this version is significantly harder to use than the earlier version. For this reason, we will present three customized examples first.

Customized versions are easy to use, usually a few clicks, so you can see once the tuning is over, the operation of the software is very easy.

Chapter 3, **FaceFinder**, presents a customized example for the front view facial images in the Feret database.

Chapter 4, **DocumentFinder**, presents a customized example for scanned document images.

Chapter 5, **ImageExaminer**, presents a customized software for comparing two basically identical images and identifying some minor differences. There are three chapters later that are related to this customized software, BioFilter II, Neural Filter II, and ABM Filter II.

Part II. Basic Architecture.

The Basic Architecture is a one-layer neural network architecture:

Image Preprocessing
    Edge Filters;
    Threshold Filters; and
    Clean Up Filters.
Normalization
    Reduction Filter.

Feature Recognition
BioFilter;
NeuralFilter.
Pixel Level Recognition
ABM Filter.

The **ImageFinder** recognizes images in two phases:

- Feature Space Matching
- Input Space Matching

The purpose of Feature Space Matching is to eliminate unmatched images. This Feature Recognition sub-layer has two filters:

BioFilter; and
NeuralFilter.

The main image recognition uses the NeuralNet Filter.

Chapter 6, Image Processing, briefly describes the image processing process required for the **ImageFinder.** You have to set three filters, Edge Filters, Threshold Filters, and Clean Up Filters. The Threshold Filter is required; the other two filters are optional.

Chapter 7, BioFilter, will describe the minimum number of steps for using the BioFilter for image recognition:

- Initialization
- Converting Images to Records
- Training
- Template Matching

Initialization sets the **ImageFinder** parameters. Then, the image signatures are calculated and stored in a record. Training teaches the BioFilter who matches with whom. After training, the BioFilter can be used for 1:1 and 1:N Matching.

Chapter 8, NeuralFilter, introduces the Neural Filter. The Neural Filter is the second classification filter in the feature space; it runs parallel to the BioFilter.

Chapter 9, NeuralNet Filter or ABM Filter, introduces the neural network used in the **ImageFinder**. The NeuralNet Filter commands are divided into three types: Short, Long, and File depending on the input. Short-Search processes images in one directory; Long-Search processes images in sub-directories; and File-Search processes images in input files. The File-Search will use the output of the Feature Space recognition as input.

Chapter 10, Parameters, gives a more detailed description of all parameters used in various filters.

Chapter 11, Batch, introduces the batch command, which allows you to save your setting and execute your problem in a few clicks. The batch file can also be used in the dos version.

Part III. More on NeuralNet Filters

Chapter 12, ABM Filter Short-Search, introduces Short-Search, which will search images in a directory.

Chapter 13, ABM Filter Long-Search, introduces Long-Search, which will search images in sub-directories.

Chapter 14, ABM Filter Parameter Advisor, introduces “Parameter Advisor”, which will help you to select the neural net parameters.

Chapter 15, Biometrics, introduces a fingerprint recognition example.

Chapter 16, **Segment-Locator**, introduces several commands, and examples that will locate a segment in images.

Chapter 17, **Point-Locator**, introduces several commands, and examples that will locate points in images.

Part IV. Extended Architecture

When you go beyond the basic architecture, the options are limitless.

We will focus our discussion on a specific application: to compare two basically identical images and identify the minor differences.

Our approach is to divide the image into smaller sections and use our basic architecture for each smaller section of the image. This will generate the following additional layers:

BioFilter II;
NeuralFilter II;
ABM Filter II.

Chapter 18, BioFilter II, introduces BioFilter II, which is similar to the BioFilter but operates on an image segment rather than the whole image.

Chapter 19, Neural Filter II, introduces Neural Filter II, which is similar to the Neural Filter but operates on an image segment rather than the whole image.

Chapter 20, ABM Filter II, introduces ABM Filter II, which is similar to the ABM Filter but operates on an image segment rather than the whole image.

Part V. Developers

Software developers can integrate the **ImageFinder** technology into their applications. The **ImageFinder Dos Version** is a quick and easy way.

Chapter 21, **ImageFinder Dos Version**, introduces a quick way to implement an image recognition application.

Chapter 22, API, introduces the standard **Application Programming Interface (API)** of the **ImageFinder** for software developers. The image recognition level library is called **TransApplet.**

Chapter 23, Reference Manual, lists and explains each of the filters, commands, and parameters.

Chapter 24, ImageFinder Support, lists the support packages.

Chapter 25, Readme.txt, lists the contents of the readme.txt file.

3. FaceFinder

This is an example of a customized version of the **ImageFinder**, applied to the Feret Facial Image Database. All parameters are eliminated.

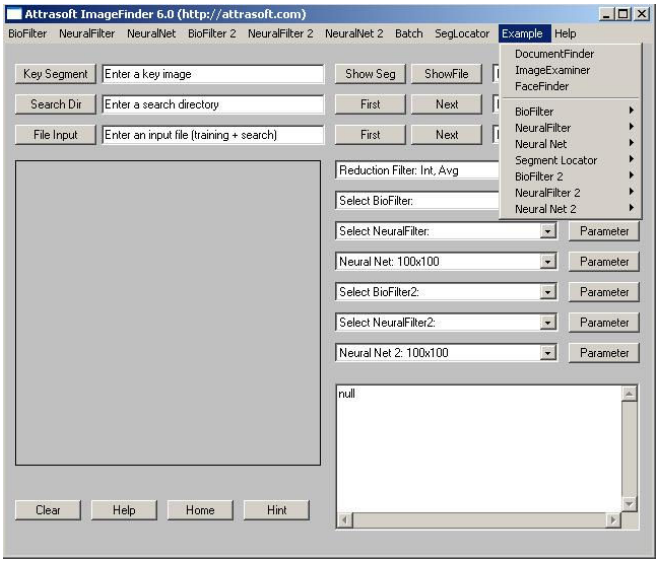


Figure 3.1 Example Menu.

To start the FaceFinder in the ImageFinder, click menu "Example/FaceFinder".

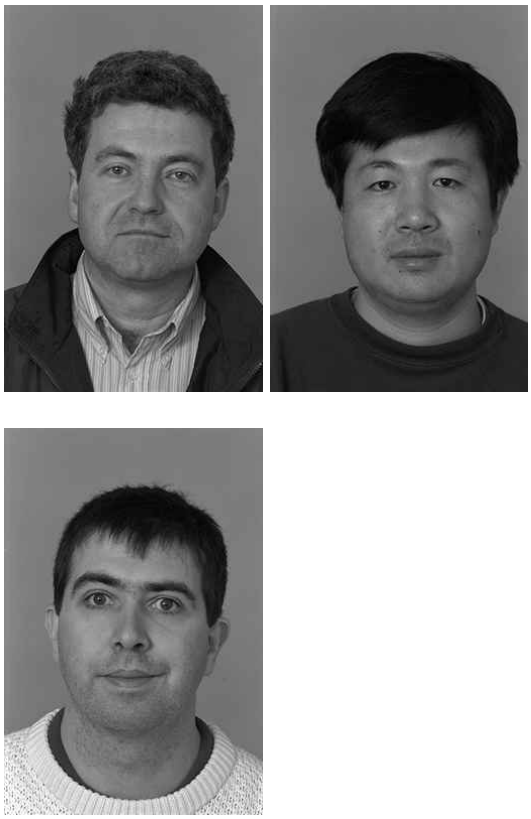


Figure 3.2 The Feret Database.

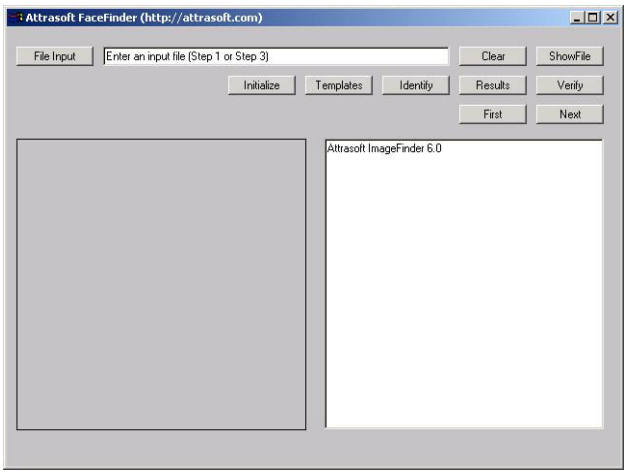


Figure 3.3 The **FaceFinder.**

There is only one example in the **FaceFinder**, with following results:

Total Images N = 1,032 = 516 pairs
Possible Matches N*N = 1,065,024

3.1 Introduction

3.1.1 Expectation

With image matching, the face images are converted into templates first. This one-time conversion runs at the speed of several images per second. **After that, the template matching runs at a speed of higher than 100,000 matches per second and it will eliminate about 98% - 99% of unmatched images.** This means if you want to match one newly captured image against 1,000,000 existing images, in a matter of seconds, the **FaceFinder** will reduce the 1,000,000 possible matches to about 10,000 candidates. At this point, the other types of matching can further narrow down the retrieved set

Attrasoft Matches = 14,300

Actual Duplicates = 2,064
Attrasoft Found Duplicates = 2,064

Percent Eliminated = 98.7 %
= 1 – 14,300/1,065,024
Percent of Duplicates Found = 100 %
= 2,064/2,064

This example has 516 pairs of face images or 1,032 images. In the N x N Match, each image matches itself and its partner in the pair, giving a total of 2 * N = 2 * 1,032 = 2,064 positive matches.

3.1.2 FaceFinder Input

There is only one way to enter data into the **FaceFinder**, by input file. The **FaceFinder** will compare each of the images in the file to all other images in the file. Clicking the "File Input" button and selecting a file can specify the input file.

The input files list one image per line. Each line specifies an absolute path. For example,

C:\xyz1\0001.jpg
C:\xyz1\0002.jpg
C:\xyz2\0003.jpg
C:\xyz2\0004.jpg
…

There is one file for **Identification** in the software CD, "facefinder_input_id.txt", and there is one file for **Verification**, "facefinder_b1_first10.txt".

3.1.3 FaceFinder

The **FaceFinder** is an example of a customized version of the **ImageFinder**. There are many parameters in the **ImageFinder**. The **FaceFinder** has eliminated all of the parameters.

The image matching will be done in three steps:

- Training and Initialization
- Converting Images to Records
- Template Matching

Let us look at each phase.

Training/Initialization

Initialization loads the **ImageFinder** parameters. Training uses the data collected in advance. Both steps are combined together and done in a single click.

Converting Images to Records:

This step is slow (several images per second); however:

(1) This step can be done once for all; and
(2) This is linear, i.e. the time is directly proportional to the number of images.

Therefore, **this step does not have much impact on the operating speed**. If you use the default images, the records are loaded when you click the "Initialize" button.

Template Matching

The **matching speed** will be between **100,000 and 1,000,000 comparisons per second.**

3.1.4 Restrictions

The restrictions are:

Feret database;

Front view;
Distance between the Camera and the face
is fixed;
Light condition is fixed;
Image dimension is 256 x 384;
Image type is jpg.

3.2 Identification

The image Identification will be done in three steps:

- Initialization and Training
- Converting Images to Records
- Template Matching

3.2.1 Operation

The Identification procedure is:

- Initialization;
- Convert images in the input file to records and save them to a1.txt;
- Complete Template Matching and print the results to b1.txt.

Now let us look at the details.

1. Initialization and Training

Initialize and train the software by clicking "Initialize" button. This button will first initialize the software, which simply means to load the parameters; then train the software, which is similar to the **ImageFinder**. At the end of this step, you should see:

Parameters loaded!
BioFilter loaded!
…
Neural Filter loaded!
BioFilter 2 loaded!
Neural Filter 2 loaded!

2. Entering Data

Select input file by clicking the "File Input" button. In this example, the input file for the Identification example is ".\facefinder_input_id.txt". Select this file. This file has 1,032 images and the software will go through each one of them to make sure the files exist. This will take a while and at the end of this process, you should see:

Number of blocks = 1
Block 0. Length = 1032
C:\…\00019ba010_960521.jpg
C:\…\00019bj010_960521.jpg
C:\…\00029ba010_960521.jpg
C:\…\00029bj010_960521.jpg
…

Buttons:

- To see the original content of this file, click the "ShowFile" button.
- To clear the text window, click the "Clear" button.
- To see the first image in the input file, click the "First" button.
- To see the next image in the file, click the "Next" button. If you keep clicking the "Next" button, the **FaceFinder** will scan through all images.

3. Templates

If you use the default images, this step is optional because the records are loaded when you click the "Initialize" button.

Convert images in the input file to records by clicking the "Templates" button. The results are saved to a1.txt. Once the image is converted into internal templates, you do not have to

convert them again. This step is slow (several images per second); however:

- This step can be done once for all; and
- This is linear, i.e. the time is directly proportional to the number of images.

Therefore, this step does not have much impact on the operating speed.

4. Template Matching

Identify matches by clicking the "Identify" button. The **FaceFinder** will compare each of the images in the file to all other images in the file. If there are N images in the input file, there will be N*N comparisons.

5. Check the Results

Check the results of the template matching by clicking the "Results" button. The checking results will be printed to the text window; and it will indicate that the number of matches found in b1.txt will agree with the specifications in the b1_matchlist.txt. This will quickly let you know the positive identification rate. The matching information is stored in file, "facefinder_b1_matchlist.txt".

3.2.2 Template Matching Output

The template matching runs at a speed of higher than 100,000 matches per second and it will eliminate about 98% - 99% of the unmatched images.

This means if you want to match one newly captured image against 1,000,000 existing images in a matter of seconds, the **FaceFinder** will reduce the 1,000,000 possible matches to about 10,000 candidates.

The result of matching each of the 1,032 images against the 1,032 images is in the file b1.txt. At the end of the matching, b1.txt will be opened for you. The b1.txt will look like this:

C:\...\data1\00019ba010_960521.jpg
C:\...\data1\00019ba010_960521.jpg
C:\...\data1\00019bj010_960521.jpg

C:\...\data1\00019bj010_960521.jpg
C:\...\data1\00019ba010_960521.jpg
C:\...\data1\00019bj010_960521.jpg

…

C:\...\data1\01209fb010_940128.jpg
C:\...\data1\01063ba010_960521.jpg
C:\...\data1\01063bj010_960521.jpg
C:\...\data1\01209fa010_940128.jpg
C:\...\data1\01209fb010_940128.jpg

Total Number of Matches = 14,300

This file is divided into blocks separated by a blank line. Each image has its own block. A block looks like this:

C:\...\data1\00019ba010_960521.jpg
C:\...\data1\00019ba010_960521.jpg
C:\...\data1\00019bj010_960521.jpg

The first line in this block is the input for Identification, the rest of the lines are the output for Identification. In the above example, 00019ba010_960521.jpg matches with itself and 00019bj010_960521.jpg.

3.2.3 Checking Identification Results

If this is a test run, i.e. you know the correct answers; you can see the matching results in seconds. You must prepare a file, which indicates the matching pairs. To test the results in b1.txt, you must prepare B1_matchlist.txt file.

An example of b1_matchlist.txt is:

```
5
1       IMAGE00053770   IMAGE01312024
2       IMAGE00053771   IMAGE01312025
3       IMAGE00053772   IMAGE01312026
4       IMAGE00053773   IMAGE01312027
```

Line 1 is the 'Line number" in this file. The "Line Number" does not have to be in order. Each line has the following format:

Number, tab, filename, tab, filename, **tab.**

Note:

(1) **You must have a tab at the end of each line;**
(2) **The file names do not contain ".jpg".**

Once you get the two files prepared, click the "Results" button.

There are two common errors:

(1) The last Tab is missing;
(2) The number of rows is less than the first number in the file.

Check the results of the template matching by clicking the "Results" button. You will see the something like the following in the text window:

Checking Template Matching Results!

Get b1.txt...

Character = 1060577

Lines = 16365

Blocks = 1033

Get b1_matchlist.txt...

Check...

Total Matches = 2,064

The message indicates b1.txt has 1,033 blocks: the 1,032 image blocks plus the last line indicating the total number of matches retrieved. The message "Total Matches = 2,064" indicates that 2,064 matches in b1.txt agree with those in b1_matchlist.txt.

3.2.4 Analysis

There are several variables used to characterize a Verification or Identification.

For **Verification**, the common variables are:

Positive Verification Rate

= 1 - False Rejection Rate

False Rejection Rate

= 1 - Positive Verification Rate

Negative Verification Rate

= 1 - False Acceptance Rate

False Acceptance Rate

= 1 - Positive Verification Rate

The single characteristic number should be:

(1 - False Rejection Rate)

- (1 - False Rejection Rate).

For **Identification**, the above method can be used. The alternative is:

Positive Verification Rate

= 1 - False Rejection Rate

False Rejection Rate

= 1 - Positive Verification Rate

Elimination Rate

= #Retrieved / #Possible Matches

Hit Ratio = # Correct Match / #Retrieved

The single characteristic number should be:

Positive Verification Rate

* Elimination Rate * Hit Ratio

There are several numbers that will be required for analysis:

Possible Matches

Let the Total Images in the input file be N; the Possible Matches will be N*N.

Attrasoft Matches

The number of retrieved matches is listed in the last line of b1.txt. Go to

the end of b1.txt, you will see something like this:

Total Number of Matches = 14,300

Actual Match

This number depends on your problem. It should be the first number in b1_matchlist.txt.

Attrasoft Found Duplicates

Click the "Results" button to get this number, as discussed in the last section.

Now you have all of the numbers, you can make an analysis.

Positive Identification

Positive Identification Rate = the results of clicking the "Results" button divided by the first number in file, b1_matchlist.txt.

Elimination Rate

The Elimination Rate is 1 minus the number at the end of b1.txt divided by the number of possible matches.

Hit Ratio

The Hit Ratio is the number indicated by the "Results" button divided by the number at the end of b1.txt.

Composite Index

Finally, an Identification is measured by the multiplication of **Positive Identification * Elimination Rate * Hit Ratio.**

3.2.5 Examples

(1) Initialization and Training

Click the "Initialize" button; you should see:

Parameters loaded!
BioFilter loaded!
…
Neural Filter loaded!
BioFilter 2 loaded!
Neural Filter 2 loaded!

(2) Select Input File

Click the "File Input" button, select "facefinder_b1_matchlist.txt"; wait for a while, and you will see:

Number of blocks = 1
Block 0. Length = 1032
C:\...\data1\00019ba010_960521.jpg
C:\...\data1\00019bj010_960521.jpg
C:\...\data1\00029ba010_960521.jpg
C:\...\data1\00029bj010_960521.jpg…

(3) Template

This step is optional if the default images are used. Click the "Templates" button and wait until the computer calms down. You should see each file name in the input file is listed, indicating the speed of converting images into templates.

(4) Template Matching

Click the "Identify" button and wait a few seconds until the b1.txt is opened.

(5) Check the Results

Click the "Results" button and see the message of 2,064 matches.

(6) Analysis

Total Images N = 1,032 = 516 pairs
Possible Matches N*N = 1,065,024
Attrasoft Matches = 14,300

Actual Duplicates = 2,064
Attrasoft Found Duplicates = 2,064

Percent Eliminated = 98.7 %
= 1 – 14,300/1,065,024
Percent Duplicate Found = 100 %
= 2,064/2,064

To Summarize,

- **Start the software;**
- **Click the "Initialize" button;**
- **Click the "Template" button (optional);**
- **Click the "Identify" button;**
- **Click the "Results" button.**

3.2.6 Identify with the Template File

Your images were already converted into a template file earlier; you can run it without image conversion. Simply save the template file to c:\...\a1.txt and:

- Start the software;
- Click the "Initialize" button;
- Click the "Identify" button;
- Click the "Results" button.

3.3 Verification

The **FaceFinder** can also perform image Verification, but it is much slower (matching several images per second, rather than matching several hundred thousand images per second).

Verification requires multiple images or templates stored for each ID, say 5 images. The example used here has 516 pairs, i.e. each ID only has only one previous stored image.

The Verification procedure is exactly the same as the Identification procedure except at the end of the Identification procedure; one more step is added.

FaceFinder-Verification requires the FaceFinder-Identification as a preprocessing step.

FaceFinder-Identification will eliminate 98 % - 99 % of the unmatched images and the Verification procedure will take care of the rest.

The output of the Identification procedure, b1.txt, will be used as input for the Verification procedure.

3.3.1 Operation

After Identification, there is one more step for Verification:

Entering data
Click the "File Input" button and select an input file.

Verification
Click the "Verify" button.

3.3.2 Example

There is one example in the **FaceFinder**. The input file is "b1_first10.txt", which lists the first 10 blocks in the b1.txt.

The steps are:

- Click the "File Input" button and select "facefinder_b1_first10.txt";
- Click the "Verify" button.

At the end of the Verification procedure, a web page is opened.

- 9 out of the 10 images are verified;
- The other one is a false rejection because there is only one previously stored image rather than 5 previously stored images;
- There is no false acceptance.

4. DocumentFinder

The **DocumentFinder** is a customized example of the **ImageFinder**. It brings the number of parameters from hundreds to two and makes it easier to use.

To start the **DocumentFinde**r, click menu item "Example/DocumentFinder".

Recent development in scanner technology has made it very easy to convert paper documents into digital documents. A $1000 scanner, for example Fujitsu 4120c, can scan and save 50 pages in a single click. The scanner creates image names via some auto-numbers you have specified. More expensive scanners can scan and save 1,000 pages in a single click.

The central task in any image data management system is to retrieve images that meet some specified constraints. This software attempts to solve a particular problem, retrieve document images similar to a given document image.

Assume you have a million pages of documents already converted into digital form, and you want to retrieve documents that meet some specified constraints. A typical document retrieval system should have several components:

1. Text;
2. Image;
3. 1-D barcode; and
4. 2-D barcode.

Each component addresses a particular area of retrieval and their functions generally do not overlap. A complete solution should use all of the above options. This software deals with the image matching only.

4.1 Why Image Matching?

Image matching deals with several particular problems, which cannot be addressed by text search, 1D barcode search, and 2D barcode search:

(1) **Tables**: image matching is responsible for retrieving documents with similar tables.

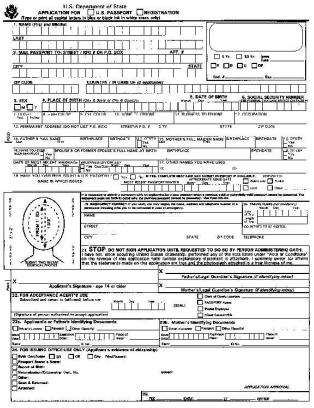


Figure 4.1 Documents with tables.

(2) **Images**: image matching is responsible for retrieving documents with similar images.

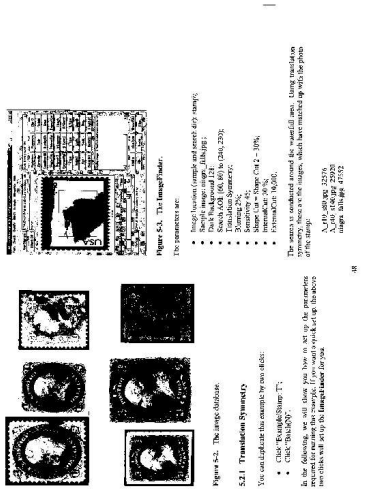


Figure 4.2 Documents with images.

(3) **Special Symbols**: image matching is responsible for retrieving documents with similar Special Symbols.

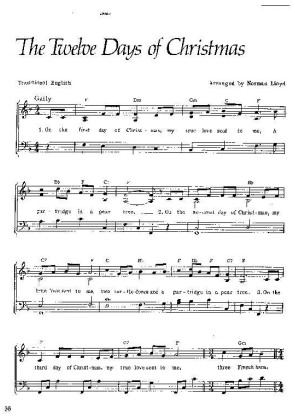


Figure 4.3 Documents with special symbols.

(4) **Figures, Drawing, Designs**: image matching is responsible for retrieving documents with similar Figures, Drawing, and Designs.

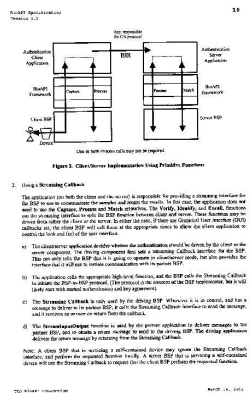


Figure 4.4 Documents with Figures, Drawing, and Designs.

(5) **Maps:** image matching is responsible for retrieving documents with similar Maps.

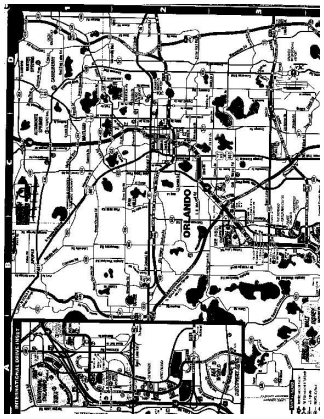


Figure 4.5 Documents with maps.

(6) **Hand Written Notes**: image matching is responsible for retrieving documents with hand written notes.

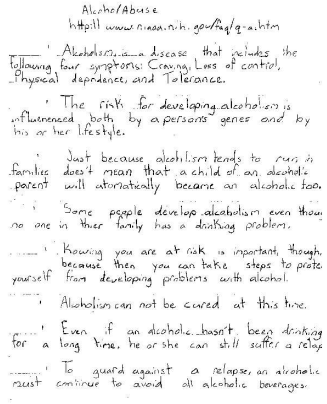


Figure 4.6 Documents with handwritten notes.

(7) …

4.1.1 Expectations

With image matching, the document images are converted into templates first. This one-time conversion runs at the speed of several images per second. **After that, the template matching runs at a speed of higher than 100,000 matches per second and it will eliminate about 98% - 99% of the unmatched images.** This means if you want to match one newly captured image against 1,000,000 existing images, in a matter of seconds, the **DocumentFinder** will reduce the 1,000,000 possible matches to about 10,000 candidates. At this point, the other types of matching can further narrow down the retrieved set.

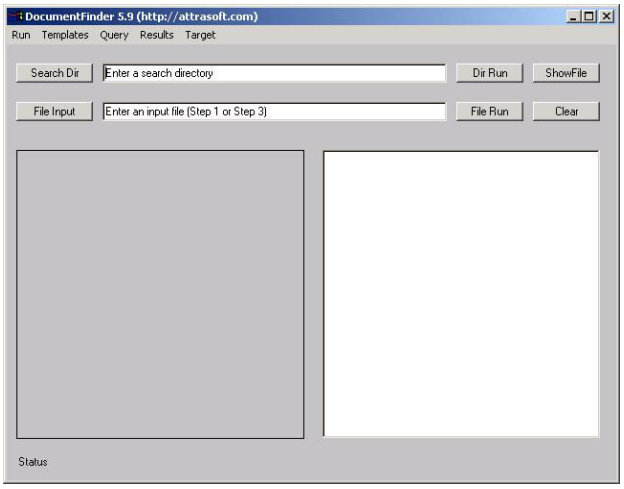


Figure 4.7 The **DocumentFinder.**

The final example in this software has the following results:

Total Images N = 1,023
Possible Matches N*N = 1,046,529
Attrasoft Matches = 4,828

Actual Duplicates = 2,883
Attrasoft Found Duplicates = 2,863

Percent Eliminated = 99.5 %
= 1 – 4,828/1,046,529
Percent of Duplicates Found = 99.3 % = 2,863/2,883

4.2 Run DocumentFinder

There are two ways to enter data into the **DocumentFinder**:

- Directory
- File

In the case of directory input, the **DocumentFinder** will compare each of the images in the directory to all images in the directory.

In the case of file input, the **DocumentFinder** will compare each of the images in the file to all images in the file.

The **DocumentFinder** can be operated in two ways:

- **Simple Matching (2 clicks); or**
- **Three-Step Matching.**

Three-Step Matching offers more options.

4.2.1 Input

- Clicking the "Search Dir" button and selecting any file in that directory can specify the input directory.
- Clicking the "File Input" button and selecting a file can specify the input file.

The input files list one image per line. Each line specifies an absolute path. For example,

C:\xyz1\0001.jpg
C:\xyz1\0002.jpg
C:\xyz2\0003.jpg
C:\xyz2\0004.jpg
…

There are three sample input files in the software CD, "df_input_misc.txt", "df_input_bioapi.txt", and "df_input_abm54.txt".

4.2.2 Training

The **DocumentFinder** is an example of **ImageFinder** customization. There are many parameters in the **ImageFinder**. The **DocumentFinder** has eliminated all parameters except two. If you choose the Simple Matching (2 clicks), there will be no parameters. If you choose the Three-Step Matching, there are two parameters.

The first parameter has three choices:

- 98% Setting (Default)
- 97% Setting
- 96% Setting

Clicking "Template/98% Setting", or "Template/97% Setting", or "Template/96% Setting" can set the parameter. The 98% Setting will retrieve about 98% of the matches; and the 96% Setting will retrieve about 96% of the matches.

The second parameter is the "Training" button:

- Training
- Training (High Accuracy)
- Training (Low Accuracy)
- Training (Lower Accuracy)

The output of "Training (High Accuracy)" is half of that of "Training"; the output of "Training" is half of that of "Training (Low Accuracy)"; the output of "Training (Low Accuracy)" is half of that of "Training (Lower Accuracy)".

The combination of "96 Setting" and "Training (High Accuracy)" can be used to quickly probe a set of images to find duplicates.

4.2.3 Matching

The image matching will be done in three steps:

- Converting Images to Records
- Training
- Template Matching

Let us look at each phase.

Converting Images to Records:

It is very important that the memory (RAM) used is less than the size of the physical RAM, i.e. avoid using virtual memory, otherwise the procedure will be slower.

This step is slow (several images per second); however:

- This step can be done once for all; and
- This is linear, i.e. the time is directly proportional to the number of images, provided the memory used is less than the physical RAM (otherwise, the problem should be done in piecemeal).

Therefore, this step does not have much impact on the operating speed.

Training

Training uses the data collected in advance. This step will take about 1 minute. Once the software is trained, it can make multiple matches until you close the software or choose a different setting.

Template Matching

Each record will take roughly 0.13K. For example, 1 million records will take 130 MB.

The matching speed will be between 100,000 – 1,000,000 comparisons per second.

4.3 Restrictions

The data used in this software are obtained as follows:

Document Scanning Dimension: 8.5 inch
x 10.66 inches.
Scanning Resolution: 70 pixels per inch.
Image Mode: Black and White.
Image File Type: 24-bit Jpeg.
Resulting Image Dimension: 576 x 746.
Scanner Used: Fujitsu 4120c.

The restrictions are:

Document Scanning Dimension: 8.5 inch
x 10.66 inches.
Scanning Resolution: 70 pixels per inch.
Image File Type: 24-bit jpeg image.

Other settings might require a different set of internal parameters.

The reason to choose 70 pixels per inch is that a typical jpeg image will be about 100K, which is neither too large nor too small.

4.4 Examples

There are four examples in the software.

Name	Location	# of Images
Misc	.\misc\	93 x 2
BioAPI	.\bioapi\	119 x 3
Abm54	.\abm54\	160 x 3
All		1,023 images

The "Misc" example has 93 matching pairs, or 186 images. It has various types of documents such as Tables, Images, Music, Figures, Drawings, Designs, Maps, …. By default, the template file for this example is a2.txt; the result file is b2.txt, and the match-list file is b2_matchlist.txt.

The "BioAPI" example has 119 matching triplets, or 357 images. BioAPI is a document, which attempts to set up a standard for the templates. This document has 119 pages. By default, the template file for this example is a3.txt; the result file is b3.txt, and the match-list file is b3_matchlist.txt.

The "Abm54" example has 160 triplets, or 480 images. By default, the template file for this example is a4.txt; the result file is b4.txt, and the match-list file is b4_matchlist.txt.

The last example is the combination of all three examples. This one has 1,023 images. By default, the template file for this example is a5.txt; the result file is b5.txt, and the match-list file is b5_matchlist.txt.

4.5 Simple Matching

The Simple Match uses "Run" Menu. The image matching will be done in three steps:

- Converting Images to Records
- Training
- Template Matching

Simple Matching combines all three steps into a single click. Note that step 1, Converting Images to Records, can be done once for all, so the Simple Matching should be used only once for each new image set.

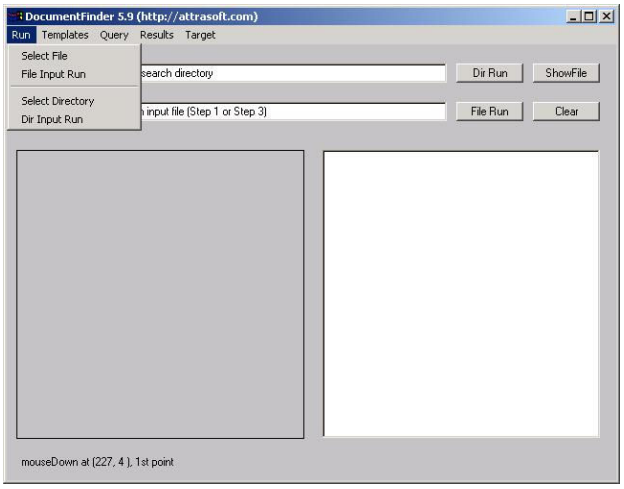


Figure 4.8 Run Menu.

After that, Three-Step Matching should be used to increase the computation speed.

4.5.1 Operation

Operation Using File Input:

- **Select input file by clicking "Run/Select File";**
- **Run by clicking "Run/File Input Run".**

Operation Using Dir Input:

- **Select input file by clicking "Run/Select Directory";**
- **Run by clicking "Run/Dir Input Run".**

The Input File should list one image per line using the absolute path (directory path + image name). For example,

C:\xyz1\0001.jpg
C:\xyz1\0002.jpg
C:\xyz2\0003.jpg
C:\xyz2\0004.jpg
…

The Simple Match will:

- Convert images in the Input File or Input Directory to records and save them to a1.txt;
- Training from sample.txt and sampletruth.txt;
- Complete Template Matching;
- Print the results to b1.txt.

4.5.2 File Input Examples

In this section, we will use the "Misc" example, which has 93 matching pairs, or 186 images. It has various types of documents like Tables, Images, Music, Figures, Drawings, Designs, Maps, ….

(1) Select the input file by clicking "Run/Select File".

To select the input file, click "Run/Select File", and select ".\df_input_misc.txt". Click "ShowFile" button to display the content of this file:

.\misc\misc_a_0001.jpg
.\misc\misc_a_0002.jpg
.\misc\misc_a_0003.jpg
…

(2) Run by clicking "Run/File Input Run".

You should first observe:

(a) The images are translated into templates;
(b) The **DocumentFinder** is trained;
(c) Image Matching.

The number of comparisons is N*(N-1)/2, or 186 * (186 –1)/2 = 17,205. **The 17,205 comparisons take a split second**. The results are in a file, b1.txt, which should be opened at this time.

(3) Results

B1.txt will look like this:

.\misc\misc_a_0001.jpg
.\misc\misc_b_0001.jpg

.\misc\misc_a_0002.jpg
.\misc\misc_b_0002.jpg
…

The first block means that misc_a_0001.jpg is compared with the other 185 images specified by the input file and one match is found, which is misc_b_0001.jpg. This is correct.

Because there are 186 images in the input file, you should have 186 blocks in the output file. Note that this is N*(N-1)/2 comparison, not N*N comparison; therefore, misc_a_0001.jpg will match with misc_b_0001.jpg; but misc_b_0001.jpg will not match with misc_a_0001.jpg, because it will never meet with misc_a_0001.jpg.

The **DocumentFinder** does have the N*N matching command, "Target/a1 + a1 ➔ b1", but we will introduce this command later.

(4) Analysis

Scroll all the way to the end of this file; the last line is:

Total Number of Matches = 139

i.e. the **DocumentFinder** found 139 matches. All 93 pairs are identified. The results are:

Total Images N = 186
Possible Matches N*(N-1)/2 = 17,205
Attrasoft Matches = 139

Actual Duplicates = 93
Attrasoft Found Duplicates = 93

Percent Eliminated = 99.2 %
= 1 - 139/17,205
Percent of Duplicates Found = 100 %
= 93/93

4.5.3 Dir Input Examples

The procedure using Dir Input is:

- Select input file by clicking "Run/Select Directory" and select any file in the directory, .\misc\;
- Run by clicking "Run/Dir Input Run".

You should get similar results like the last section, except the order of appearance.

4.6 Three-Step Matching

Besides the Simple-Match, the software also supports matching step by step. The Three-Step Matching will be used most often because you do not need to convert the images to templates each time.

In this chapter, we will show how to match in three steps. Please remember, **Step 1**, Converting Images to Template Records, **can be done once for all,** so you do not have to do step 1 in each run.

4.6.1 Matching Procedure

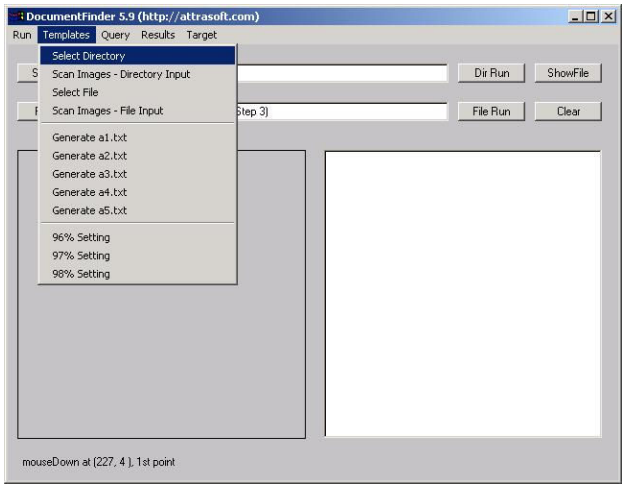


Figure 4.9 Template Menu is used to convert images to templates.

Step 1. Converting Images to Records:

- Click “Templates/Select File” and select a file;
- Click “Templates/Scan File (File Input)”;
- Click “Templates/Generate a1.txt”;
- Click “a1.txt” to see the records generated using Windows Explorer.

Note that the output is saved into a1.txt. You also have the option to save it to a2.txt, a3.tx, a4.txt and a5.txt.

Step 2. Training:

- Click “Query/Training”.

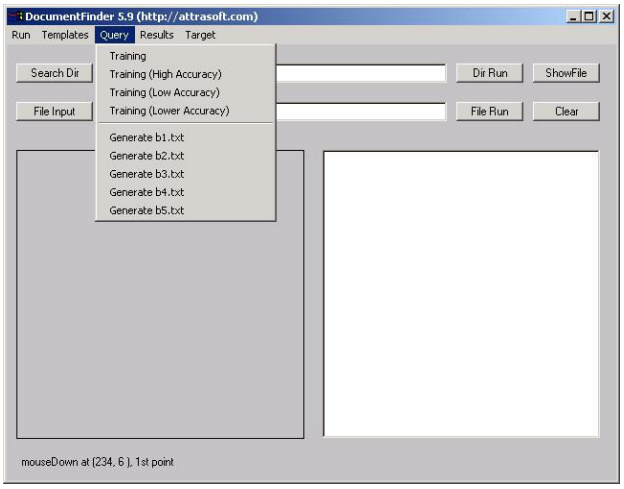


Figure 4.10 Query Menu for Training and Template Matching.

Note that in the next computation, as long as you do not close the program, you do not need to train again.

Step 3. Matching

- Click “Query/Generate b1.txt”.

Note that “b1.txt” is generated from “a1.txt”. You also make the following substitutions in the above commands:

A2.txt ==> b2.txt
A3.txt ==> b3.txt
A4.txt ==> b4.txt
A5.txt ==> b5.txt

The effect of these three steps is the same as the Simple Matching. Once the images are converted into records, you should not do Step 1, converting images to templates, over and over again. The reason for this Three-Step approach is to save time; you will convert images to records only once. The converted records are saved into a1.txt (or a2.txt, a3.txt, a4.txt, or a5.txt, depending on the command used). If the same images are used, you do not have to convert them again the second time.

Also, as long as you do not close the software, you do not need to train each time; it is calculated once for all.

4.6.2 Example “BioAPI”

In this section, we will use the “BioAPI” example, which has 119 matching triplets, or 357 images. BioAPI is a document, which attempts to set up a standard for the templates. This document has 119 pages.

Select input file by clicking “Templates/Select File”

Click “Templates/Select File”, and select “.\df_input_bioapi.txt”;

Click the “ShowFile” button to display the content of this file:

.\bioapi\bioapi_a_0001.jpg
.\bioapi\bioapi_a_0002.jpg
.\bioapi\bioapi_a_0003.jpg
…

(2) Convert images into templates by clicking “Templates/Scan Images – File Input”.

You should first observe that the images are translated into templates. After that, you must save the templates into 1 of 5 files:

- **Click "Templates/Generate a1.txt";**
- **Click "a1.txt" to see the records generated.**

(3) Training

To train the **DocumentFinder**, click "Query/Training".

(4) Match

Click "Query/Generate b1.txt".

Note that, "b1.txt" is generated from "a1.txt". You also make the following substitutions in the above commands:

A2.txt ==> b2.txt
A3.txt ==> b3.txt
A4.txt ==> b4.txt
A5.txt ==> b5.txt

The number of images is N = 357 images. The number of comparisons is N (N-1)/2, or 357 * (357 –1)/2 = 63,546. **The 63,546 comparisons take only a split second.** The results are in file, b1.txt, which should be opened at this time.

(5) Results

B1.txt will look like this:

.\bioapi\bioapi_a_0001.jpg
.\bioapi\bioapi_b_0001.jpg
.\bioapi\bioapi_c_0001.jpg

.\bioapi\bioapi_a_0002.jpg
.\bioapi\bioapi_a_0091.jpg
.\bioapi\bioapi_b_0002.jpg
.\bioapi\bioapi_b_0091.jpg
.\bioapi\bioapi_c_0002.jpg
.\bioapi\bioapi_c_0091.jpg…

The first block means that bioapi_a_0001.jpg is compared with all of the other images specified by the input file and 2 matches are found; this result is correct.

Because there are 357 images in the input file, you should have 357 blocks in the output file. Note that this is N*(N-1)/2 comparison, not N*N comparison; therefore, bioapi_a_0001 will match with bioapi_b_0001; but bioapi_b_0001 will not match with bioapi_a_0001, because it will never meet with bioapi_a_0001.

(6) Analysis

Scroll all the way to the end of the output file; the last line is:

Total Number of Matches = 874

i.e. the **DocumentFinder** found 874 matches. The actual matching pairs are as follows, 119 for (a, b) pairs; 119 for (a, c) pairs; 119 for (b, c) pairs; and 357 total. The results are:

Total Images N = 357
Possible Matches N*(N-1)/2 = 63,546
Attrasoft Matches = 874

Actual Duplicates = 357
Attrasoft Found Duplicates = 357

Percent Eliminated = 98.6 %
= 1 - 874/63,546
Percent of Duplicates Found = 100 %
= 357/357

4.7 Parameters

The first of the two parameters has three settings:

- 96% Setting
- 97% Setting
- 98% Setting (Default)

The 96% Setting will find 96% of the duplicates and the 98% Setting will find 98% of the duplicates. The 98% Setting will let more images pass through.

There are 3 menu items:

Templates/96% Setting
Templates/97% Setting
Templates/98% Setting

Click one of the above to make a selection.

The second of the two parameters is to choose one of the four training commands:

Query/Training
Query/Training (High Accuracy)
Query/Training (Low Accuracy)
Query/Training (Lower Accuracy)

4.8 Checking the Results

If this is a test run, i.e. you know the correct answers. You can see the matching results in seconds. You must prepare a file, which indicates the matching pairs. To test the results in b1.txt, you must prepare B1_matchlist.txt file; to test the results in b2.txt, you must prepare B2_matchlist.txt file; ….

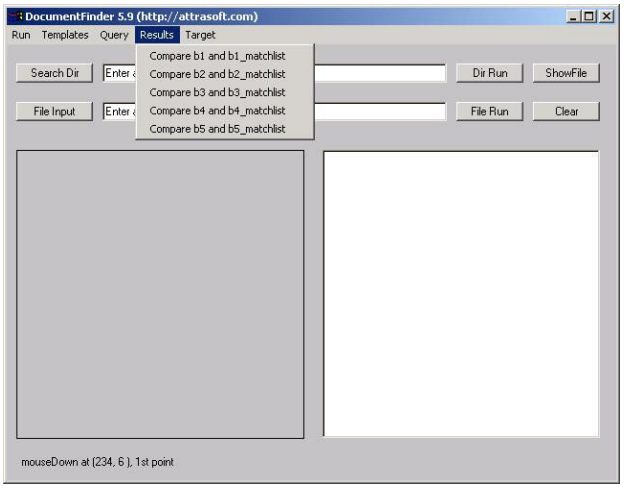


Figure 4.11 The Results Menu.

4.8.1 B1_matchlist.txt

To check the results, see b1.txt, you must have a matching file, called b1_matchlist.txt. An example of b1_matchlist.txt is:

```
5
1       IMAGE00053770    IMAGE01312024
2       IMAGE00053771    IMAGE01312025
3       IMAGE00053772    IMAGE01312026
4       IMAGE00053773    IMAGE01312027
5       IMAGE00053774    IMAGE01312028
```

Line 1 is the number of matches in this file. Each line has the following format:

Number, tab, filename, tab, filename, **tab.**

Note:

(1) You must have a tab at the end of each line;
(2) The file names do not contain ".jpg".

Once you get the two files prepared, click "Results\Compare b1 and b1_matchlist" to check format. There are four other commands for b2.txt, …, b5.txt.

There are two common errors:

(1) The last Tab is missing;
(2) The number of rows is less than the first number in the file.

4.8.2 Example "Abm54"

In this section, we will use the "Abm54" example, which has 160 triplets, or 480 images. ABM54 is the User's Guide for Attrasoft **ImageFinder** 5.4, which is the parent software of the **DocumentFinder.**

Generating a4.txt:

- Click “Templates/Select File” and select “Input_abm54.txt”;
- Click “Templates/Scan File (File Input)”;
- Click “Templates/Generate a4.txt”;
- Click “a4.txt” to see the records generated.

Template Matching:

- Click “Query/Train”;
- Click “Query/Generate b4.txt”.

Correct Match

Now open “b4_matchlist.txt”, you will see the correct matches are:

480
4108 abm54_a_0001 abm54_b_0001
4109 abm54_a_0002 abm54_b_0002
4110 abm54_a_0003 abm54_b_0003
…

Results

Click “Results/Compare b4 and b4_matchlist” and you will get:

N= 480

Check...
Clear...
Total Memory = 12,164,966
Total Matches = 478
Total Memory = 12,188,142

These results show that there are 480 matches in the b4_matchlist file and b4.txt has 478 of the 480 matches.

Analysis

Scroll all the way to the end of the b4.txt file, the last line is:

Total Number of Matches = 900

i.e. the **DocumentFinder** found 900 matches. The actual number of matching pairs are computed as follows, 160 for (a, b) pairs; 160 for (a, c) pairs; 160 for (b, c) pairs; and 480 total. The results are:

Total Images N = 480
Possible Matches N*(N-1)/2 = 114,960
Attrasoft Matches = 900

Actual Duplicates = 480
Attrasoft Found Duplicates = 478

Percent Eliminated = 99.2 %
= 1 - 900/114,960
Percent of Duplicates Found = 99.6 % = 478/480

4.8.3 Example “All”

In this section, we will use the “All” example, which combines the three previous examples and has 1,023 images.

If you have not changed the default data, the “a5.txt” file is the template file, which has 1,022 records.

Template Matching:

- Click “Query/Train”;
- Click “Query/Generate b5.txt”.

Open b5.txt, the last line is:
Total Number of Matches = 1,913.

Correct Match

There are 930 correct matches in “b5_matchlist.txt, which is the summation of all correct matches in the previous examples, 93 + 357 + 480 = 930.

Results

Click “Results/Compare b5 and b5_matchlist” and you will get:

N= 930

Check...
Clear...
Total Memory = 11,318,022
Total Matches = 928
Total Memory = 11,347,118

These results show that there are 930 matches in the b5_matchlist file and b4.txt has 928 of the 930 matches.

Analysis

Scroll all the way to the end of the b5.txt file, the last line is:

> Total Number of Matches = 1913.

i.e. the **DocumentFinder** found 1,913 matches.

Total Images N = 1,023
Possible Matches N*(N-1)/2 = 522,753
Attrasoft Matches = 1,913

Actual Duplicates = 930
Attrasoft Found Duplicates = 928

Percent Eliminated = 99.6 %
= 1 – 1,913/522,753
Percent of Duplicates Found = 99.78% = 928/930

4.9 Query Set Vs Target Set

4.9.1 Query Set against Target Set

The number of comparisons for a set with N images are N (N –1)/2. If N is large, this number is large. A natural division is, say,

N = A + B + C

And N (N-1)/2 is converted into AA, AB, AC, BB, BC, CC. Now AA, BB, and CC are self-matching, but AB, AC, and BC are not. The data set is divided into:

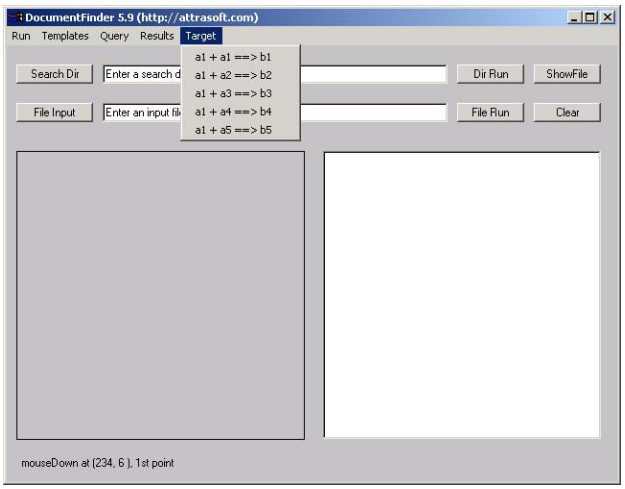


Figure 4.12 Query Set against Target Set.

Target Set

> The Target Set is a dataset being searched against in a given test or subtest: an experiment searches a Query Set against a **Target Set**. The Target Set is also known as a database.

Query Set

> The Query Set is a dataset containing the searches for a given test or subtest: an experiment searches a **Query Set** against a Target Set. The Query Set is also known as a Search set.

There are 5 commands for this purpose, see **Figure 4.12**

Note that "Target\ a1 + a1 ==> b1" is different from the earlier command:

- "Query/Generate b1.txt" is N (N-1)/2 comparison;

- "Target\ a1 + a1 ==> b1" is N*N comparison.

4.9.2 Examples

If you have not changed the default data, the "a5.txt" file is the template file, which has 1,023 records, and "b1_matchlist.txt" is the match-list file. Save a5.txt to a1.txt.

In the N*(N-1)/2 Matching, there are 930 matching pairs. In the N*N matching, there are 1,023 self matching and 930 mirror matching pairs. The total number of matches should be 930 + 1,023 + 930 = 2,883.

Click "Target\ a1 + a1 ==> b1" to make an N*N comparison and you will get 4,828. The total number of matches is 1,023 x 1,023 = 1,046,529, which will take about 5 seconds. Click "Results/Compare b1 and b1_matchlist" and the **DocumentFinder** will indicate 2,863 out of 2,883 duplicates have been identified.

Total Images N = 1,023
Possible Matches N*N = 1,046,529
Attrasoft Matches = 4,828

Actual Duplicates = 2,883
Attrasoft Found Duplicates = 2,863

Percent Eliminated = 99.5 %
= 1 – 4,828/1,046,529
Percent of Duplicates Found = 99.3 % = 2,863/2,883

5. ImageExaminer

The **ImageExaminer is** again **a customized version** of the **Attrasoft ImageFinder.**

The **Attrasoft ImageExaminer** provides the users with **a tool for Matching similar images.** The images are basically the same, but with minor changes. For example, in two satellite images, there is a car in one image, but not in the other.

To start the **ImageExaminer**, click "Example/ImageExaminer".

5.1 Introduction

The Attrasoft **ImageExaminer** compares two basically identical images and identifies the minor differences. For example,

- In two satellite images, there is a car in one image, but not in the other.
- In Product Labels, one has a different color than the other.

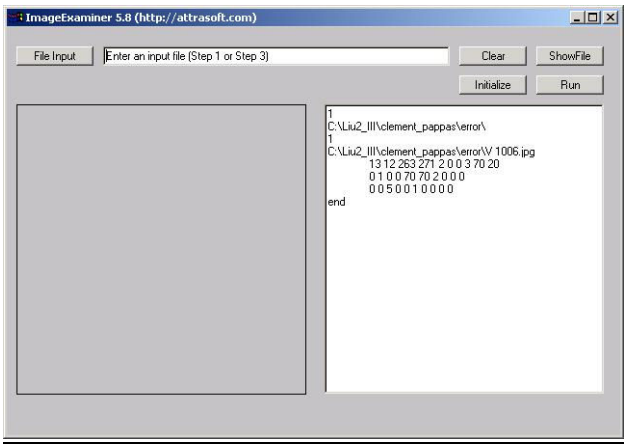


Figure 5.1 Attrasoft **ImageExaminer**.

Possible applications are:

Satellite Image Recognition
Quality Control
Product Label Recognition…

The parameters and options of the **ImageExaminer** are hidden; users have no control of the parameters. In general, each type of image has some characteristics, which require special fine-tuning. Because the options are not exposed, you might need a customized version for higher recognition rates. Customized versions of the **ImageExaminer** are available upon request.

5.2 Operations

Attrasoft ImageExaminer looks at a new image (jpg or gif) and compares it with a set of previously stored images.

5.2.1 Data

An input file enters data into the **ImageExaminer**. The input file is a text file, which looks like this:

C:\imagedata\Newlycaptured\L12063.jpg
C:\imagedata\Stored\L12063.jpg

In this file, the image path is listed one per line. Please do not a add blank space to the end of the line. The first line,

C:\imagedata\Newlycaptured\L12063.jpg,

is the newly captured image, which will be matched against the existing images, C:\imagedata\Stored\L12063.jpg.

The number of the previously stored images is arbitrary; for example, in this input file:

C:\newlycaptured\L12063.jpg
C:\masterdata1\L12063.jpg
C:\masterdata2\L12063.jpg
C:\masterdata3\L12063.jpg

The newly captured image, C:\newlycaptured\L12063.jpg, will be

matched against three previously stored images.

Multiple matches can be entered in a single input file. For example, in this input file:

C:\newlycaptured\L12063.jpg
C:\masterdata1\L12063.jpg
C:\masterdata2\L12063.jpg
C:\masterdata3\L12063.jpg

C:\newlycaptured\L12064.jpg
C:\masterdata1\L12064.jpg
C:\masterdata2\L12064.jpg
C:\masterdata3\L12064.jpg

C:\newlycaptured\L12065.jpg
C:\masterdata1\L12065.jpg
C:\masterdata2\L12065.jpg
C:\masterdata3\L12065.jpg

Three comparisons will be made in a single run.

5.2.2 Commands

After the software is started, you must click the **"Initialize" button**, which will set up the software. This will take a few seconds. This button is clicked only once.

To enter data, click the **"File Input"** button to select the input file. Click the **"Show File"** button to check the content of the input file; and click the **"Clear"** button to clear the text window.

Matching images require only a single click: click the **"Run"** button.

Initialize
: Use the "Initialize" button to initialize the software. You must click the **"Initialize"** button. After the software is started, you only need to click it once.

File Input
: Use the "File Input" button to select the input file.

ShowFile
: Use the "ShowFile" button to see the content of the input file.

Clear
: Use the "Clear" button to clear the text window.

Run
: Use the "Run" button to match images in the input file.

5.2.3 Parameters

All of the **ImageFinder** parameters and options in the **ImageExaminer** are hidden; users have no control of the parameters. The parameters must be tuned in a customized version.

5.3 Getting Started

In this chapter, we will introduce several examples.

5.3.1 The Problem

The quality control department wants to make sure the newly produced product images match the existing images.

Requirements:

- **Match or No Match?**
- **If not, where is the error?**

5.3.2 Good Match

Good Match verifies the newly captured image matches the existing images. In this section, we will match the two images in Figure 3.2.

Figure 5.2 Good Match.

Operation:

1. Click Example/ImageExaminer to start the **ImageExaminer**;

2. Click the "Initialize" button;

3. Click the "File Input" button and open "ie_good_match1.txt". Click "ShowFile" to see:

.\imgexam1\9961400239 1001.jpg
.\imgexam1\9961400239\9961400239 1002.jpg

You should see Figure 5.3.

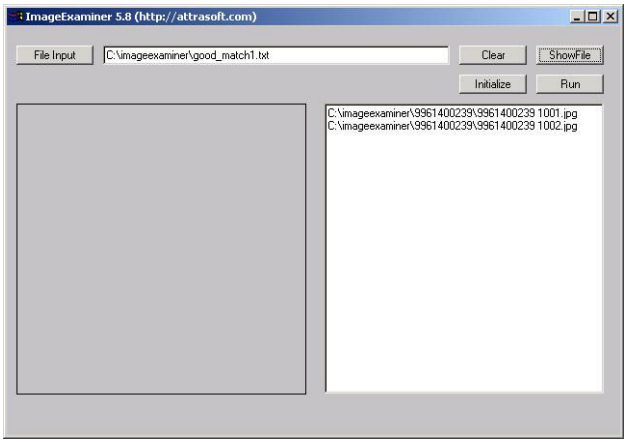


Figure 5.3 Click the "File Input" button and the "ShowFile" button.

4. Click the "Run" button and after the computation is over, you will see **Figure** 5.4.

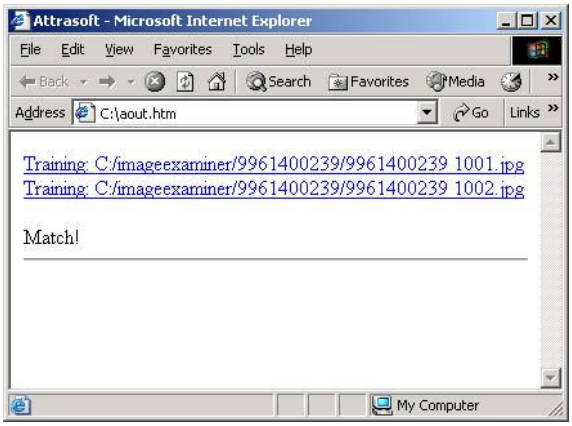


Figure 5.4 Match Results.

5.3.3 Bad Match

Bad Match rejects the newly captured image because it does not match the existing images. In this section, we will attempt to match the two images in Figure 5.5.

Figure 5.5 Bad Match.

The errors are:

Error Number	Difference	Coordinate
1	0g vs. 4%	21
2	Canned vegetable and Fruits for good nutrition Try a variety of key food's Vs. Fruits for good nutrition Try a variety of key food's Canned vegetable and 22, 32	
3	Extra word "Cranberry" 13	

The images are 200 pixels per inch. The coordinate 21 means this: starting from upper left corner, go 2 inches to the right, 1 inch down and look at the 1 inch square box.

Operation:

1. Click Example/ImageExaminer to start the **ImageExaminer**;

2. Click the “Initialize” button;

3. Click the “File Input” button and open “ie_bad_match1.txt”. Click “ShowFile” to see:

.\biofilter2_bad10\l01016key.jpg
.\biofilter2_good10\l01016key.jpg

4. Click the “Run” button and after the computation is over, you will see the following in the output web page:

Training: C:/…/l01016key.jpg
Training: C:/…/l01016key.jpg
Check: 1 3! Check: 3 2! Check: 3 3! Check!

Again, the coordinate 13 means this: starting from upper left corner, go 1 inch to the right, 3 inches down. At this point, look 1 inch in each direction (right, down, up, left).

Error: 21, 22, 32, 13.
Check: 13, 32, 33.

5.4 Multiple Matches

5.4.1 Multiple Good Matches

1. Click Example/ImageExaminer to start the **ImageExaminer**;

2. Click the “Initialize” button;

3. Click the “File Input” button and open “ie_good_match10.txt”. Click “ShowFile” to see:

.\imgexam1\9961400239 1001.jpg
.\imgexam1\9961400239 1002.jpg

.\imgexam1\9948240555 1001.jpg
.\imgexam1\9948240555 1002.jpg

.\imgexam1\9948240556 1001.jpg
.\imgexam1\9948240556 1002.jpg
…

4. Click the “Run” button; this will take a while.

5. Open c:\aout.htm to see the partial results. Keep clicking the “Refresh” button to see the updated results.

There are 10 good pairs.
10 out of 10 received “Match!”.

5.4.2 Multiple Bad Matches

1. Click Example/ImageExaminer to start the **ImageExaminer**;

2. Click the “Initialize” button;

3. Click the “File Input” button and open “ie_bad_match10.txt”. Click “ShowFile” to see:

.\biofilter2_bad10\l01016key.jpg
.\biofilter2_good10\l01016key.jpg

.\biofilter2_bad10\l12063nec-b.jpg
.\biofilter2_good10\l12063nec-b.jpg

.\biofilter2_bad10\l14002par.jpg
.\biofilter2_good10\l14002par.jpg
…

4. Click the “Run” button; this will take a while.

5. Open c:\aout.htm to see the partial results. Keep clicking the “Refresh” button to see the updated results.

There are 10 bad pairs.
10 out of 10 received “Check!” .

Figure 5.6 Another Bad Match.

The error table for the 10 bad-pairs is:

Error Number	Difference	Coordinate
1. l01016key.jpg		
1	0g vs 4%	21
2	Canned vegetable and Fruits for good nutrition Try a variety of key food's Vs. Fruits for good nutrition Try a variety of key food's Canned vegetable and	22, 32
3	Extra word "Cranberry"	13
2. L12063		
1	Extra "Juice"	12
2	Chill and Shake wee Before Using Pasteurized Vs. Chill and Shake wee Before Using	12, 22
3		

160
41g 02

4 extra "130%" 03

5 extra "OZ.", "1"
24

6 extra "0" 14

3. L14002
1 Shift “% Daily Value”
2,15 2,16

2 shift “og” 1,15
1,16

3 Extra “Contain ..” 0,15
0,16

4 2 icons
vs
“To maintain product quality..) 2,17

4. L19063
1 Extra " Calories from Fat 0" 01

2
Dietary Fiber 0g
Sugars 39g
vs
Sugars 39g
Dietary Fiber 0g 02

3 extra "a" 13

4
CA Cash Refund
vs
Distributed by Aberson's Inc
General office Boise, ID 83726
13, 23

5. l31063fol-b

1 Extra FoodLion Icon 01 11

2
contain 100% Fuit juice
vs
contain 100% Fuit juice – P
aterized - Refrigirate after openning
02 12

3
Extra "not in regular grappefruit juice"
12 22

4
Foodlion label

Distributed by foodlion inc.
Foodlion, LLC.
13 23

6. l31063kro-f.jpg
1
Shake well
vs
Paterized
Refrigrate after opeening 24, 34

7. l33063rk-f.jpg
1
100
vs
100% 01 02
11 12

2
%
U 22 32

8. l35063rk-b
1 Extra "With", "% JUICE" 12

2 Extra "C" 21

3 Extra "Flavor" and freshness" 22

4
0
vs
8 14

5

REF ME 5c
CA Cash refund
64 FL. OZ.

ME 5c
Distributed BY
CLEMENT PAPPAS CO. ,
INC
SEABROOK. NJ. 08302
23 33

9. l46045whf-b

1	Extra " Calories From Fat 0"	11
2	Extra "100%"	13
3	99482 vs. 40322	23

10. l54763pm-b

1	Juice 25% Contain vs Contain 25% JUICE	00, 10
2	Shift "% Daily Value"	01, 11
3	Extra "Water"	21
4	26 vs 100	02
5	extra “Made In USA C1999”	
6	extra “64 Oz Strawberry Kiwi Splash”	12, 22
7	100 vs 26g	02
8	Extra "Made in USA" 22	
9	Extra "Pasteurized ..."	13 23
10	Extra pasteurized	04

5.4.3 Limitation of the Software

All of the error images in the last section are identified; however, if each of the following 3 errors is a standalone error, the **ImageExaminer** might not be able to detect each one individually. The errors are:

Error Number	Difference	Coordinate
1. l01016key.jpg		
1	0g vs 4%	21
8. l35063rk-b		
2	Extra "C"	21
4	0 vs 8	14

These are single letter errors, which cannot be detected by the **ImageExaminer**. In order for the **ImageExaminer** to detect the error, the difference must be greater than a certain size.

6. Image Processing

The image processing will be applied to all images before recognition. As far as the operation is concerned, it means to set three filters:

Edge Filters;
Threshold Filters; and
Clean Up Filters.

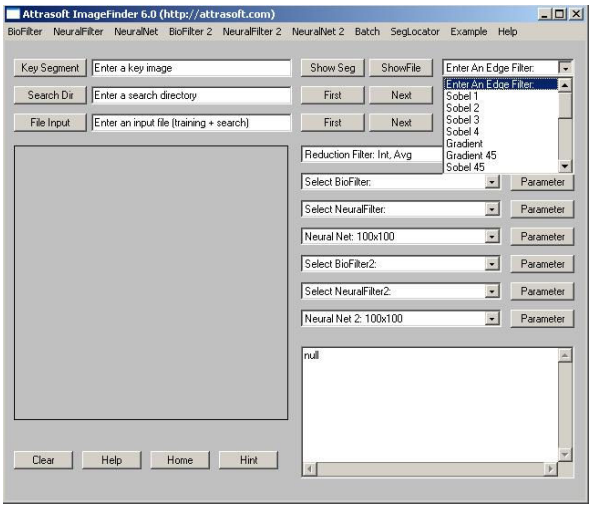


Figure 6.1 Edge Filter.

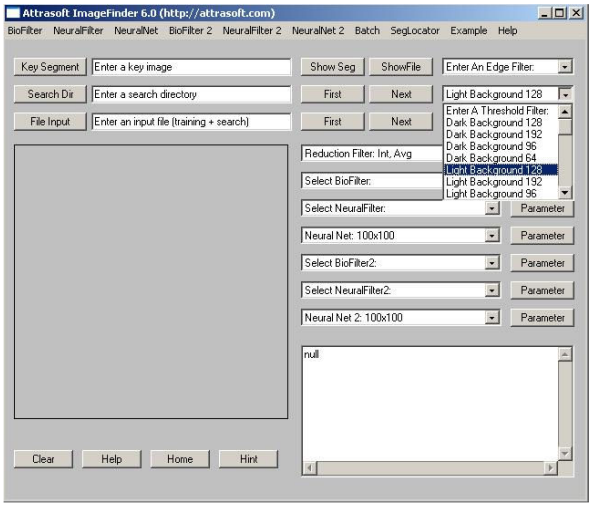


Figure 6.2 Threshold/Background Filter.

To select an Edge Filter, click the Edge Filter Drop Down List, which is the first List. To select a Threshold Filter, click the Threshold Filter Drop Down List, which is the second List. To select a Clean Up Filter, click the Clean Up Filter Drop Down List, which is the third List.

The Edge Filters attempt to exaggerate the main features a user is looking for. The Threshold Filters attempt to suppress the background. The Clean-Up Filters will smooth the resulting image to reduce recognition error.

The default setting is:

Edge Filter = None (Default)
Threshold Filter = Light Background 128 (Default)
Clean-Up Filter = None (Default)

The default setting should be your first choice. The advantage is this setting is fast. Your second choice should be:

- The first drop down list is "Edge Filter"; select "Sobel 1"(first choice).
- The second drop down list is "Threshold Filter"; select "Dark Background 128" (first choice).

Another option is:

Edge Filter = "Sobel 2"(second choice).
Threshold Filter = "Dark Background 128" (first choice).

To see how this setting works, select a training image. The **ImageFinder** uses the image display area in the following way:

(1) Right after you select a training image, the image processing filters will be applied to the training image, which will then be displayed.
(2) Right after you select a search directory or file, the first image in the search directory or file will be displayed. The image processing filters will NOT be applied to this image, so what you will see is the original image.

(3) To switch between these two images: click the "Key Segment" textbox and hit Enter on the keyboard to display the PROCESSED training image; click the "Search Dir" textbox and hit Enter on the keyboard to displayed the first image in the search directory, which is NOT PROCESSED.

Image Processing can make it or break it. For many problems like finger prints, palm prints, …, special image processing filters will be required.

Attrasoft ImageFinder learns an image in a way similar to human eyes:

- Ignore the background;
- Focus on an object in the image.

The image processing prepares the image for the **ImageFinder**. The image processing process is not unique; there are many options available. Some are better than others.

The principle of choosing the image processing filters is to make the **sample objects stand out, otherwise change the options**.

If you do not have a good image processing filter in the off-the-shelf **ImageFinder**, **a customized filter has to be built**. Do not make too many things stand out, i.e. as long as the area of interest stands out, the rest should show as little as possible.

7. BioFilters

Before the ABM engine processes an image directly (Input Space Matching), it will go through a pre-matching step, called Feature Space Matching. The purpose of Feature Recognition is to eliminate unmatched images. This Feature Recognition sub-layer has two filters:

BioFilter; and
NeuralFilter.

Additional Feature Space Filters (for example, the filter used in the software Attrasoft DecisionMaker) can be ordered in customized version.

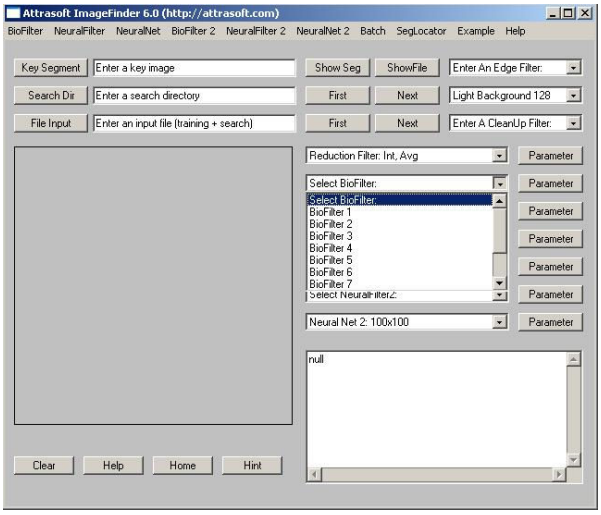


Figure 7.1 Selecting BioFilter.

This chapter will demonstrate how BioFilter works using a label-matching example. The operation of the Neural Filter is identical to the BioFilter. This chapter also introduces input mode, learning mode, and matching mode, and shows how these settings work.

Input Mode.

The images are entered into the **ImageFinder** in two ways:

- Search Directory
- Search File

Where:

- Search Directory is a folder containing the images to be searched;
- Search File is a file listing the images to be searched.

Learning Mode.

Image recognition has two learning modes:

- Supervised Learning;
- Unsupervised Learning.

Supervised Learning requires training, which teaches the **ImageFinder** what to look for. Unsupervised Learning does not require training.

Matching Mode.

The matching can be 1:N or N : N.

- 1:N compares one key image with the images in a search directory or search file; the key image is specified in the "Key Segment" textbox or selected by the "Key Segment" button.
- N: N compares each image, specified in the search directory or search file, with every image in the search directory or search file. N: N matching is further divided into N: N matching and N : (N-1)/2 matching.

Let a and b be two images; N:N matching is {aa, ab, ba, bb} and the "N : (N-1)/2" matching is {ab}. The N: N matching has N * N comparisons; and the "N : (N-1)/2" matching has N * (N-1)/2 comparisons.

Throughout this chapter, we will use a label recognition example. There are 304 images in this example, forming 152 matching pairs. They are located at the directory ".\biofilterex1", where ".\" is the **ImageFinder** software location. This example will continue throughout this chapter.

7.1 Test-Driving the Label Matching Problem, Unsupervised N:N Matching

In this section, we will help you get some experience in using the software. We will use:

- Directory Input
- Unsupervised Learning
- N : N Matching

We will go through the following steps in this section:

- Initialization
- Converting Images to Records
- Template Matching
- Results

Initialization sets the **ImageFinder** parameters. The **ImageFinder** then maps an image into a record in a Feature Space. Finally, we will do a matching.

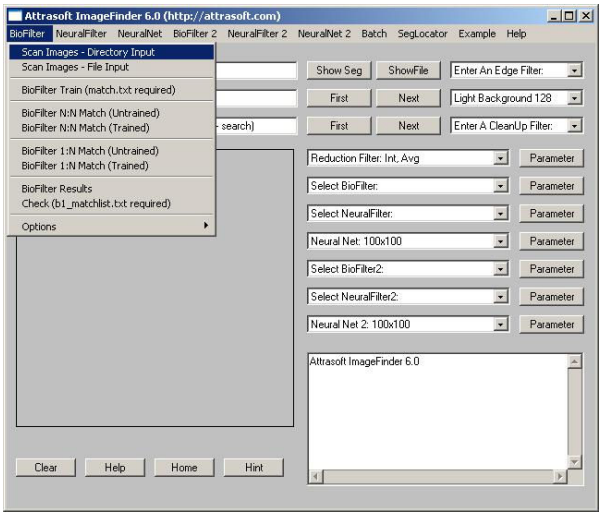


Figure 7.2 BioFilter Menu

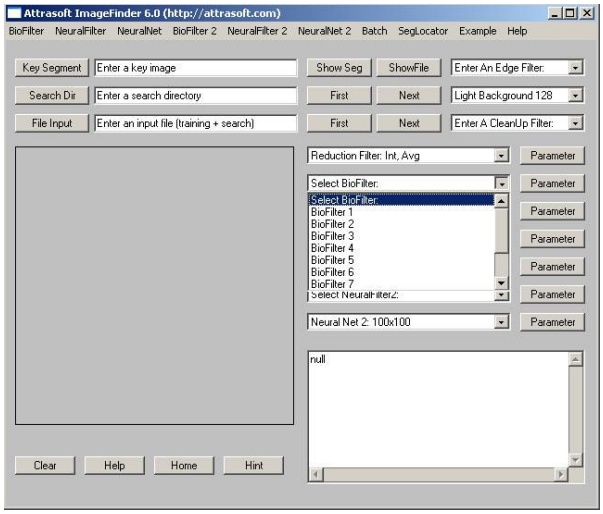


Figure 7.3 BioFilter/Drop Down List

There are 304 images in this example; they are located at the directory ".\biofilterex1", where ".\" is the **ImageFinder** software location. The N:N match will compare each of the 304 images with all 304 images.

Figure 7.4 An image in the ".\biofilterex1" folder.

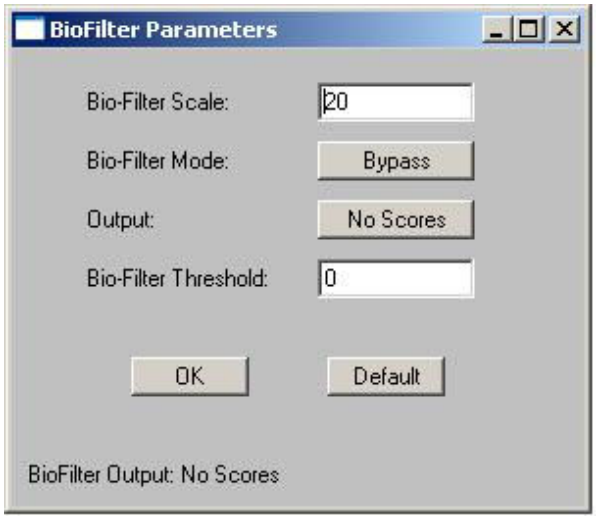


Figure 7.5 BioFilter Parameter.

Initialization

- The first drop down list is "Edge Filter"; select "Sobel 1"(first choice).
- The second drop down list is "Threshold Filter"; select "Dark Background 128" (first choice).
- The third drop is "CleanUp Filter; select "Small" (first choice).
- The fourth drop down list is "Reduction Filter"; use the default setting.
- The fifth drop down list is "BioFilter"; use the "CL" setting (tenth choice). (Figure 7.3)
- To set the BioFilter parameter, click the "Parameter" button next to the BioFilter and set the "BioFilter Scale" to 50. (Figure 7.5)

Converting Images to Records

- Click "Search Dir" button to specify the search directory, which contains the search images. After the "Open Dialog" dialog box comes up, go to "biofilterex1" and select any file in the folder. This will specify the input directory.
- Click menu item "BioFilter/Scan Images - Directory Input" to convert images to records (Figure 7.2). You should see the **ImageFinder** scan through the images at this point (Figure 7.2). The result will be stored in ".\a1.txt", where ".\" is the **ImageFinder** directory.

Template Matching

- Click menu item "BioFilter/BioFilter N:N Matching (Untrained)" button to match.

Results

- The result is in a file b1.txt, which will be opened at this time.

C:\…\L01008gi_r90.jpg
C:\…\L01008gi_r90.jpg
C:\…\L01008gi-020501.jpg

C:\…\L01008KEY_m.jpg
C:\…\L01008KEY_m.jpg

…

The result file contains many blocks. The number of blocks is the same as the number of images in the search directory, i.e. each image has a block. Line 1 in each block is input and the rest of the lines are output; i.e. the first line is the image matched against all images in the search directory; the rest of the lines represent the matched images. For example, "C:\…\L01008gi_r90.jpg" is matched against all 304 images in the search directory; there are two matches, listed in the next two lines. We will continue this example in section 3, Unsupervised 1:N Matching.

Unsupervised Learning is not as accurate the Supervised Learning, which will be introduced later.

7.2 BioFilter Overview

The image matching will be done in several steps:

- Initialization
- Converting Images to Records
- Training
- Template Matching

Let us look at each phase.

Initialization

Initialization sets the **ImageFinder** parameters.

Converting Images to Records:

- An image is mapped into a record in a Feature Space. This step is slow (several images per second); however:

- This step can be done once for all; and
- This is linear, i.e. the time is directly proportional to the number of images. Therefore, this step does not have much impact on the operating speed.
- The result will be stored in ".\a1.txt", where ".\" is the **ImageFinder** directory.

Training

Training uses the data collected in advance to teach the BioFilter how to match. Training requires two files, a1.txt and match.txt:

y records. Each image is converted into a record. A1.txt is
l represents features of an image in a feature space.
will teach the **ImageFinder** who will match with whom. You
t of match.txt later.

Template Matching

The **matching speed** will be between **100,000 – 1,000,000 comparisons per second.** Both BioFilter and Neural Filter will do the template matching. There are several commands for the matching and all of the commands will be used in this chapter.

7.3 Unsupervised 1:N Matching

1:N Matching compares one key image with the images in a search directory or search file; the key image is specified in the "Key Segment" textbox or selected by the "Key Segment" button.

To continue the label recognition problem for Unsupervised 1:N Matching:

- Click "Key Segment" button, go to "biofilterex1" folder, and select the first image "L01008gi-020501.jpg";
- Click "BioFilter/BioFilter 1:N Match (Untrained)".

The result is in file, b1.txt, which will be opened at this point:

C:\…\L01008gi-020501.jpg
C:\…\L01008gi_r90.jpg
C:\…\L01008gi-020501.jpg

Total Number of Matches = 2

This result is correct. We will continue this example in the next section, Training.

7.4 Training

Training teaches the **ImageFinder** what to look for. Unsupervised Learning does not require training. Supervised Learning requires training, which teaches the **ImageFinder** what to look for.

Each filter is trained differently. **For the BioFilter, training requires two files, a1.txt and match.txt:**

- A1.txt is the record file, which contains many records. Each image is converted into a record. A record represents features of an image in a feature space.
- Match.txt is a list of matching pairs. This file will teach the **ImageFinder** who will match with whom.

These two files for training are fixed; you cannot change the names of these two files. You obtain a1.txt automatically, as you convert images into records. You have to prepare match.txt for each problem.

The match.txt looks like this:

152
1 L01008gi_r90 L01008gi-020501
2 L01008KEY_m L01008key-082301_m
3 L010103C L010103C-081502_m
4 L01010co_m L01010CODE_m
5 L010163C_m L010163C-083100_m
…

Line 1 is the number of matches in this file. This match file indicates images, L01008gi_r90, will match with image, L01008gi-020501. Each line has the following format:

Number, tab, filename1, tab, filename1, **tab.**

Note:

(1) You must have a tab at the end of each line;

(2) The file names do not contain ".jpg".

There are two common errors:

(1) The last Tab is missing;
(2) The number of rows is less than the first number in the file.

Once you get the two files prepared, click "BioFilter\Train (match.txt required)" to train the BioFilter. (Figure 7.2) After training, you can use these commands:

"BioFilter/BioFilter 1:N Match (Trained)"
"BioFilter/BioFilter N:N Match (Trained)"

To continue the label recognition example, we must prepare the match.txt file now. This file is already prepared for you and we will simply open it and save it to match.txt. The steps are:

- Go to the **ImageFinder** folder, (The default folder is ("C:\program files\Attrasoft\ImageFinder 6.0\".), and open the file, biofilterex1_match.txt. This file lists 152 matching pairs. Save it to match.txt (overwrite the existing file). Now the training file is prepared.
- Click "BioFilter/Train" to train the BioFilter.

Now the **ImageFinder** is trained for the label recognition problem. We will continue this example in the next section, N:N Matching.

7.5 Supervised N:N Matching

N: N compares each image, specified in the search directory or search file, with every image in the search directory or search file. N: N matching is further divided into N: N matching and N : (N-1)/2 matching.

Let a and b be two images; N:N matching is {aa, ab, ba, bb} and the "N : (N-1)/2" matching is {ab}. The N: N matching has N * N comparisons; and the "N : (N-1)/2" matching has "N * (N-1)/2" comparisons. The purpose of "N : (N-1)/2" matching is to reduce the number of comparisons.

Now go back to our example, click "BioFilter/BioFilter N:N Match (Trained)" to make an N:N match. The result will go to a file, b1.txt, which will be opened right after the click. The file will look like this:

C:\…\L01008gi_r90.jpg
C:\…\L01008gi_r90.jpg
C:\…\L01008gi-020501.jpg

C:\…\L01008KEY_m.jpg
C:\…\L01008KEY_m.jpg
C:\…\L01008key-082301_m.jpg

C:\…\L010103C.jpg
C:\…\L010103C.jpg
C:\…\L010103C-081502_m.jpg

Again, line 1 in each block is the input and the rest of the lines are output. Go all the way to the end of the file; the last line is:

Total Number of Matches = 850

There are 152 pairs or 304 images. Each image will match itself and its partner in the pair, giving a total of 608 matches. As we will see next, all of the 608 matches are identified. There is a small amount of false acceptance, 850 – 608. We will further analyze the results in the analysis section.

7.6 Supervised 1:N Matching

Again, 1:N Matching compares one key image with the images in a search directory or search file; the key image is specified in the "Key Segment" textbox or selected by the "Key Segment" button.

To continue the label recognition problem for Supervised 1:N Matching:

- Click the "Key Segment" button, go to the "biofilterex1" folder, and select the first images "L01008gi-020501.jpg";
- Click "BioFilter/BioFilter 1:N Match (Trained)"

The results are in file, b1.txt, which will be opened at this point:

C:\…\L01008gi-020501.jpg
C:\…\L01008gi_r90.jpg
C:\…\L01008gi-020501.jpg

Total Number of Matches = 2

These results are correct. We will continue this example in the next section, Checking.

7.7 Checking the Results

If this is a test run (i.e., you know the correct answers), you can see the matching results in seconds. You must prepare a file, which indicates the matching pairs. To test the results in b1.txt, you must prepare B1_matchlist.txt file.

An example of b1_matchlist.txt is:

608
1 L01008gi_r90 L01008gi-020501
2 L01008KEY_m L01008key-082301_m
3 L010103C L010103C-081502_m
4 L01010co_m L01010CODE_m
5 L010163C_m L010163C-083100_m
…

Line 1 is the number of matches in this file. The format is exactly the same as match.txt.

Number, tab, filename, tab, filename, **tab.**

Note:
You must have a tab at the end of each line;
The file names do not contain ".jpg".

There are two common errors:

The last Tab is missing;
The number of rows is less than the first number in the file.

To continue the label recognition example, we must prepare the b1_matchlist.txt file now. This file is already prepared for you and we will simply open it and save it to b1_matchlist.txt (overwrite the existing file). The steps are:

- Go to the **ImageFinder** folder (The default folder is C:\program files\Attrasoft\ImageFinder 6.0\"), and open the file, biofilterex1_matchlist.txt. This file lists 608 matching pairs. Save it to b1_matchlist.txt (overwrite the

existing file). Now this file is prepared.

- Now generate the result file, b1.txt, for N:N matching by clicking the "BioFilter/BioFilter N:N Match (Trained)" menu item.
- Click "BioFilter/Check (b1_matchlist.txt required)" to check the results of the BioFilter.

You will see the something like following in the text window:

Checking Template Matching Results!

Get b1.txt...

Character = 68045

Lines = 1459

Blocks = 305

Get b1_matchlist.txt...

Check...

Total matches = 608

The message indicates b1.txt has 305 blocks: the 304 image blocks plus the last line indicating the total number of matches retrieved. The message "Total Matches = 608" indicates that 608 matches in b1.txt agrees with those in b1_matchlist.txt.

7.8 File Input

The Directory Input has two limits:

(1) The images are limited to the Input Directory only;
(2) The number of images is limited to 1,000.

The Input File has no limits; however, you must prepare the Input File. The Input Files list one image per line. Each line specifies an absolute path. For example,

C:\xyz1\0001.jpg
C:\xyz1\0002.jpg
C:\xyz2\0003.jpg
C:\xyz2\0004.jpg
…

The only difference between the Directory Input and the File Input is how images are entered into the **ImageFinder**; after that, all other steps (Initialization, Training, Matching) are the same.

To continue the label recognition example, we must prepare the input file now. There is one file for the label recognition problem, biofilterex1_input.txt. Select input file by clicking "File Input" button. Select the file, biofilterex1_input.txt. This file has 304 images and the software will go through each one of them to make sure each image in the file exists. This will take a few seconds and at the end of this process, you should see:

Number of blocks = 1
Block 0. Length = 304
C:\…\L01008gi_r90.jpg
C:\…\L01008KEY_m.jpg
C:\…\L010103C.jpg

To see the original content of this file, click "ShowFile" button. To clear the text window, click "Clear" button.

To convert the images into records, click menu item "BioFilter/Scan Images - File Input" to convert images to records. You should see the **ImageFinder** scan through the images at this point.

7.9 Analysis

Possible Matches

Let the Total Images in the input file be N, the Possible Matches will be N*N. In our example, N * N = 304 * 304 = 92,416.

Attrasoft Matches

The number of retrieved matches is listed in the last line of b1.txt. Go to the end of b1.txt, you will see something like this:

Total Number of Matches = 850

Actual Match

This number depends on your problem. It should be the first number in b1_matchlist.txt. In our example, it is 608.

Attrasoft Found Duplicates

Click "BioFilter/Check(b1_matchlist.txt required)" menu item to get this number, as discussed in the last section.

Now that you have all of the numbers, you can make an analysis.

Positive Identification Rate

Positive Identification Rate = the results of clicking "BioFilter/Check (b1_matchlist.txt required)" menu item divided by the first number in file, b1_matchlist.txt.

In our example, the
Positive Identification Rate is 100%,
i.e., 608/608 = 100%.

Elimination Rate

The Elimination Rate is 1 minus the number at the end of b1.txt divided by the number of possible matches. This number should be normalized so that if all mismatches are eliminated, it should be 100%. In our example, this number is:

Absolute Elimination Rate = 99.08%
= 1 – 850/92,416

Maximum Absolute Elimination Rate
= 1 – 608/92,416
Elimination Rate = 99.74 %
= (1 – 850/92,416) / (1 – 608/92,416)

Hit Ratio

The Hit Ratio is the number indicated by "BioFilter/Check (b1_matchlist.txt required)" menu item divided by the number at the end of b1.txt. In our example, this number is: 608/850 = 71.53%.

Composite Index

Finally, an identification is measured by multiplication of **Positive Identification Rate * Elimination Rate * Hit Ratio.** In our example, this number is 100% * 99.74% * 71.53% = 71.34 %.

7.10 Summary

Summary of the steps for N:N matching:

I. Preparations

Data

Data is stored at ".\biofilterex1". There are 304 images.

Training File

1. Training file must have name "match.txt";
2. Find the file, ".\biofilterex1_match.txt", and save it to match.txt.

Checking File

1. Checking file must have the name "b1_matchlist.txt";
2. Find the file, ".\biofilterex1_matchlist.txt", and save it to b1_matchlist.txt.

II. Operation

- Start the **ImageFinder**;
- Edge Filter: select Sobel 1;
- Threshold Filter: select Dark Background 128;
- CleanUp Filter: select Small;
- Reduction Filter: do nothing;
- BioFilter: select CL;
- BioFilter Parameter: set "BioFilter Scales:" to 50;
- Entering Data: click "Search Dir" and select any file in "biofilterex1" directory;
- Records: click menu item "BioFilter/Scan Images - Directory Input";
- Training: click "BioFilter\Train (match.txt required)";
- N:N Matching: click "BioFilter/BioFilter N:N Match (Trained)";
- Attrasoft Matches: go to the last line of b1.txt and see: "Total Number of Matches = 850";
- Checking: click "BioFilter/Check (b1_matchlist.txt required)" and see "Total Matches = 608";
- Results, out of 304 * 304 = 92,416 possible comparisons, there are 608 matches. You have found 850, including all 608 matches.

Congratulations on your very first image recognition example with the **ImageFinder**.

8. Neural Filters

Before the ABM engine processes an image directly (Input Space Matching), it will go through a pre-matching step, called Feature Space Matching. The purpose of Feature Recognition is to eliminate unmatched images. This Feature Recognition sub-layer has two filters:

BioFilter; and
NeuralFilter.

Additional Feature Space Filters (for example, the filter used in the software Attrasoft **DecisionMaker**) can be ordered in a customized version.

The BioFilter will eliminate about 80% of the unmatched images and the Neural Filter will eliminate an additional 19%, leaving only about 1% unmatched images.

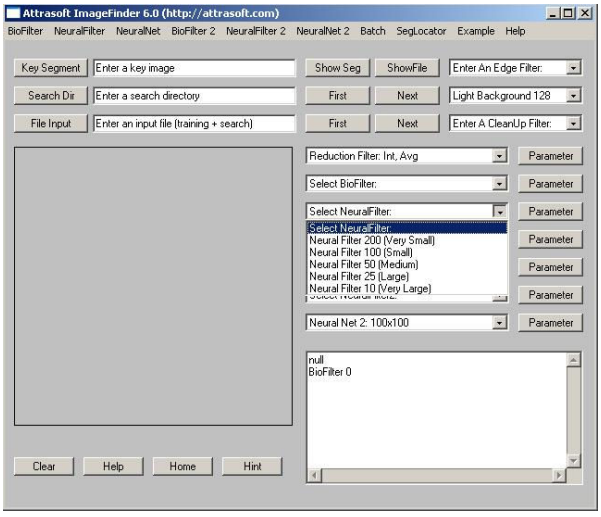


Figure 8.1 Selecting NeuralFilter.

Operating the Neural Filter is exactly the same as the BioFilter. This chapter will demonstrate how the Neural Filter works using the label-matching example of the last chapter. The Neural Filter is more accurate. The Neural Filter will automatically include the BioFilter. Also, the Neural Filter only has supervised Learning; therefore, the Neural Filter must be trained before it can be used.

8.1 Test-Driving the Label Matching Problem

We will go through the following steps in this section:

- Initialization
- Converting Images to Records
- Template Matching
- Results

Initialization sets the **ImageFinder** parameters. The **ImageFinder** then maps an image into a record in a feature space. Finally, we will do a matching.

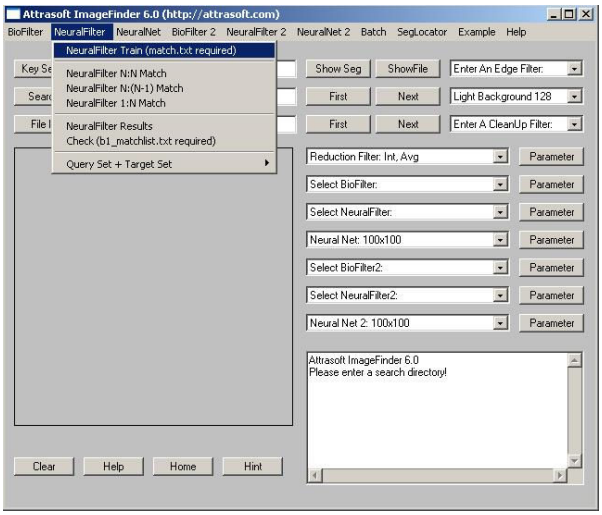


Figure 8.2 NeuralFilter Menu.

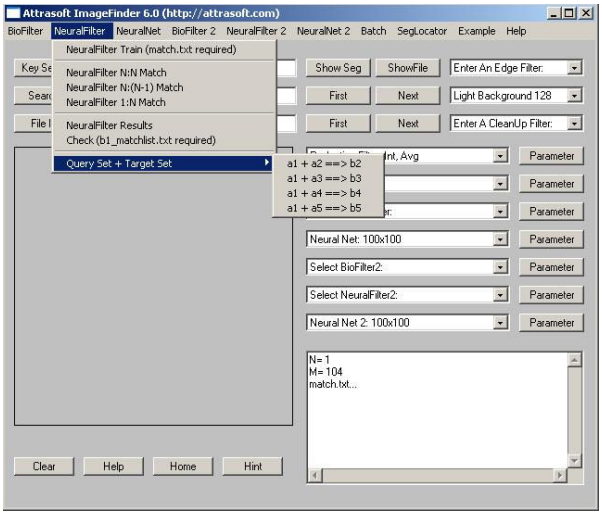


Figure 8.3 "NeuralFilter\Query Set + Target Set" Menu; please see the Reference menu chapter.

There are 304 images in this example; they are located at the directory ".\biofilterex1", where ".\" in the **ImageFinder** software location. The N:N match will compare each of the 304 images with all 304 images.

Figure 8.4 An image in the ".\biofilterex1" folder.

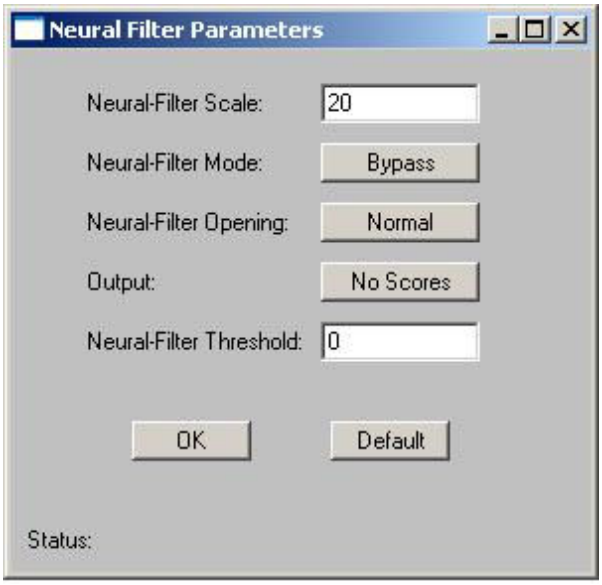


Figure 8.5 NeuralFilter Parameter.

Initialization of Neural Filter

In the last chapter, we have done the following for the BioFilter:

- The first drop down list is "Edge Filter"; select "Sobel 1"(first choice).
- The second drop down list is "Threshold Filter"; select "Dark Background 128" (first choice).
- The third drop is "CleanUp Filter; select "Small" (first choice).
- The fourth drop down list is "Reduction Filter"; use the default setting.
- The fifth drop down list is "BioFilter"; use the "CL" setting (tenth choice).

Please add two more steps:

- To set the BioFilter parameter, click the "Parameter" button next to the BioFilter and set the "BioFilter Scale" to 20.
- To set the Neural Filter parameter, click the "Parameter" button next to the NeuralFilter and set the "Neural Filter Scale" to 20.

Converting Images to Records

We have done this step in the last chapter. If you have not done so, follow these steps:

- Click "Search Dir" button to specify the search directory, which contains the search images. After the "Open Dialog" dialog box comes up, go to ".\biofilterex1" and select any file in the folder. This will specify the input directory.
- Click menu item "BioFilter/Scan Images - Directory Input" to convert images to records. You should see the **ImageFinder** scan through the images at this point.

Training

- Click "NeuralFilter\Train (match.txt required)".

Template Matching

- Click menu item "NeuralFilter/NeuralFilter N:N Matching" to initiate matching.

Results

- The results are in a file b1.txt, which will be opened at this time.

C:\…\L01008gi_r90.jpg
C:\…\L01008gi_r90.jpg

C:\…\L01008gi-020501.jpg

…

The result file contains many blocks. The number of blocks is the same as the number of images in the search directory, i.e. each image has a block. Line 1 in each block is input and the rest of the lines are output; i.e. the first line is the image matched against all images in the search directory, the rest of the lines represent the matched images. For example, "C:\…\L01008gi_r90.jpg" is matched against all 304 images in the search directory, and there are two matches, listed in the next two lines.

8.2 Neural Filter Overview

The image matching will be done in several steps:

- Initialization
- Converting Images to Records if you have not done so in the BioFilter
- Training
- Template Matching

Let us look at each phase.

Initialization

Initialization sets the **ImageFinder** parameters.

Converting Images to Records if you have not done so in the BioFilter**:**

An image is mapped into a record in a feature space. This step is slow (several images per second); however:

- This step can be done once for all; and
- This is linear, i.e. the time is directly proportional to the number of images.

Therefore, this step does not have much impact on the operating speed.

Training

Neural Filter training requires the same input files as the BioFilter. Training uses the data collected in advance to teach the BioFilter how to match. Training requires two files, a1.txt and match.txt:

- A1.txt is the record file, which contains many records. Each image is converted into a record. A record represents features of an image in a feature space.
- Match.txt is a list of matching pairs. This file will teach the **ImageFinder** who will match with whom. We will discuss the format of match.txt later.

Template Matching

The matching speed will be between 100,000 – 1,000,000 comparisons per second. Both BioFilter and Neural Filter will do the template matching.

8.3 Training

Training teaches the **ImageFinder** what to look for. Each filter is trained differently.

If you have done the BioFilter training in the last chapter, all you have to do is to click "NeuralFilter\Train (match.txt required)" to train the Neural Filter.

For both the BioFilter and the Neural Filter, training requires two files, a1.txt and match.txt:

- A1.txt is the record file, which contains many records. Each image is converted into a record. A record represents features of an image in a feature space.
- Match.txt is a list of matching pairs. This file will teach the ImageFinder who will match with whom.

These two files for training are fixed; you cannot change the names of these two files. You obtain a1.txt automatically, as you convert images to records. You have to prepare match.txt for each problem.

The match.txt looks like this:

152
1 L01008gi_r90 L01008gi-020501
2 L01008KEY_m L01008key-082301_m
3 L010103C L010103C-081502_m
4 L01010co_m L01010CODE_m
5 L010163C_m L010163C-083100_m
…

Line 1 is the number of matches in this file. This match file indicates image, L01008gi_r90, will match with image, L01008gi-020501. Each line has the following format:

Number, tab, filename1, tab, filename1, **tab.**

Note:

- **You must have a tab at the end of each line;**
- **The file names do not contain ".jpg".**

There are two common errors:

- The last Tab is missing;
- The number of rows is less than the first number in the file.

Once you get the two files prepared, click "NeuralFilter\Train (match.txt required)" to train the Neural Filter.

To continue the label recognition example, we must prepare the match.txt file now. This file is already prepared for you and we will simply open it and save it to match.txt. The steps are:

- Go to the **ImageFinder** folder, (the default folder is C:\program files\Attrasoft\ImageFinder 6.0\"), and open the file, biofilterex1_match.txt. This file lists 152 matching pairs. Save it to match.txt (overwrite the existing file). Now the training file is prepared.
- Click "NeuralFilter\Train (match.txt required)" to train the Neural Filter.

Now the **ImageFinder** is trained for the label recognition problem. We will continue this example in the next section, N:N Matching.

8.4 N:N Matching

N: N compares each image specified in the search directory or search file, with every image in the search directory or search file. N: N matching is further divided into N: N matching and N : (N-1)/2 matching.

Let a and b be two images; N:N matching is {aa, ab, ba, bb} and the N : (N-1) matching is {ab}. The N: N matching has N * N comparisons; and the N : (N-1)/2 matching has N * (N-1)/2 comparisons. The purpose of N : (N-1)/2 matching is to reduce the number of comparisons.

Now, go back to our example. Click "NeuralFilter/NeuralFilter N:N Match"; we get b1.txt, which will be opened right after the click:

C:\…\L01008gi_r90.jpg
C:\…\L01008gi_r90.jpg
C:\…\L01008gi-020501.jpg

C:\…\L01008KEY_m.jpg
C:\…\L01008KEY_m.jpg
C:\…\L01008key-082301_m.jpg

Again, line 1 in each block is input and the rest of the lines are output. Go all the way to the end of the file; the last line is:

Total Number of Matches = 608

There are 152 pairs or 304 images. Each image will match itself and its partner in the pair, giving a total of 608 matches. As we will see next, all of the 608 matches and only the 608 matches are identified.

Click "NeuralFilter/NeuralFilter N:(N-1) Match"; we get b1.txt, which will be opened right after the click; the last line is:

Total Number of Matches = 152

There are 152 pairs or 304 images. The first image in a pair will match its partner in the pair, but the second image in a pair will not match the first one, giving a total of 152 matches. As we will see next, all of the 152 matches and only the 152 matches are identified.

8.5 1:N Matching

Again, 1:N Matching compares one key image with the images in a search directory or search file; the key image is specified in the "Key Segment" textbox or selected by the "Key Segment" button.

To continue the label recognition problem for Supervised 1:N Matching:

- Click the "Key Segment" button; go to ".\biofilterex1" folder, and select the first image "L01008gi-020501.jpg";
- Click "NeuralFilter/BioFilter 1:N Match (Trained)"

The result is in file, b1.txt, which will be opened at this point:

C:\…\biofilterex1\L01008gi-020501.jpg
C:\…\L01008gi_r90.jpg
C:\…\L01008gi-020501.jpg

Total Number of Matches = 2

This result is correct. We will continue this example in the next section, Checking.

8.6 Checking the Results

If this is a test run (i.e. you know the correct answers) you can see the matching results in seconds. You must prepare a file, which indicates the matching pairs. To test the results in b1.txt, you must prepare B1_matchlist.txt file.

If you have done the BioFilter checking in the last chapter, all you have to do is to click NeuralFilter/Check (b1_matchlist.txt required)" to check the result of the NeuralFilter.

An example of b1_matchlist.txt is:

608
1 L01008gi_r90 L01008gi-020501
2 L01008KEY_m L01008key-082301_m
3 L010103C L010103C-081502_m
4 L01010co_m L01010CODE_m
5 L010163C_m L010163C-083100_m
…

Line 1 is the number of matches in this file. The format is exactly the same as match.txt.

Number, tab, filename, tab, filename, **tab.**

Note:
You must have a tab at the end of each line;
The file names do not contain ".jpg".

There are two common errors:

The last Tab is missing;
The number of rows is less than the first number in the file.

To continue the label recognition example, we must prepare the b1_matchlist.txt file now. This file is already prepared for you and we will simply

open it and save it to b1_matchlist.txt (overwrite the existing file). The steps are:

- Go to the **ImageFinder** folder, (the default folder is C:\program files\Attrasoft\ImageFinder 6.0\".), and open the file, biofilterex1_matchlist.txt. This file lists 608 matching pairs. Save it to b1_matchlist.txt (overwrite the existing file). Now this file is prepared.
- Now generate the result file, b1.txt, for the N:N matching by clicking the N:N Matching button.
- Click the "NeuralFilter/Check (b1_matchlist.txt required)" to check the results of the Neural Filter.

You will see the something like following in the text window:

Checking Template Matching Results!

Get b1.txt...

Character = 68045

Lines = 1459

Blocks = 305

Get b1_matchlist.txt...

Check...

Total matches = 608

The message indicates b1.txt has 305 blocks: the 304 image blocks plus the last line indicating the total number of matches retrieved. The message "Total Matches = 608" indicates that 608 matches in b1.txt agrees with those in b1_matchlist.txt.

8.7 Analysis

Possible Matches

Let the Total Images in the input file be N; the Possible Matches will be N*N. In our example, N * N = 304 * 304 = 92,416.

Attrasoft Matches

The number of retrieved matches is listed in the last line of b1.txt. Go to the end of b1.txt, you will see something like this:

Total Number of Matches = 608

Actual Matches

This number depends on your problem. It should be the first number in b1_matchlist.txt. In our example, it is 608.

Attrasoft Found Duplicates

Click the "Results" button to get this number, as discussed in the last section.

Now that you have all of the numbers, you can make an analysis.

Positive Identification Rate

Positive Identification Rate = the result of clicking "Neural Filter/Check (b1_matchlist.txt required)" menu item divided by the first number in file, b1_matchlist.txt. In our example, 608/608 = 100%, i.e. the **Positive Identification Rate is 100%.**

Elimination Rate

The Elimination Rate is 1 minus the number at the end of b1.txt divided by the number of possible matches. In our example, this number is:

(1 – 608/92,416) / (1 – 608/92,416) = **Elimination Rate = 100%.**

Hit Ratio

The Hit Ratio is the number indicated by "Neural Filter/Check (b1_matchlist.txt required)" menu item divided by the number at the end of b1.txt. In our example, this number is: 608/608 = 100%. **Hit Ratio = 100%.**

Composite Index

Finally, an identification is measured by multiplication of **Positive Identification Rate * Elimination Rate * Hit Ratio.** In our example, this number is 100% * 100% * 100% = 100 %, i.e.
Composite Index = 100%.

8.8 Summary

Summary of the steps for N:N matching:

I. Preparations

Data

Data is stored at ".\biofilterex1". There are 304 images.

Training File

1. Training file must have the name "match.txt";
2. Find the file, ".\biofilterex1_match.txt", and save it to match.txt.

Checking File

1. Checking file must have the name "b1_matchlist.txt";
2. Find the file, ".\biofilterex1_matchlist.txt", and save it to b1_matchlist.txt.

II. Operation

- Start the **ImageFinder;**
- Edge Filter: select Sobel 1;
- Threshold Filter: select Dark Background 128;
- CleanUp Filter: select Small;
- Reduction Filter: do nothing;
- BioFilter: select CL;
- BioFilter Parameter: set "BioFilter scales:" to 20;
- Entering Data: click "Search Dir" and select any file in "biofilterex1" directory;
- Converting to Records (if you have not done so): click menu item "BioFilter/Scan Images - Directory Input";
- Training: click "NeuralFilter\Train (match.txt required)";
- N:N Matching: click "NeuralFilter/Query Set + Target Set/a1 + a1 ==> b1";
- Attrasoft Matches: go to the last line of b1.txt and see: "Total Number of Matches = 608";
- Checking: click "NeuralFilter/Check (b1_matchlist.txt required)" and see "Total matches = 608";
- Results: out of 304 * 304 = 92,416 possible comparisons, there are 608 matches. **You have found 608, including all 608 matches.**

9. NeuralNet Filters

The **ImageFinder** recognizes images in two phases:

- Feature Space Matching
- Input Space Matching

The purpose of Feature Recognition is to eliminate unmatched images. This Feature Recognition sub-layer has two filters:

BioFilter; and
NeuralFilter.

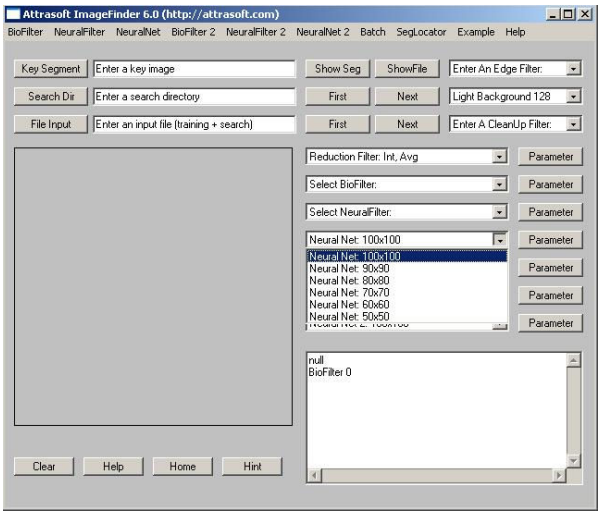


Figure 9.1 Selecting NeuralNet Filter.

The main image recognition uses the NeuralNet or ABM Filter. This chapter will demonstrate how NeuralNet Filter works using a logo-matching example. The output file of the Neural Filter will be the input file of the NeuralNet Filter.

Throughout this chapter, two filters will be used: the Neural Filter works in the Feature Space Matching and the NeuralNet filter works in the Input Space Matching (Pixel Space).

Input Mode

The images are entered into the **ImageFinder** in two ways:

- Search Directory
- Search File

Where:

- Search Directory is a folder containing the images to be searched;
- Search File is a file listing the images to be searched.

Matching Mode

The matching can be 1:N or N : N.

- 1:N compares one key image with the images in a search directory or search file; the key image is specified in the "Key Segment" textbox or selected by the "Key Segment" button.
- N: N compares each image, specified in the search directory or search file, with every image in the search directory or search file. N: N matching is further divided into N: N matching and N : (N-1)/2 matching.

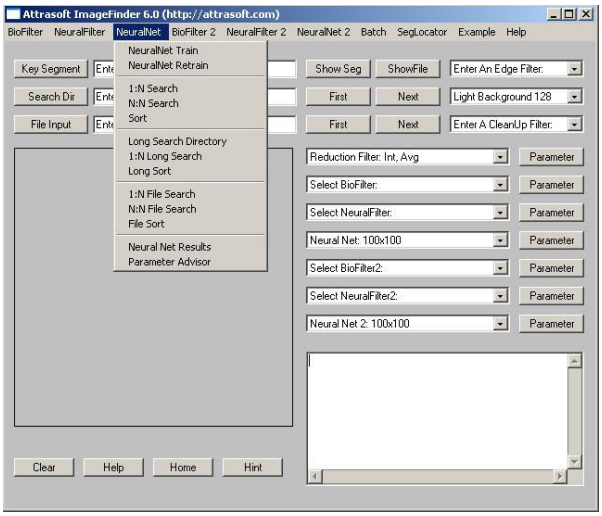


Figure 9.2 NeuralNet Menu.

In the last chapter, we have seen that for scanner images, the BioFilter will eliminate about 80% of the unmatched images and the Neural Filter will eliminate an additional 19%, leaving only about 1% of the unmatched images. In this chapter, we

will use camera images. Camera images have more degrees of freedom with distances and directions, therefore, they are hard to recognize. The Feature Space Recognition alone will not be enough. The NeuralFilter will eliminate 95%, leaving only about 5% of the unmatched images for the NeuralNet Filter.

Throughout this chapter, we will use a logo recognition example. There are 489 images in this example, representing about 30 different logos. They are located in the directory ".\neuralnetex1", where ".\" is the **ImageFinder** software location.

9.1 Test-Driving the Logo Matching Problem

We will go through the following steps in this section:

- Feature Space Recognition
- Input Space Recognition

For Feature Space Recognition, the steps are the same as the last chapter:

- Initialization
- Converting Images to Records
- Training
- Template Matching
- Results

For the Input Space Recognition, the steps are:

- Initialization
- Pixel Matching
- Results

Initialization sets the **ImageFinder** parameters. In the Feature Space Recognition, the **ImageFinder** then maps an image into a record in a feature space and does a matching with the records. In the Input Space Matching, the **ImageFinder** works with the pixels directly.

There are 489 images in this example; they are located at the directory ".\neuralnetex1", where ".\" is the **ImageFinder** software location. In the following, we will combine the initialization of the NeuralNet Filter and Neural Filter together.

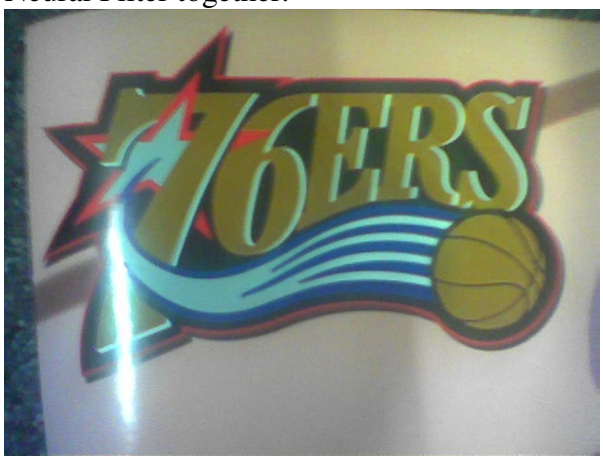


Figure 9.3 An image in ".\neuralnetex1".

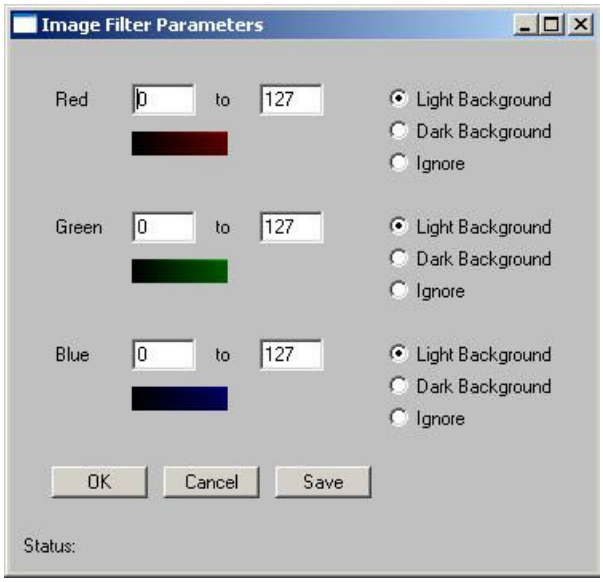


Figure 9.4 Threshold Filter Parameter.

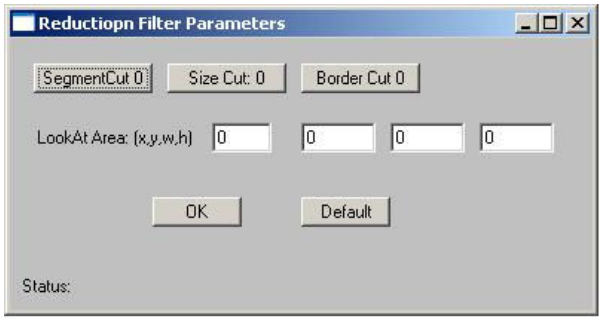


Figure 9.5 Reduction Filter Parameter.

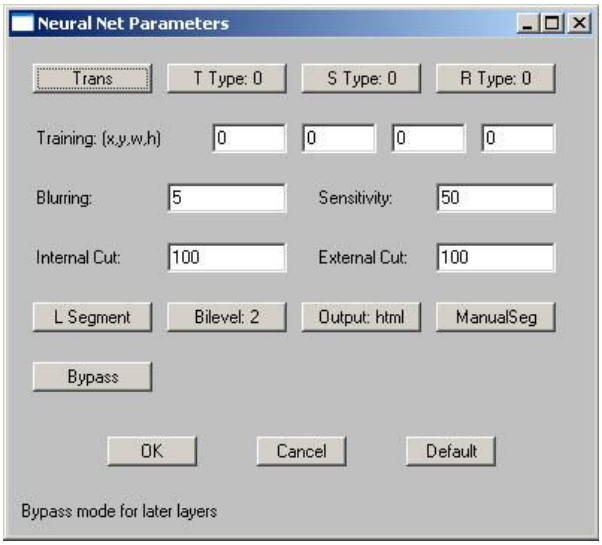


Figure 9.6 Neural Net Filter Parameter.

Summary of Parameters

1. Edge: Sobel 2

2. Threshold:
   Average Filter
   155 255 Dark Background
   155 255 Dark Background
   155 255 Dark Background

3. Clean Up: Small

4. Reduction: Default
   Parameters:
   Border Cut = 9

5. BioFilter: 6

6. NeuralNet Filter: 200 (Very Small)
   Parameters:
   Opening = very small
   Threshold = 1200

7. Neural Net: 100x100
   Parameters:
   Blurring = 32
   Sensitivity = 80
   Internal Cut = 50
   External Cut = 82000
   Output = text
   Segment type = "AutoSeg 10"

Initialization of Neural Filter

- The first drop down list is "Edge Filter"; select "Sobel 2"(Second choice).
- The second drop down list is "Threshold Filter"; select "Average Filter" and set it to (See Figure 9.4):

  155 255 Dark Background
  155 255 Dark Background
  155 255 Dark Background

- The third drop is "CleanUp Filter; select "Small" (first choice).
- The fourth drop down list is "Reduction Filter"; use the default setting. Click the Parameter button next to it and set Border Cut to 9 (See Figure 9.5). We will explain the Border Cut parameter later. To set the parameter, keep clicking the button.
- The fifth drop down list is "BioFilter"; use the "BioFilter 6" setting.
- The sixth drop down list is "NeuralFilter"; select the "NeuralFilter 200 (Very Small)" setting.
- To set the Neural Filter parameter, click the "Parameter" button next to the NeuralFilter and set:

  "Neural-Filter Scale" = 0.
  "Neural-Filter Opening" = Very Small
  "Neural-Filter Threshold" = 1200

Initialization of NeuralNet Filter

- The seventh drop down list is "NeuralNet Filter"; use the default setting.
- To set the NeuralNet Filter parameter, click the "Parameter" button next to the NeuralNet Filter and set (Figure 9.6):

  Blurring = 32
  Sensitivity = 80
  Internal Cut = 50
  External Cut = 82000

Output = text
Segment type = "AutoSeg 10"

Again, we will explain these parameters later.

Converting Images to Records

We have done similar steps in the last two chapters:

- Click "Search Dir" button to specify the search directory, which contains the search images. After the "Open Dialog" dialog box comes up, go to ".\neuralnetex1" and select any file in the folder. This will specify the input directory.
- Click menu item "BioFilter/Scan Images - Directory Input" to convert images to records. You should see the **ImageFinder** scan through the images at this point. The results will be in a1.txt.

Neural Filter Training

Neural Filter training requires two files, a1.txt and match.txt. This file has been prepared for you in advance.

- Go to the **ImageFinder** folder, (the default folder is C:\program files\Attrasoft\ImageFinder 6.0\"), and open the file, neuralnetex1_match.txt. Save it to match.txt (overwrite the existing file). Now the training file is prepared.
- Click "NeuralFilter\Train (match.txt required)".

Template Matching

- Click menu item "NeuralFilter/Query Set + Target Set/a1 + a1 ==> b1" button to do the matching.
- The results are in a file b1.txt, which will be opened at this time. Go to the last line, you will see:

Total Number of Matches = 13,575

The result file contains 489 blocks. The number of blocks is the same as the number of images in the search directory, i.e. each image has a block. Line 1 in each block is input and the rest of the lines are output.

The total number of comparisons are: 489x489 = 239,121. The number is reduced to 13,575, a 94.3% reduction. The NeuralNet Filter will process the images left.

Input Space Matching

The b1.txt can be used as input for the NeuralNet Filter, however, 13575 matchings will take a long time. Instead, we will use just the first logo, rather than all 30 logos. The Recognition Accuracy is comparable for all 30 logos.

Cut the 17 blocks from b1.txt and put it into a file, "b1_neuralnetex1_1.txt". This file is already prepared for you in advance. The file looks like this:

.\neuralnetex1\76e_01l.jpg
.\neuralnetex1\76e_01l.jpg
.\neuralnetex1\76e_02l.jpg
.\neuralnetex1\76e_05m.jpg
…

The Neural Net Matching Steps are:

- Click the "File Input" button and select "b1_neuralnetex1_1.txt". The **ImageFinder** will check each image to make sure it exists. Wait for a while until you see:

  Number of blocks = 17
  Block 0. Length = 84

- Click menu item "NeuralNet/1:N File-Search" button to do the matching.

The NeuralNet filter will process several images per second. The file "b1_neuralnetex1_1.txt" has about 550 matches so the computation will take a few minutes. At the end of the computation, a file will be opened and the output looks like:

Training: C:/.../76e_01l.jpg
C:/.../76e_01l.jpg 344000
C:/.../76e_02l.jpg 100800
C:/.../76e_06l.jpg 112640
C:/.../76e_12l.jpg 93760
C:/.../76e_18l.jpg 84032
C:/.../spu_03l.jpg 83008

Training: C:/.../76e_02l.jpg
C:/.../76e_01l.jpg 102272
C:/.../76e_02l.jpg 130560
C:/.../76e_07l.jpg 95104
C:/.../76e_08l.jpg 82432
C:/.../76e_13l.jpg 89408
C:/.../76e_15l.jpg 96448
C:/.../net_03s.jpg 84352
C:/.../net_17m.jpg 84352
C:/.../war_16l.jpg 82752

Training: C:/.../76e_03m.jpg
C:/.../76e_03m.jpg 192000
C:/.../76e_16m.jpg 95808
C:/.../76e_17m.jpg 97600

Results

In the above output, "Training: C:/.../76e_01l.jpg" is the input file. This image is compared with 489 images. The Neural Filter reduces the number of comparisons from 489 to 84. The NeuralNet filter further reduces the number of comparisons from 84 to 6:

C:/.../76e_01l.jpg 344000
C:/.../76e_02l.jpg 100800
C:/.../76e_06l.jpg 112640
C:/.../76e_12l.jpg 93760
C:/.../76e_18l.jpg 84032
C:/.../spu_03l.jpg 83008

The first one is self-matching and it does not count. Out of the rest of the 5, the highest score is 112640, which classifies the input image, 76e_01l.jpg, to category 76e_06l.jpg, which is correct. 16 out of 17 logos are classified correctly. **The Identification Rate is 94%.**

To speed up the neural computing, select a different NeuralNet filter:

Neural Net: 50x50
Blurring = 25
Sensitivity = 70
Internal Cut = 100
External Cut = 0
Output = text
Segment type = AutoSeg 10

This time, the computation is much faster. The result is:

Training: C:/.../76e_01l.jpg
C:/.../76e_01l.jpg 770000
C:/.../76e_02l.jpg 13372
C:/.../76e_12l.jpg 12634
C:/.../76e_18l.jpg 12598
C:/.../pis_10l.jpg 11194

Training: C:/.../76e_02l.jpg
C:/.../76e_02l.jpg 130700
C:/.../76e_07l.jpg 12886
C:/.../76e_08l.jpg 12274
C:/.../76e_13l.jpg 13408
C:/.../76e_15l.jpg 13192
C:/.../cli_05m.jpg 12544
C:/.../cli_10m.jpg 12526
C:/.../kin_01m.jpg 12868
C:/.../mag_01m.jpg 12634
C:/.../net_17m.jpg 13282

Training: C:/.../76e_03m.jpg
C:/.../76e_03m.jpg 136000
C:/.../76e_17m.jpg 13318

Training: C:/.../76e_05m.jpg
C:/.../76e_03m.jpg 12220
C:/.../76e_05m.jpg 700000
C:/.../76e_17m.jpg 12238
C:/.../cli_10m.jpg 12148
C:/.../net_11l.jpg 11860
C:/.../pis_10l.jpg 11230
C:/.../pis_13l.jpg 11752

Again, 16 out of 17 logos are classified correctly. **The Identification Rate is 94%.**

Now, identify the entire 489 images:

- Click the "File Input" button and select "b1.txt". The **ImageFinder** will check each image to make sure it exists. Wait for a while until you see:

  Number of blocks = 489
  Block 0. Length = 84

- Click menu item "NeuralNet/1:N File-Search" button to do the matching.

9.2 NeuralNet Filter Overview

The **ImageFinder** recognizes images in two phases:

- Feature Space Matching
- Input Space Matching

The Feature Space Matching will be done in several steps:

- Initialization
- Converting Images to Records
- Training
- Template Matching

Input Space Recognition Steps are:

- Initialization
- Pixel Matching
- Results

We have discussed Feature Space Matching in the last two chapters.

Let us look at Input Space Matching:

Initialization

Initialization sets the **ImageFinder** parameters.

Entering Data:

The output of the Neural Filter, b1.txt, can be used as the input for the neural net matching. If this file is too large, you can split it into several pieces and match one piece at a time.

Training and Matching

NeuralNet Filter learns the first image in each block and matches it against the rest of the images in the block.

9.3 Training

Training teaches the **ImageFinder** what to look for. Each filter is trained differently. NeuralNet filter learns directly from the training image, therefore, a match file will not be required.

9.4 N:N Matching

When using file input, N: N Matching compares each image in the block against every other image in the block. N: N Matching will be done only to the very first image block in the input file.

9.5 1:N Matching

Again, 1:N Matching compares the first image in a block against the rest of the images in the block.

1:N Matching can handle many 1:N Matchings at the same time. For example, in early this chapter, a single click will handle 17 comparisons.

9.6 Summary

I. Preparations

Data

Data is stored at ".\neuralnetex1". There are 489 images.

Neural Filter Training File

1. The Training file must have the name "match.txt";
2. Find the file, ".\neuralnetex1_match.txt", and save it to match.txt.

II. Operation

- Start the **ImageFinder**;
- Edge Filter: select Sobel 2;
- Threshold Filter: select Average Filter;

Set the	Avg	Filter
155	255	Dark Background
155	255	Dark Background
155	255	Dark Background

- CleanUp Filter: select Small;
- Reduction Filter: Default; Reduction Filter Parameter: Border Cut = 9
- BioFilter: select 6; BioFilter Parameter: default;
- Entering Data: click "Search Dir" and select any file in "neuralnetex1" directory;
- Converting to Records: click menu item "BioFilter/Scan Images - Directory Input";
- Neural Filter Training: click "NeuralFilter\Train (match.txt required)";
- N:N Matching: click "NeuralFilter/Query Set + Target Set/a1 + a1 ==> b1";
- Attrasoft Matches: go to the last line of b1.txt and see: "Total Number of Matches = 13575";
- Click the "File Input" button and select "b1_neuralnetex1_1.txt"; The **ImageFinder** will check each image to make sure it exists. Wait for a while until you see:

  Number of blocks = 17
  Block 0. Length = 84

- Click the menu item "NeuralNet/1:N File-Search" button to do the matching;
- **Results, 16 out of 17 are identified**.

10. Parameters

Chapters 3, 4, and 5 study the customized software. Chapters 7, 8 and 9 provide test-driving of the **ImageFinder**. This chapter studies the **ImageFinder** parameters.

Attrasoft ImageFinder looks at a sample jpg or gif images and then looks for similar images from a directory or file. The similar images are defined as images containing the sample segments, or:

- Translated segments;
- Rotated segments;
- Scaled segments;
- Rotated and Scaled segments;
- Brighter or Darker segments.

Definition:

- "**Search-Directory**" refers to a directory containing images to be searched.
- "**Search-File**" refers to a file containing images to be searched.
- "**Keys**" are images or image segments used to tell the software what to look for.
- **Training** or **Retraining** refers to teaching the software which images to look for. **Training** will first delete all old training and start to train the software from the beginning, while **retraining** will maintain all old keys.

10.1 Overview

To match an image, the **ImageFinder** pushes the image through many filters. For example, a set of filters could be:

Image Preprocessing
    Edge Filters;
    Threshold Filters; and
    Clean Up Filter.

Normalization
    Reduction Filter.

Feature Recognition
    BioFilter;
    Neural Filter.

Pixel Level Recognition
    NeuralNet Filter or ABM Filter.

Multi-layered Pixel Recognition
    BioFilter 2;
    NeuralFilter 2;
    ABM Filter 2.

…

Both image processing filters and normalization filters are preprocessing filters. The rest of the filters are recognition filters. Each recognition filter, in turn, will have two main clicks:

- Training; and
- Matching.

10.2 Filter Parameters

Many parameters and options of the **ImageFinder** are hidden. The users have only limited control of the parameters. Still, the **ImageFinder** has many parameters which can be adjusted by users. (The **ImageFinder** has 3000 parameters, of which the users can control approximately 70 in this version). **In a typical search, you will set these parameters and leave the other parameters with default values.**

There are 10 types of filters in this **ImageFinder** version**:**

    Edge Filters;
    Threshold Filters;
    Clean Up Filters;
    Reduction Filters;
    BioFilter;
    NeuralFilter.

ABM Filter;

BioFilter 2;
NeuralFilter 2;
ABM Filter 2.

Each type has many selections. For example,

Type = Edge Filter;
Selection = Sobel filter.

Another example is:

Type = Edge Filter;
Selection = Laplace Filter.

Each selection will have many parameters. The Main Problem is to select a set of parameters, which fits your specific problem.

The procedure is:

1. For each of the following types:

Edge Filters;
Threshold Filters;
Clean Up Filter;
Reduction Filter;
BioFilter;
NeuralFilter;
ABM Filter;

BioFilter 2;
NeuralFilter 2;
ABM Filter 2;

select a filter.

2. For the selected filter, specify the relevant parameters.

The **ImageFinder** has a large number of parameters. This version exposes approximately 70 parameters. Specifying a particular set of parameters for a given problem is Attrasoft's expertise. We are going to address each type of filter in this chapter.

10.3 Image Processing

Image Processing can make it or break it. For many problems like finger prints, palm prints, …, special image processing filters will be required.

Attrasoft ImageFinder learns an image in a way similar to human eyes:

- Ignore the background;
- Focus on an object in the image.

Image processing prepares the image for the **ImageFinder**. The image processing process is not unique; there are many options available. Some are better than others.

The main focus of choosing image processing filters is to **make** the **sample object(s) stand out, otherwise, change the options**.

If the image processing filters in the off-the-shelf **ImageFinder** are not sufficient, **a customized filter has to be built**. Do not make too many things stand out, i.e. as long as the area of interest stands out, the rest should show as little as possible.

The Image Pre-Processing Layer consists of three types of filters:

Edge Filters;
Threshold Filters; and
Clean Up Filters.

The **ImageFinder** applies these three filters in the above order.

The Edge Filters attempt to exaggerate the main feature(s) a user is looking for.

The Threshold Filters attempt to suppress the background.

The Clean-Up Filters will smooth the resulting image to reduce recognition error.

Each type of filter offers many selections, which we will study next. These three types of filters do not have parameters.

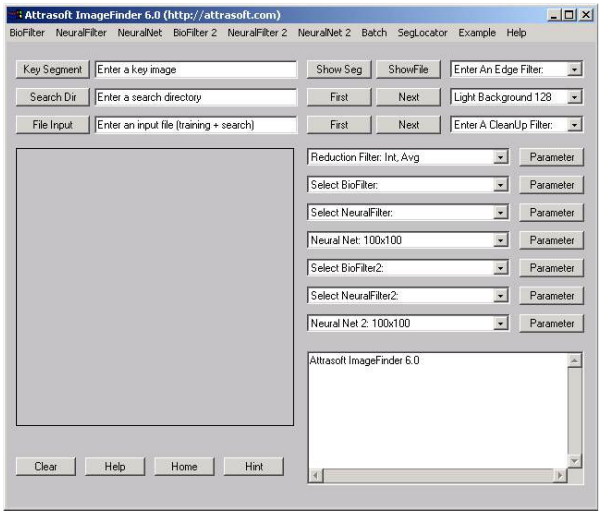


Figure 10.1 ImageFinder.

To select an Edge Filter, click the Edge Filter Drop Down List, which is the first List.
To select a Threshold Filter, click the Threshold Filter Drop Down List, which is the second List. To select a Clean Up Filter, click the Clean Up Filter Drop Down List, which is the third List.

10.3.1 Edge Filters

Edge Filters extract and enhance edges & contours in an image by expressing intensity differences (gradients) between neighboring pixels as an intensity value. The basic variables are the differences between the top and bottom rows; the differences between the left and right columns; and the differences between the center point and its neighbors.

Edge Filters have the following selections:

Code	Meaning
0	No Edge Filter
1	Sobel 1 (Prewitt)
2	Sobel 2 (Sobel)
3	Sobel 3
4	Sobel 4
5	Gradient
6	Gradient, 45°
7	Sobel 1, 45°
8	Sobel 1, - 45°
9	Laplacian 4
10	CD 11
11	FD 11
12	FD 9
13	FD 7
14	Laplacian 5
15	Laplacian 8
16	Laplacian 9
17	Laplacian 16
18	Laplacian17

All other filters have to be ordered in a Customized Version. These names really do not make any sense to common users; the best way to figure out what these filters are is to select a training image and try each of the filters. In general, these filters require the "Dark Background 128" Threshold Filter.

If you do not want to know the details, please skip the rest of this section. The details will be given below so you will know how to order a customized filter:

Sobel 1:

-1 0 1	-1 -1 -1
-1 0 1	0 0 0
-1 0 1	1 1 1

Sobel 2:

-1 0 1	-1 -2 -1
-2 0 2	0 0 0
-1 0 1	1 2 1

Sobel 3:

-1 0 1	-1 -3 -1
-3 0 3	0 0 0
-1 0 1	1 3 1

Sobel 4:

-1 0 1	-1 -4 -1

-4 0 4
-1 0 1

Gradient:

0 0 0
-1 0 1
0 0 0

Gradient, 45°

0 0 1
0 0 0
-1 0 0

Sobel 1, 45°

0 1 1
-1 0 1
-1 -1 0

Sobel 2, 45°

0 1 2
-1 0 1
-2 -1 0

Laplacian 4

0 -1 0
-1 4 -1
0 -1 0

Laplacian 5

0 -1 0
-1 5 -1
0 -1 0

Laplacian 8

-1 -1 -1
-1 8 -1
-1 -1 -1

Laplacian 9

-1 -1 -1
-1 9 -1
-1 -1 -1

Laplacian 16

0 0 -1 0 0
0 -1 -2 -1 0
-1 -2 16 -2 -1
0 -1 -2 -1 0
0 0 -1 0 0

Laplacian 17

0 0 0
1 4 1

0 -1 0
0 0 0
0 1 0

-1 0 0
0 0 0
0 0 1

1 1 0
1 0 -1
0 -1 -1

2 1 0
1 0 -1
0 -1 -2

0 0 -1 0 0
0 -1 -2 -1 0
-1 -2 17 -2 -1
0 -1 -2 -1 0
0 0 -1 0 0

10.3.2 Threshold Filters

After Edge Filters, the Threshold Filter will be applied to images. **Choose these two filters where the sample objects stand out, otherwise change the filters**. If no filter in this version fits your problem, **a Customized Filter has to be built**. DO NOT make too many things stand out, i.e. as long as the area of interest stands out, the rest should show as little as possible.

Once you make a selection, the objects in the images are black and the background is white (like a book, white paper, black print). **You should make the black area as small as possible, as long as it covers the key-segment(s). Otherwise, switch to a different background.**

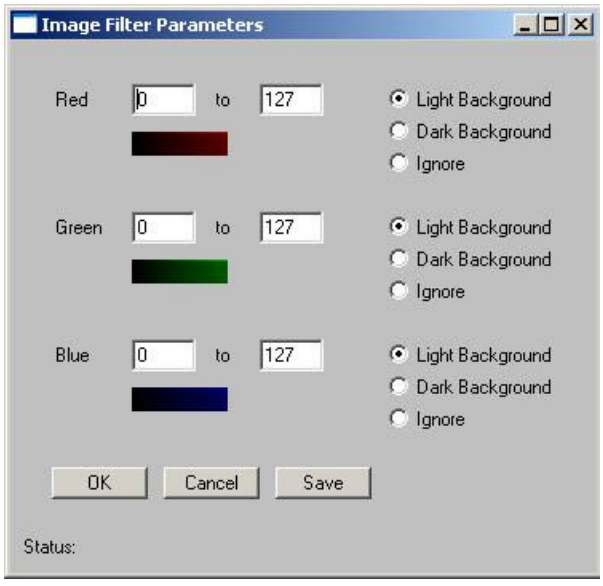


Figure 10.2 Threshold Filter Parameters.

There are 30 Threshold filters in the **ImageFinder**. A few filters, including the

average-filter and the customized-filter, allow you to specify any color range. Color is specified by three separate colors: Color = (red, green, blue). Each of the colors ranges from 0 to 255. (0, 0, 0) is black; (255, 255, 255) is white.

You should choose a filter where the sample object(s) stand out. You may want to know the meaning of the filters; example, "Light Background 128" means:

- "RGB Average in 0 – 127 " ➔ objects; and
- "RGB Average in 128 - 255" ➔ background.

To Summarize:

- **Choose an Edge Filter and a Threshold Filter where the sample object(s) stand out**;
- **Choose an Edge Filter and a Threshold Filter where the black area is as small as possible, as long as it covers the key-segment(s).**

10.3.3 Clean Up Filters

Clean Up Filters will clear noise off the image, but it will take more computation time.

10.3.4 Starting the Selection

If you are not familiar with image processing, please try the following:

Setting 1:

Edge Filters: Sobel 1 (or Sobel 2)
Threshold Filters: Dark Background 128
Clean Up Filter: Medium.

Setting 2:

Edge Filters: None ("Enter An Edge Filter")
Threshold Filters: Light Background 128
Clean Up Filter: None ("Enter A CleanUp Filter")

Beyond that, please use the trial and error approach to exaggerate the main features you are looking for and to suppress the background.

10.4 Normalization Filter

The Normalization Layer connects the image to the underlying neural nets. This layer has one type of filter, the Reduction Filter.

10.4.1 Reduction Filters

Let the underlying neural net be 100x100: if an image is larger than 100x100, say 350x230, then this image will be reduced to 100x100 or smaller.

When reducing images, a scaling factor can be introduced easily. Although the scaling symmetry can compensate for this scaling factor, the scaling symmetry is computationally expensive.

It is important to know that the Reduction Filter will match the selected underlying neural net, therefore, the behavior of the Reduction Filter not only depends on the selection of this filter itself, but also depends on the NeuralNet Filter chosen.

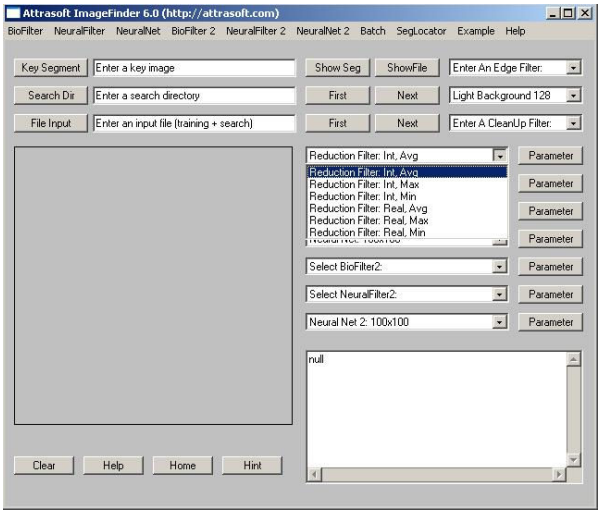


Figure 10.3 Selecting Reduction Filter.

There are several ways to reduce images:

- Integer;
- Real; or
- All images are reduced by the same amount.

Integer Reduction

Images are reduced by an integer factor to maximally fit 100x100 without distortion. For example, a 350x230 image will be reduced to 87x57.

Real Reduction

Images are reduced by a real number to maximally fit 100x100 without distortion. For example, a 350x230 image will be reduced to 100x65.

All

All training images and images in the search directory are reduced by the same integer to fit 100x100 without distortion.

Assume a given image is scaled into several sizes, say 350x230, 450x292, and 550x358, then the **Real Reduction** will reduce all three to a same size, 100x65. This option is available in customized versions.

When you choose **"All"-reduction**, all training images and images in the search directory are reduced by the same integer. The software will open the training image and images in the search directory to find the reduction factor, which will fit all images. If the image sizes vary significantly, (say some images are 100x100, and some images are 1000x1000), then the 1000x1000 image will be reduced to 100x100 and the 100x100 image will be reduced to 10x10. Reducing a 100x100 image to 10x10 can make image recognition difficult. Therefore in this case, this option can be used for searching the larger images (1000x1000), but not for the smaller images (100x100).

Within each type of reduction, there are 3 more settings. Assume a 3x3 pixel array is reduced to 1 pixel,

- Avg: Assign the average of the 3x3 pixels array to the new pixel;
- Max: Assign the maximum of the 3x3 pixels array to the new pixel; or
- Min: Assign the minimum of the 3x3 pixels array to the new pixel.

To select the Reduction Filter, use the fourth drop down list.

10.4.2 Parameters

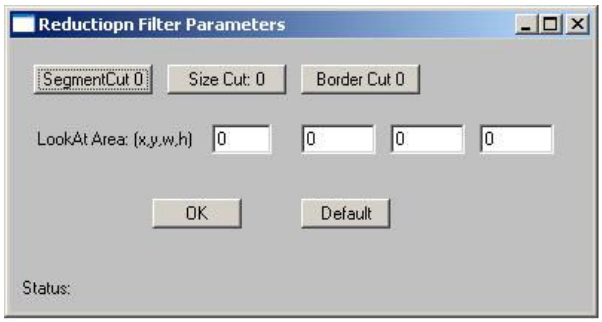


Figure 10.4 Reduction Filter Parameter.

The Reduction Filter has seven parameters. This section will explain these seven parameters. To set these parameters, click the "Parameter" button next to the Reduction filter; a new window will

pop up. Set the parameters in this new window (See Figure. 10.4).

Segment Cut

This parameter deals with the edges of the segments in the images. The Segment Cut parameter ranges from 0 to 12. The larger this parameter is, the smaller the segment the **ImageFinder** will use.

The possible settings in the user interface are: 0, 1, 2, .., and 12. To set the parameter, keep clicking the button; the setting will switch from one to the next each time you click the Blurring button.

Size Cut

In some applications, the users only want to search images of certain dimensions and ignore other images. An example is given below:

In this example, the two stamps belong to two different classes based on the image dimension alone.

The Size Cut parameter ranges from 0 to 9. To set the parameter, keep clicking the Size Cut button; the setting will switch from one to the next each time you click the button.

If the setting is 0, this parameter will be ignored. If the parameter is 1, then the longest edge of the image to be considered must be at least 100, but less than 199. If the parameter is 2, then the longest edge of the image to be considered must be at least 200, but less than 299; …

Border Cut

The Border Cut parameter ranges from 0 (no cut) to 9 (18% border cut). For some images (see the picture below), you might want to get rid of the sections of images close to the borders. To get rid of the border section, use the Border Cut.

The possible settings in the user interface are: 0, 1, 2, .., and 9. Assume an image is (0,0; 1,1), setting Border Cut to 1 means the **ImageFinder** will look at the section (0.02, 0.02; 0.98, 0.98); setting Border Cut to 2 means the **ImageFinder** will look at the section (0.04, 0.04; 0.96, 0.96); … . To set the parameter, keep clicking the button.

Look-At Area

The Look-At Area is the area the **ImageFinder** will use in a matching operation. A 100 x 100 window specifies a whole image. If an integer Reduction Filter is used, the actual area can be less than 100x100.

Four numbers specify the Look-At Area:

(x, y, w, h)

(x, y) are the coordinates of the upper-left corner and (w, h) are the width and height of the Look-At Window.

To use this Look-At Window, enter (x, y, w, h) to the 4 text boxes.

The image display area in the **ImageFinder** is 300x300; therefore, the training segment is specified within a 300x300 area. The Look-At Window is specified within a 100x100 area.

10.5 BioFilter

BioFilter is used for a quick study of a test set of images. There are 11 BioFilters in this version.

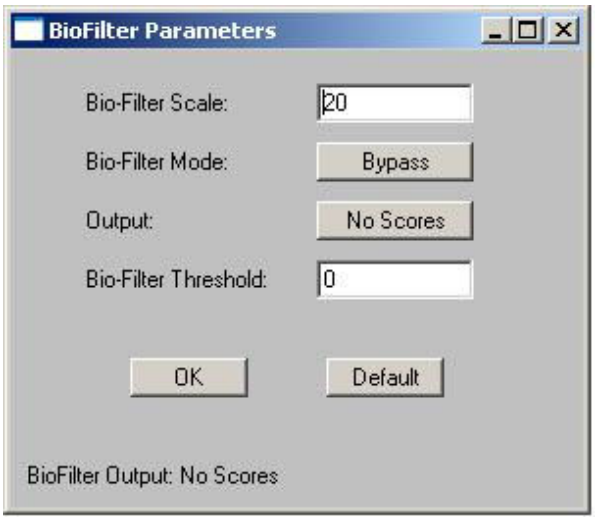


Figure 10.5 BioFilter Parameter.

BioFilter will do the following:

- Converting one image in the Input Space to one record in the Feature Space;
- Feature Space Matching.

To select a BioFilter, use the BioFilter Drop Down List.

The BioFilter has four parameters. This section will explain these four parameters. To set these parameters, click the "Parameter" button next to the BioFilter; a new window will pop up. Set the parameters in this new window (See Figure. 10.5).

Bio-Filter Scale

Use this parameter to control the amount of output. This parameter ranges from 0 to 100. The larger this number is, the more matches you will get. To set this parameter, enter a number between 0 and 100 to the text box.

Bio-Filter Mode

Since BioFilter is one of the recognition filters, you may or may not use this filter. This parameter decides whether the BioFilter will be used by other filters.

This parameter has three values:

Untrained
Trained
Bypass

The "Untrained" setting will do an image matching without training. The "Trained" setting requires the BioFilter be trained first. The "Bypass" setting will by pass this filter.

For the BioFilter, training requires two files, a1.txt and match.txt:

- A1.txt is the record file, which contains many records. Each image is converted into a record. A record represents features of an image in a feature space.
- Match.txt is a list of matching pairs. This file will teach the **ImageFinder** who will match with whom.

To set the parameter, keep clicking the button; the setting will switch from one to the next each time you click the Blurring button.

Output

To determine if one image "matches" another image, they must be compared using a unique

algorithm. Generally, the result of this comparison is a "score", indicating the degree to which a match exists. This score is then compared to a pre-set threshold to determine whether or not to declare a match.

The Output parameter has two settings:

No Scores
Scores

If the BioFilter is an intermediate step, this score will not show up in the output file. The “No Scores” setting (default setting) will not show the scores in the output file. If the BioFilter is the only filter used in the matching, then you can show the score in the output file by selecting the “Scores” setting.

To set the parameter, keep clicking the button; the setting will switch from one to the next each time you click the Blurring button.

BioFilter Threshold

The result of image comparison is a "score", indicating the degree to which a match exists. This score is then compared to a pre-set Threshold to determine whether or not to declare a match. This parameter sets the threshold.

To decide what threshold to use, you should make a test run first and look at the scores. Matching images have higher scores; unmatched images have lower scores. Select a threshold to separate these two groups. There will be a few images in the middle, representing both groups. Under these circumstances, the threshold selection depends on your applications.

To set the Threshold parameter, enter a number into the Threshold text box.

10.6 Neural Filters

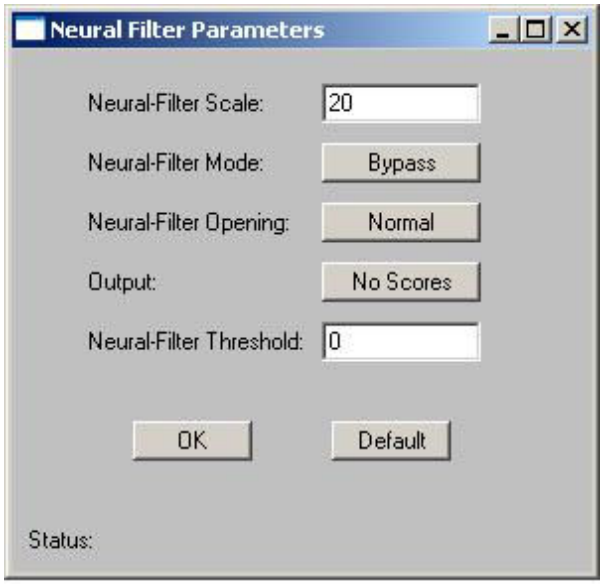


Figure 10.6 NeuralFilter Parameter.

Neural Filter is the main Feature Space Matching filter. There are 5 Neural Filters in this version. Neural Filter is very similar to the BioFilter in operation. Neural filter is more accurate than the BioFilter.

To select a Neural Filter, use the Neural Filter Drop Down List.

The Neural Filter has five parameters. This section will explain these five parameters. To set these parameters, click the “Parameter” button next to the Neural Filter; a new window will pop up. Set the parameters in this new window (See Figure. 10.6).

Neural-Filter Scale

Use this parameter to control the amount of output. This parameter ranges from 0 to 100. The larger this number is, the more matches you will get. To set this parameter, enter a number between 0 and 100 to the text box.

Neural-Filter Mode

Since Neural Filter is one of the recognition filters, you may or may not use this filter. This parameter decides whether the Neural Filter will be used by other filters.

This parameter has two values:

Trained
Bypass

The “Trained” setting requires the Neural Filter to be trained first, before matching. The “Bypass” setting will make other filters bypass this filter.

For the Neural Filter, training requires two files, a1.txt and match.txt:

- A1.txt is the record file, which contains many records. Each image is converted into a record. A record represents features of an image in a Feature Space.
- Match.txt is a list of matching pairs. This file will teach the **ImageFinder** who will match with whom.

To set the parameter, keep clicking the button; the setting will switch from one to the next each time you click the Blurring button.

Neural Filter Opening
This parameter controls the amount of the output. This parameter has 5 settings:

Very Large
Large
Normal
Small
Very Small

Large opening will allow more output than small opening. To set the parameter, keep clicking the button; the setting will switch from one to the next each time you click the Blurring button.

Output

To determine if one image "matches" another image, they must be compared using a unique algorithm. Generally, the result of this comparison is a "score", indicating the degree to which a match exists. This score is then compared to a pre-set threshold to determine whether or not to declare a match.

The Output parameter has two settings:

No Scores
Scores

If the Neural Filter is an intermediate step, this score will not show in the output file. The “No Scores” setting (default setting) will not show the scores in the output file. If the Neural Filter is the final filter used in the matching, then you can show the score in the output file by selecting “Scores”.

To set the parameter, keep clicking the button; the setting will switch from one to the next each time you click the Blurring button.

Neural Threshold

The result of image comparison is a "score", indicating the degree to which a match exists. This score is then compared to a pre-set threshold to determine whether or not to declare a match. This parameter sets the threshold.

To decide what threshold to use, you should make a test run first and look at the scores. Matching images have higher scores; unmatched images have lower scores. Select a threshold to separate these two groups. There will be a few images in the middle, representing both groups. Under these circumstances, the threshold selection depends on your application.

To set the Threshold parameter, enter a number into the Threshold text box.

10.7 NeuralNet Filter

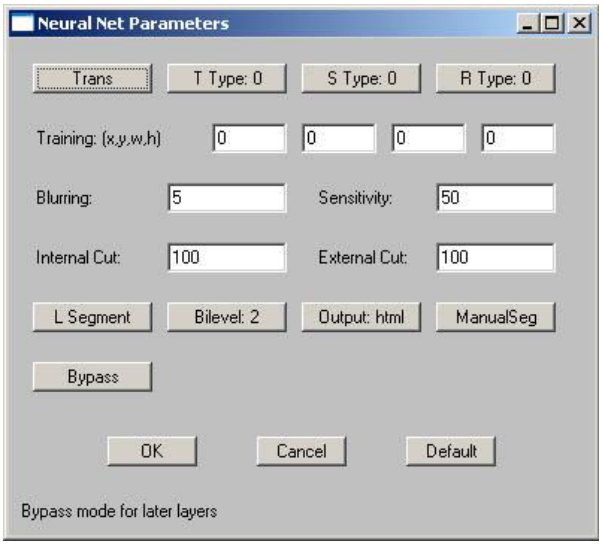


Figure 10.7 Neural Net Parameter.

If the Feature Space Recognition is not good enough, the **ImageFinder** will move into Input Space Recognition, which operates on image pixels. The NeuralNet filter, or the ABM Filter is the Input Space matching filter.

There are 6 NeuralNet Filters in this version. To select a NeuralNet Filter, use the NeuralNet Filter Drop Down List.

The default setting is 100x100. The matching speed crucially depends on the NeuralNet filter chosen. For example, if the filter is changed to 50x50, then the underlying neural net size is reduced by a factor of 4; and the neural computation speed will be decreased by a factor of 16.

The available filters are:

- 100x100 (Most Accurate)
- 90x90
- 80x80
- 70x70
- 60x60
- 50x50 (Least Accurate)

Let the speed of 100x100 filter be a base, then the overall speed for:

- 90x90 filter is 1 times faster;
- 80x80 filter is 1.6 times faster;
- 70x70 filter is 2.7 times faster;
- 60x60 filter is 5 times faster; and
- 50x50 filter is 10 times faster.

The NeuralNet Filter has 17 parameters. This section will explain these 17 parameters. To set these parameters, click the "Parameter" button next to the NeuralNet Filter; a new window will pop up. Set the parameters in this new window (See Figure. 10.7).

In the following, we explain these parameters in the order as they appear in the parameter window (Figure 10.7).

The first row has four buttons, which will be described from subsection 1 to 4. The second row has 4 text boxes to specify training segment, which will be described in subsection 5. The third row has two text boxes, Blurring and Sensitivity, which will be explained in subsection 6 and 7. The fourth row has two text boxes, Internal and External Weight Cut, which will be explained in subsection 8. After that each subsection explains one parameter

10.7.1 Symmetry

Symmetry or Invariance means similarity under certain types of changes. For example, considering two images, one with a face in the middle and the other with the face moved to the edge; we say these two images are similar because of the face.

Attrasoft ImageFinder can implement any symmetry (or combination of symmetries), which can be described by mathematics. However, symmetries are computationally expensive.

The symmetry defines "similar images". The **Attrasoft ImageFinder** supports five symmetry settings:

- No symmetry (0);
- Translation symmetry (3);
- Scaling symmetry (4);
- Rotation symmetry (5); and
- Rotation and Scaling symmetries (6).

The numbers are the codes in the batch file. Currently, Scaling symmetry and Oblique symmetry are the same. Other symmetries, or combination of symmetries, can be built for Customized Orders. Every symmetry setting has the translation symmetry, except "No Symmetry". In addition, each of the above settings support:

- Intensity symmetry.

Symmetries are computationally expensive, meaning it will take a longer time to do the job. You should use them only when they are required. To set the Symmetry, keep clicking the Symmetry button; the setting will switch from one to the next each time you click the button. The default setting in this version is Translation Symmetry.

For example, it seems that Stamp-Recognition requires translation and rotation symmetries. But because the edges of a stamp can be detected easily, the stamp can be rotated and shifted to a fixed position where the horizontal side is longer than the vertical side. All you need to do is recognize a stamp or an upside-down stamp. Therefore, stamp-recognition does not really require translation and rotation symmetries.

10.7.2 Translation Type

The Translation Type defines the accuracy of the translation symmetry. The Translation Type settings (and their codes) are:

- Most Accurate (0);
- Accurate (1); and
- Least Accurate (2).

To set the Translation Type, keep clicking the "T Type" button; the setting will switch from one to the next each time you click the button. The default setting is 0, the most accurate setting.

10.7.3 Scaling Type

The Scaling Type defines the accuracy of the scaling symmetry. The Scaling Type settings (and their codes) are:

- Least Accurate (0);
- Accurate (1);
- Accurate (2); and
- Most Accurate (3).

To set the Scaling Type, keep clicking the "T Type" button; the setting will switch from one to the next each time you click the button. The default setting is 0, the least accurate setting.

10.7.4 Rotation Type

The Rotation Type defines the accuracy of the Rotation symmetry. The Rotation Type settings (and their codes) are:

- 360° rotation, least accurate (0);
- -5° to 5° rotation (1);
- -10° to 10° rotation (2);
- 360° rotation, accurate (3);
- 360° rotation, more accurate (4);
- 360° rotation, most accurate (5).

To set the Rotation Type, keep clicking the "Rotation Type" button; the setting will switch from one to the next each time you click the button. The default setting is 360° rotation, least accurate (0).

10.7.5 Area of Interest (AOI)

Selecting an image segment is very important for training. Use image segments for searching **similar images**. Only use the whole image for **exact matches**.

Training requires an "Area of Interest" (AOI) or "Set Focus", which selects a key-segment. If an AOI is not chosen, the whole image is the AOI. Four numbers specify AOI: the upper-left corner coordinates, and the length and width. Once the segment specification is successful, a black box will cover the selected area. When you look at the training image, if the selected area is not what you want, just re-select the area again. To select the training image, use the "Key Segment" button.

Four text boxes specify AOI: (x y w h), where (x, y) is the upper-left corner, w is width, and h is height. All images are scaled to 300x300 for presentation on the **ImageFinder**. The maximum AOI is (0, 0) to (300, 300). The default setting is the whole image; the code is (x y w h) = (0 0 0 0). (0000) means ignore segment.

There are two situations where you should create a new sample image out of a sample segment:

- You repeatedly use an image segment;
- The image segment is not a rectangle; say a polygon.

The Windows Paint program will help you to create an image from a segment. When you create an image segment, please do not change the original image size. For example, if your image is 512x512 and you want create a segment of 400x200, please paste the 400x200 segment into a 512x512 empty image.

10.7.6 Blurring

This is one of the most important search parameters and the first parameter you should adjust.

Blurring compensates for minor image changes, which are not visible to human eyes. For example, if you use software to compress an image, to change the intensity of an image, or to translate, scale, or rotate an image, the image will be distorted a bit at the pixel level. You have to set "Blurring" to compensate for this.

The Blurring setting ranges from 0 to 50. The default setting is 5. "0%"-Blurring means the exact match. When the "Blurring" is increased, you will get more and more similar images. As the Blurring goes higher, the speed will slow a bit.

To set Blurring, enter a number between 0 and 50 to the Blurring text box. You should set the parameters in the following order:

> **Blurring, Internal Weight Cut, Sensitivity, External Weight Cut.**

To Summarize:

- **When a search yields no results, increase Blurring;**
- **When a search yields too many results, decrease Blurring.**

10.7.7 Sensitivity

The Sensitivity parameter ranges from 0 (least sensitive) to 100 (most sensitive). To search small segment(s), use high sensitivity search. To search large segment(s), use low sensitivity search. **The higher this parameter is, the more results you will get.**

The Sensitivity parameter ranges from 0 to 100. The default is 50.

To set the Sensitivity, enter a number between 0 and 100 to the Sensitivity text box. The

Sensitivity parameter controls the number of similar images retrieved by the **ImageFinder**.

To Summarize:

- **When a search yields no results, increase sensitivity;**
- **When a search yields too much result, decrease sensitivity.**

10.7.8 Internal/External Weight Cut

You can set the "**Internal Weight Cut**" (Internal Cut) or "**External Weight Cut**" (External Cut) to list only those retrieved images with scores or weight greater than a certain value (called threshold).

It is better to give no answer than a wrong answer. Assume you are searching images and all similar images have weights ranging from 1,000 to 10,000. It is possible that some other images pop up with weights ranging from 10 to 100. To eliminate these images, you can set the "External Weight Cut" to 1,000.

The Internal Cut plays a similar role as the External Cut. There are two differences between these two cuts:

- The Internal Cut ranges from 0 to 99; the External Cut can be any number;
- The Internal Cut stops the images from coming out, whereas the External Cut can bring the eliminated images back if you set the External Cut to 0. You might need to see the eliminated images sometimes for the purpose of adjusting parameters.

To set the Internal Cut, enter a number between 0 and 100 to the Internal Cut text box. To set the External Cut, enter a number to the External Cut text box.

To Summarize:

- **Set the "Internal Cut" or "External Cut" to eliminate errors.**

10.7.9 L/S Segments

The **ImageFinder** is currently tuned to search for large image segments (size of the whole image). It can look for small segments via "Small Segment" setting; however, only translation symmetry is supported for small segments. A Customized Version can be ordered for other symmetries.

To search large segments, use "L Segment" (Large Segment). To search small segments, use "S Segment" (Small Segment). For example, if a sample segment is one quarter of the sample image, it is a large segment. If the segment is 1/20 of the sample image, it is a small segment.

Currently, "S Segment" only supports translation symmetry. If you need rotation or/and scaling symmetry, please use "L Segment". Other symmetries can be added in a Customized Version.

To set the Large/Small Segment, keep clicking the Large or Small button; the setting will switch from one to the next each time you click the button.

10.7.10 Image Type

There are BW and Color images. For each of them, there are "sum-search", "maximum-search", and "average-search". This generates 6 image types:

- Bi-level 1 (0)
- Bi-level 2 (1)
- Bi-level 3 (2)
- Color 1 (3)
- Color 2 (4)
- Color 3 (5)

"Bi-level 1" is like an integration of function f (x); "Bi-level 2" is like a maximum value of f

(x); and "Bi-level 3" is the average of the above two.

"Bi-level 1" search will produce a higher weight than a "Bi-level 2" search. "Bi-level 3" search is in the middle. Similarly, a "Color 1" search will produce a higher weight than a "Color 2" search. "Color 3" is in the middle

To set the image type, keep clicking the Image Type button; the setting will switch from one to the next each time you click the Image Type button.

10.7.11 Output

The output can be a text file and html page. If this is an intermediate step, the output is usually a text page.

To set the output type, keep clicking the Output button; the setting will switch from one to the next each time you click the button.

10.7.12 Segment

The training segment can be specified in two ways:

Manual Specification
Automatic Specification

The default is Manual Specification. In this setting the segment will be specified by the four text boxes (x, y, w, h), as we discussed earlier.

Use image segments for searching **similar images**. Only use the whole image for **exact matches**. In general, the training segment is used rather than the whole image.

If you do not want to pick up a training segment, then let the **ImageFinder** pick up the segment for you by using the Automatic Specification. This parameter has 4 settings:

ManualSeg
AutoSeg 10
AutoSeg 20
AutoSeg 30

If the "AutoSeg 10" setting does not yield enough results, use "AutoSeg 20". If the "AutoSeg 20" setting does not yield enough results, use "AutoSeg 30". The reverse is also true. If the "AutoSeg 30" setting yields too many results, use "AutoSeg 20" or "AutoSeg 10".

10.7.13 Bypass

Since NeuralNet Filter is one of the recognition filters, you may or may not use this filter. This parameter decides whether the NeuralNet Filter will be used.

This parameter has two values:

Trained
Bypass

To set the Bypass, keep clicking the button; the setting will switch from one to the next each time you click the button.

10.7.14 Summary

The NeuralNet filter is hard to use because it has so many parameters. Not all parameters are equal. We divide the parameters into two groups. The beginners should use only parameters in the first group. Note that:

- **The most important parameters for Training are Image Processing, AOI, Symmetry, and Segment Cut.**
- **The most important parameters for Matching are Blurring, and Sensitivity.**
- Some parameters, like Segment Cut, will be used in both.

In a typical search, you will set these parameters and leave other parameters with default values. These are the 7 parameters you should focus first**:**

Training (3 parameters):

- Segment: selecting “AutoSeg 10” so the **ImageFinder** will select a training segment for you at the beginning.
- Symmetries
- Segment Cut

Matching (4 parameters):

- Sensitivity
- Blurring
- External Weight Cut
- Internal Weight Cut

Ignore these parameters at the beginning:

Training (7 parameters):

- Area of Interest (AOI): (x, y, w, h)
- Translation Type
- Scaling Type
- Rotation Type

Matching (3 parameters):

- Image Type
- Image Dimension
- Search Segment Size

The "Area of Interest" specifies a training segment, which is specified by 4 numbers: the coordinates of the upper-left corner, plus width and height. The default setting is the whole image.
You can set the “Segment” button (Default setting is “ManualSeg”) to “AutoSeg 10” to avoid choosing these 4 parameters.

Besides the above parameters, image-processing filters are also very important to training and matching.

If each parameter has 2 choices, 10 parameters together will have 1,000 settings; 20 parameters together will have 1,000,000 settings. Therefore, it is important to know how to use the software in order to get the desired results. By focusing only on 7 parameters, you can significantly reduce the complexity of operating this software.

11. Batch Job

When matching images, you will need to select many filters. For each selected filter, you will need to select many parameters.

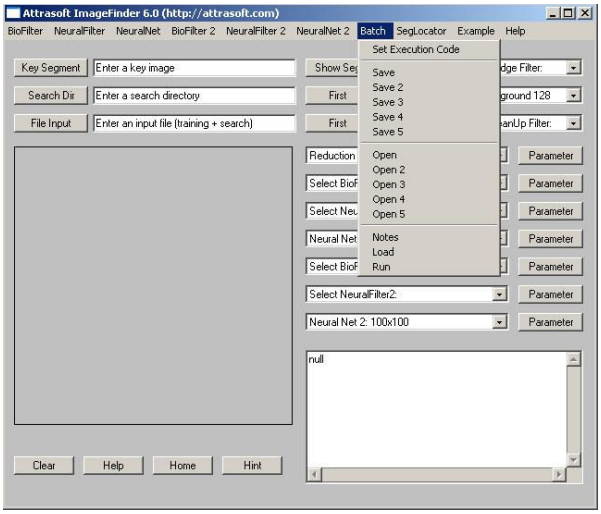


Figure 11.1 Batch Menu.

One simple approach is:

Edge Filters: Sobel 1 (or Sobel 2)
Threshold Filters: Dark Background 128
Clean Up Filter: None.
NeuralNet Filter:
Segment: AutoSeg 10
Segment Cut: Trial and Error
Symmetry: Default
Blurring: Start with 5, Trial and Error
Sensitivity: Start with 50, Trial and Error.

If this is your first matching and you do not like the default values, you will have go through a trial and error process**. However, if this is your second matching, you can save everything in the first matching and then use a batch command.** Click the "Batch/Save" menu command; you will get the batch file in the text area.

Throughout this chapter, we will use the two examples in the three previous chapters:

- Label Recognition Problem
- Logo Recognition Problem

11.1 Creating Batch Code

The Filter selection and Parameter setting can be saved in one of 5 files by the following commands:

Batch/Save
Batch/Save 2
Batch/Save 3
Batch/Save 4
Batch/Save 5

These 5 commands create the batch codes and save them to 5 different files. The batch codes can also be recalled later by clicking the following commands:

Batch/Open
Batch/Open 2
Batch/Open 3
Batch/Open 4
Batch/Open 5

These 5 commands open existing batch codes.

11.2 Sample Batch

A matching requires you specify the following Filters and their Parameters:

Image Preprocessing
Edge Filters;
Threshold Filters; and
Clean Up Filter.

Normalization
Reduction Filter.

Feature Recognition
BioFilter;
Neural Filter.

Pixel Level Recognition

NeuralNet Filter or ABM Filter.

Multi-Layered Pixel Recognition
BioFilter 2;
NeuralFilter 2;
ABM Filter 2.

A typical batch code looks like this:

```
[ImageFinder 6.0]
    executionCode=1001
[Input]
    trainFileName=None
    searchDirName=None
    fileInputName=None
[Image Processing Filters]
    edgeFilter=1
    thresholdFilter=1
    cleanUpFilter=1
[Reduction Filter]
    reductionType=0
    segmentCut=0
    sizeCut=0
    borderCut=0
    lookAtX=0
    lookAtY=0
    lookAtXLength=0
    lookAtYLength=0
[BioFilter]
    bioFilter=10
    bioFilterPercent=50
    bioFilterMode=2
    bioFilterFinal=0
    bioFilterCutOff=0
[NeuralFilter]
    neuralFilter=0
    neuralFilterPercent=20
    neuralFilterMode=0
    neuralFilterSize=2
    neuralFilterFinal=0
    neuralFilterCutOff=0
[Neural Net]
    neuralNetFilter=0
    segmentX=0
    segmentY=0
    segmentXlength=0
    segmentYLength=0
    symmetry=3
    rotationType=0
    translationType=0
    scalingType=0
    sensitivity=50
    blurring=5
    internalWeightCut=100
    externalWeightCut=100
    segmentSize=0
    imageType=1
    fileDisplayType=0
    autoSegment=0
    neuralNetMode=0
[BioFilter 2]
    bioFilter=0
    bioFilterPercent=20
    bioFilterMode=2
    bioFilterFinal=0
    bioFilterCutOff=0
[NeuralFilter 2]
    neuralFilter=0
    neuralFilterPercent=20
    neuralFilterMode=0
    neuralFilterSize=2
    neuralFilterFinal=0
    neuralFilterCutOff=0
[Neural Net 2]
    neuralNetFilter=0
    segmentX=40
    segmentY=40
    segmentXlength=220
    segmentYLength=220
    symmetry=3
    rotationType=0
    translationType=0
    scalingType=0
    sensitivity=70
    blurring=20
    internalWeightCut=90
    externalWeightCut=10000
    segmentSize=0
    imageType=1
    fileDisplayType=0
    autoSegment=0
    neuralNetMode=0
```

This batch code has the following sections:

Batch Execution Code
Input

Image Processing Filters
Reduction Filter
BioFilter
Neural Filter
NeuralNet Filter or ABM Filter

BioFilter 2
NeuralFilter 2
ABM Filter 2

When you create the batch code by command Batch/Save, you will see the above code in the text area. When you open a batch file by command Batch/Open, you will see the above code in the text area.

11.3 Overview

This section will explain how batch code is used.

(1) Create an application using the **ImageFinde**r;
(2) Save the setting to batch code with the following commands:

Batch/Save
Batch/Save 2
Batch/Save 3
Batch/Save 4
Batch/Save 5

You might find the following online note useful in helping you remember what you saved into these 5 batch files:

Batch/Notes

(3) Later you can open the batch file with the following commands:

Batch/Open

Batch/Open 2

Batch/Open 3

Batch/Open 4

Batch/Open 5

(4) To load the parameter without running, click:
Batch/Load.

(5) To load the parameter and run, click:
Batch/Run.

The batch/Save command saves the following information:

- Filter selection and their Parameter settings;
- The template file, which contains the records from images.

11.4 Batch Execution Code

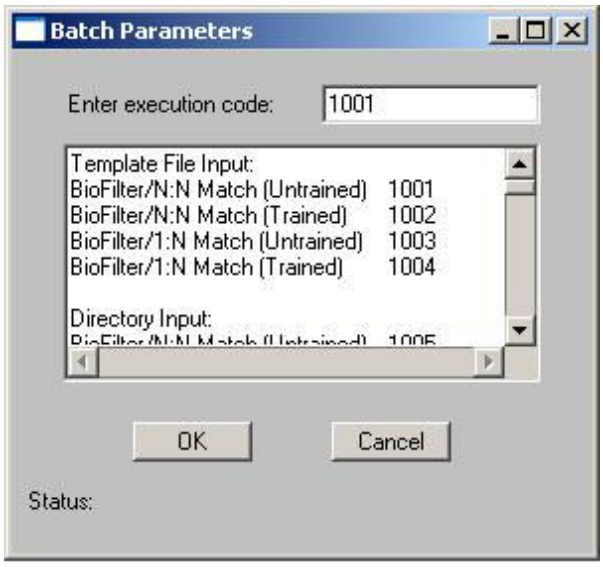


Figure 11.2 Execution Code Window.

There are many commands in the **ImageFinder.** Each command has an integer for identification. This integer is called Batch Execution Code. The "Batch/Run" command uses this code to run the command specified by the batch file. To find the batch code for each command, click:

Batch/Set Execution Code

You will see a textbox and the following codes:

Template File Input:
BioFilter/N:N Match (Untrained) 1001
BioFilter/N:N Match (Trained) 1002
BioFilter/1:N Match (Untrained) 1003
BioFilter/1:N Match (Trained) 1004

Directory Input:
BioFilter/N:N Match (Untrained) 1005
BioFilter/N:N Match (Trained) 1006
BioFilter/1:N Match (Untrained) 1007
BioFilter/1:N Match (Trained) 1008

File Input:
BioFilter/N:N Match (Untrained) 1009
BioFilter/N:N Match (Trained) 1010
BioFilter/1:N Match (Untrained) 1011
BioFilter/1:N Match (Trained) 1012

Template File Input:
NeuralFilter/N:N Match 1013
NeuralFilter/N:(N-1) Match 1014
NeuralFilter/1:N Match 1015

Directory Input:
NeuralFilter/N:N Match 1016
NeuralFilter/N:(N-1) Match 1017
NeuralFilter/1:N Match 1018

File Input:
NeuralFilter/N:N Match 1019
NeuralFilter/N:(N-1) Match 1020
NeuralFilter/1:N Match 1021

Template File Input:
NeuralFilter/a1 + a2 ==> b2 1022
NeuralFilter/a1 + a3 ==> b3 1023
NeuralFilter/a1 + a4 ==> b4 1024
NeuralFilter/a1 + a5 ==> b5 1026

Neural Net:
NeuralNet/1:N Search 1027
NeuralNet/1:N Search+Sort 1028
NeuralNet/N:N Match 1029
NeuralNet/1:N Long-Search 1030
NeuralNet/1:N Long-Search+Sort 1031
NeuralNet/1:N File-Search 1032
NeuralNet/1:N File-Search+Sort 1033
NeuralNet/N:N File Match 1034

BioFilter 2:
BioFilter 2/1:N Match 1035
BioFilter 2/N:N Match 1036

Neural Filter 2:
NeuralFilter 2/1:N Match 1037
NeuralFilter 2/N:N Match 1038

Neural Net 2:
NeuralNet 2/1:N Match 1039

The default batch code is 1001. **You must specify the Batch Execution Code for your batch files**. The easiest way is:

- Click Batch/Set Execution Code;
- Enter the Batch Execution Code to the text box and click OK button.

You can also make changes directly in the batch files. The batch files are abm60.txt, abm60_2.txt, abm60_3.txt, abm60_4.txt, abm60_5.txt.

In this chapter, we will reproduce the examples in the early chapters.

11.5 BioFilter Examples

The Batch Execution can take three types of input:

- Template Input
- File Input
- Directory Input

For the Template Input, the template file is loaded, so the **ImageFinder** does not convert images to records, but rather works on the

saved records directly. For the File Input and Directory Input, the template file is not loaded, so the **ImageFinder** will first convert images to records. The **ImageFinder** will scan through all images in the input file or the input directory.

There are 8 examples in this section:

N:N Match, Untrained, Label Template

N:N Match, Trained, Label Template

1:N Match, Untrained, Label Template

1:N Match, Trained, Label Template

1:N Match, Trained, Label File

N:N Match, Trained, Label File

1:N Match, Trained, Label Directory

N:N Match, Trained, Label Directory

The first 4 examples use Template Input, which is fast because the conversion from Input Space to Feature Space is already done. This will be followed by 2 examples for File Input and 2 examples for Directory Input.

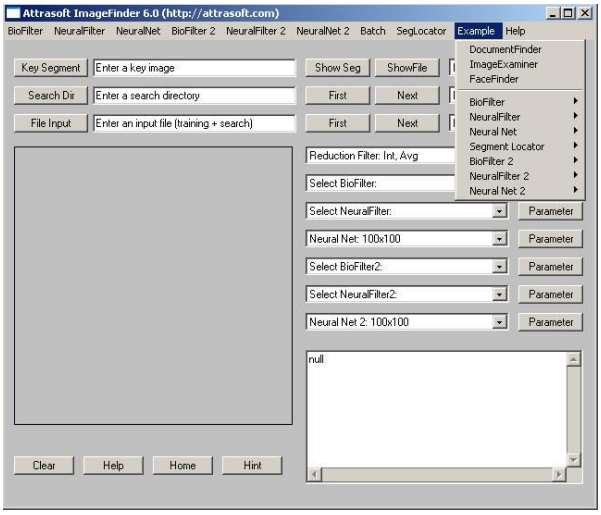


Figure 11.3 Example Menu.

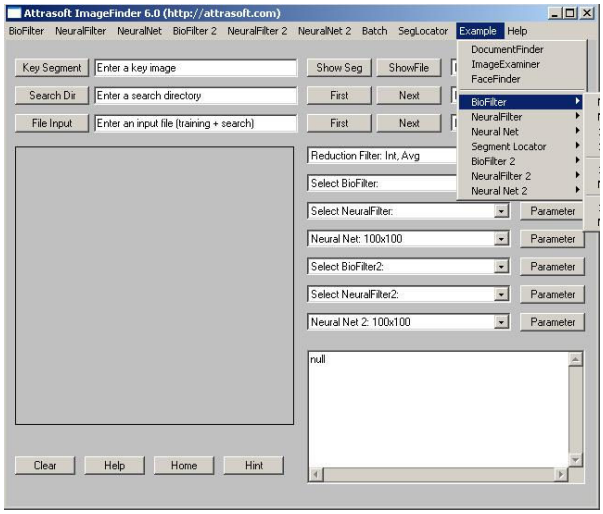


Figure 11.4 Example/BioFilter Menu.

Example 1. Untrained N:N Matching, Template Input.

This is the Label Recognition example. This example uses Untrained N:N Matching.

- Click "Example\BioFilter\N:N Match, Untrained, Label Template";
- Click: Batch\Run.

The result is the same as BioFilter Untrained N:N matching. In this example, the template file is loaded, so the **ImageFinder** does not convert images to records, but rather works on the saved records directly.

Example 2. Trained N:N Matching, Template Input.

This is the Label Recognition example. This example uses Trained N:N matching.

- Click "Example\BioFilter\N:N Match, Trained, Label Template";
- Click: Batch\Run.

The result is the same as BioFilter Trained N:N matching. In this example, the template file is loaded, so the **ImageFinder** does not convert images to records, but rather works on the saved records directly.

Example 3. Untrained 1:N Matching, Template Input.

This is the Label Recognition example. This example uses untrained 1:N matching.

- Click "Example\BioFilter\1:N Match, Untrained, Label Template";
- Click: Batch\Run.

The result is the same as BioFilter Untrained 1:N matching. In this example, the template file is loaded, so the **ImageFinder** does not convert images to records, but rather works on the saved records directly. However, the sample image is not converted into records in advance, so the **ImageFinder** will first convert the sample image into a record and then, make a 1:N matching.

Example 4. Trained 1:N Matching, Template Input.

This is the Label Recognition example. This example uses Trained 1:N matching.

- Click "Example\BioFilter\1:N Match, Trained, Label Template";
- Click: Batch\Run.

The result is the same as BioFilter Trained 1:N matching. In this example, the template file is loaded, so the **ImageFinder** does not convert images to records, but rather works on the saved records directly. However, the sample image is not converted into a record in advance; so the **ImageFinder** will first convert the sample image into a record and then, make a 1:N matching.

Example 5. Trained 1:N Matching, File Input.

This is the Label Recognition example. This example uses Trained 1:N matching and uses the input file.

- Click "Example\BioFilter\1:N Match, Trained, Label File";
- Click: Batch\Run.

This example uses an input file, biofilterex1_input1.txt. This input file lists the first 4 pairs of images in the label recognition problem.

The result is the same as BioFilter Trained 1:N matching. In this example, the template file is not loaded, so the **ImageFinder** will first convert images to records. There are 8 images in this example; you will see the **ImageFinder** scan through these 8 images. After that, the **ImageFinder** will match the sample image with the 8 images.

Example 6. Trained N:N Matching, File Input.

This is the Label Recognition example. This example uses Trained N:N matching and uses the input file.

- Click "Example\BioFilter\N:N Match, Trained, Label File";
- Click: Batch\Run.

This example uses an input file, biofilterex1_input1.txt. This input file lists the first 4 pairs of images in the label recognition problem.

The result is the same as BioFilter Trained N:N matching. In this example, the template file is not loaded, so the **ImageFinder** will first convert images to records. There are 8 images in this example; you will see the **ImageFinder** scan through these 8 images. After that, the **ImageFinder** will match each image with the 8 images.

Example 7. Trained 1:N Matching, Directory Input.

This is the Label Recognition example. This example uses Trained 1:N matching and uses the directory file.

- Click "Example\BioFilter\1:N Match, Trained, Label Directory";
- Click: Batch\Run.

This example uses an input directory, ".\biofilterex2\", where ".\" is the **ImageFinder** directory. This input directory lists the first 4 pairs of images in the label recognition problem.

The result is the same as BioFilter Trained 1:N matching. In this example, the template file is not loaded, so the **ImageFinder** will first convert images to records. There are 8 images in this example; you will see the **ImageFinder** scan through these 8 images. After that, the **ImageFinder** will match the sample image with the 8 images.

Example 8. Trained N:N Matching, Directory Input.

This is the Label Recognition example. This example uses Trained N:N matching and uses the directory file.

- Click "Example\BioFilter\N:N Match, Trained, Label Directory";
- Click: Batch\Run.

This example uses an input directory, ".\biofilterex2\", where ".\" is the **ImageFinder** directory. This input directory lists the first 4 pairs of images in the label recognition problem.

The result is the same as BioFilter Trained N:N matching. In this example, the template file is not loaded, so the **ImageFinder** will first convert images to records. There are 8 images in this example; you will see the **ImageFinder** scan through these 8 images. After that, the **ImageFinder** will match each image with the 8 images.

11.6 NeuralFilter Examples

The Batch Execution can take three types of input:

- Template Input
- File Input
- Directory Input

For the Template Input, the template file is loaded, so the **ImageFinder** does not convert images to records, but rather works on the records directly. For the File Input and Directory Input, the template file is not loaded, so the **ImageFinder** will first convert images to records. The **ImageFinder** will scan through all images in the input file or the input directory.

There are 10 examples in this section:

N:N Match, Label Template
N:N-1 Match, Label Template
1:N Match, Label Template
N:N Match, Label File
N:N-1 Match, Label File
1:N Match, Label File
N:N Match, Label Directory
N:N-1 Match, Label Directory
1:N Match, Label Directory
N:N Match, Logo Template

The first 9 examples are for the Label Identification problem and the last one is for Logo Identification. The results will match the Neural-Filter chapter and the NeuralNet Filter chapter.

The first 3 examples use Template Input, which is fast because the conversion from Input Space to Feature Space is already done. This will be followed by 3 examples for File Input and 3 examples for Directory Input.

Example 1. N:N Matching, Template Input.

This is the Label Recognition example. This example uses N:N Matching.

- Click "Example\NeuralFilter\N:N Match, Label Template";
- Click: Batch\Run.

The result is the same as NeuralFilter N:N Matching. In this example, the template file is loaded, so the **ImageFinder** does not convert images to records, but rather works on the saved records directly.

Example 2. N:N-1 Matching, Template Input.

This is the Label Recognition example. This example uses N:N-1 Matching.

- Click "Example\NeuralFilter\N:N-1 Match, Label Template";
- Click: Batch\Run.

The result is the same as NeuralFilter N:N-1 Matching. In this example, the template file is loaded, so the **ImageFinder** does not convert images to records, but rather works on the saved records directly.

Example 3. 1:N Matching, Template Input.

This is the Label Recognition example. This example uses 1:N Matching.

- Click "Example\NeuralFilter\1:N Match, Label Template";
- Click: Batch\Run.

The result is the same as NeuralFilter 1:N matching. In this example, the template file is loaded, so the **ImageFinder** does not convert images to records, but rather works on the records directly.

Example 4. N:N Matching, File Input.

This is the Label Recognition example. This example uses N:N Matching.

- Click "Example\NeuralFilter\N:N Match, Label File";
- Click: Batch\Run.

This example uses an Input File, biofilterex1_input1.txt. This input file lists the first 4 pairs of images in the label recognition problem. The result is the same as NeuralFilter N:N Matching. In this example, the template file is not loaded, so the **ImageFinder** will first convert images to records. There are 8 images in this example; you will see the **ImageFinder** scan through these 8 images.

Example 5. N:N-1 Matching, File Input.

This is the Label Recognition example. This example uses N:N-1 Matching.

- Click "Example\NeuralFilter\N:N-1 Match, Label File";
- Click: Batch\Run.

This example uses an input file, biofilterex1_input1.txt. This input file lists the first 4 pairs of images in the label recognition problem. The result is the same as NeuralFilter N:N-1 Matching. In this example, the template file is not loaded, so the **ImageFinder** will first convert images to records. There are 8 images in this example; you will see the **ImageFinder** scan through these 8 images.

Example 6. 1:N Matching, File Input.

This is the Label Recognition example. This example uses 1:N Matching.

- Click "Example\NeuralFilter\1:N Match, Label File";
- Click: Batch\Run.

This example uses an input file, biofilterex1_input1.txt. This input file lists the first 4 pairs of images in the label recognition

problem. The result is the same as NeuralFilter 1:N Matching. In this example, the template file is not loaded, so the **ImageFinder** will first convert images to records. There are 8 images in this example; you will see the **ImageFinder** scan through these 8 images.

Example 7. N:N Matching, Directory Input.

This is the Label Recognition example. This example uses N:N Matching.

- Click "Example\NeuralFilter\N:N Match, Label Directory";
- Click: Batch\Run.

This example uses an input directory, ".\biofilterex2\", where ".\" is the **ImageFinder** directory. This input directory lists the first 4 pairs of images in the label recognition problem. The result is the same as NeuralFilter N:N matching. In this example, the template file is not loaded, so the **ImageFinder** will first convert images to records. There are 8 images in this example; you will see the **ImageFinder** scan through these 8 images.

Example 8. N:N-1 Matching, Directory Input.

This is the Label Recognition example. This example uses N:N-1 Matching.

- Click "Example\NeuralFilter\N:N-1 Match, Label Directory";
- Click: Batch\Run.

This example uses an input directory, ".\biofilterex2\", where ".\" is the **ImageFinder** directory. This input directory lists the first 4 pairs of images in the label recognition problem. The result is the same as NeuralFilter N:N-1 Matching. In this example, the template file is not loaded, so the **ImageFinder** will first convert images to records. There are 8 images in this example; you will see the **ImageFinder** scan through these 8 images.

Example 9. 1:N Matching, Directory Input.

This is the Label Recognition example. This example uses 1:N Matching.

- Click "Example\NeuralFilter\1:N Match, Label Directory";
- Click: Batch\Run.

This example uses an input directory, ".\biofilterex2\", where ".\" is the **ImageFinder** directory. This input directory lists the first 4 pairs of images in the label recognition problem. The result is the same as NeuralFilter N:N Matching. In this example, the template file is not loaded, so the **ImageFinder** will first convert images to records. There are 8 images in this example; you will see the **ImageFinder** scan through these 8 images.

Example 10. N:N Matching, Logo Identification, Template Input.

This is the Logo Recognition example. This example uses N:N Matching.

- Click "Example\NeuralFilter\N:N Match, Logo Template";
- Click: Batch\Run.

11.7 NeuralNet Filter Examples

This section has 2 Logo Recognition examples, 1:N Matching via file input and N:N Matching via file input. The Logo Recognition example in the last chapter is 1:N Match. We will also use the same example for N:N Matching.

Example 1. 1:N Matching, File Input.

This is the Logo recognition example. This example uses 1:N Matching.

- Click "Example\Neural Net\1:N Match, Logo, File";
- Click: Batch\Run.

The result is the same as NeuralNet Filter 1:N Matching.

In this example, the input file is the output file of the Neural Filter, so the results cover both Feature Space recognition (Neural Filter) and Input Space recognition (NeuralNet Filter).

Example 2. N:N Matching, File Input.

This is the Logo Recognition example. This example uses N:N Matching.

- Click "Example\Neural Net\N:N Match, Logo, File";
- Click: Batch\Run.

In this example, the input file is the output file of the Neural Filter, so the results cover both Feature Space recognition (Neural Filter) and Input Space recognition (NeuralNet Filter). The input file has many blocks of data; in each block, the first line is the Neural Filter input, and the rest of the lines are Neural-Filter output. The N:N Matching will only process the first block and ignore the rest of the data. The **ImageFinder** will match each image in the first block against all other images in the first block.

12. NeuralNet Filter And Short-Search

This chapter will focus on the NeuralNet Filter and Short-Search. In all of the previous chapters, we used the output of the BioFilter and Neural Filter to feed the NeuralNet Filter. In this chapter, we will not use the Feature Space Filters, i.e. BioFilter and NeuralFilter. The purpose of this chapter is to see the how NeuralNet Filter works alone and learn the NeuralNet Filter commands.

The distinction between Identification and Search is their Outputs:

Identification

> Identification is a one-to-many (1:N) matching of a single sample set against a database of samples. The single image is generally the newly captured sample and the database contains all previously enrolled samples. Scores are generated for each comparison, and an algorithm is used to determine the matching record, if any. Generally, the highest score exceeding the threshold results in Positive Identification.

Search or Retrieval

> Search is similar to Identification, i.e. 1:N Matching; however, the result is a set of possible matching images, not a classification. "Identification" returns a classification, while "Search" returns multiple matched images.

12.1 Overview

The NeuralNet Filters, like BioFilters and Neural Filters, operate in two phases:

- Training; and
- Search.

This software only has 2 main Input Parameters:

1. **The Keys**: key-image(s), or key-segment(s) used to tell this software what to look for.
2. **The Search-Directory**: images you want to search through.

Several clicks can specify these two parameters. Keys are fed into the software for training, i.e. teaching the NeuralNet Filters what to look for. After that, the **ImageFinder** will be ready to select similar images.

Attrasoft ImageFinder learns an image in a way similar to human eyes:

- Ignore the background;
- Focus on an object in the image.

Apart from the parameters, here is what you need to do:

1. Enter key-segments into the **ImageFinder** (keys are used to teach the NeuralFilter what to look for);
2. Click the “NeuralNet/NeuralNet Train” command to teach the NeuralNet what to look for.
3. Save all the images you want to look through into a directory (search-directory) and enter it into the software;
4. Click the “NeuralNet/1:N Search” command --- the NeuralNet Filter is now looking through the images.
5. The Output is a web page or text file, which is a list of names and weights (scores):
   - The weight of an image is related to the characteristics you are looking for (the weight is similar to an Internet search engine score);
   - Click the link of each image and an image will pop up on the screen.

12.2 Parameters

The NeuralNet Filter does require the following filters:

Image Preprocessing
Edge Filters;
Threshold Filters; and
Clean Up Filter.

Normalization
Reduction Filter.

These filters will need to be set. The NeuralNet Filters will be divided into Training and Search Phase. The parameters are:

1. Training
   - Set Symmetry
   - Set Translation Type
   - Set Scaling Type
   - Set Rotation Type
   - Set Reduction Type
   - Set Border Cut
   - Set Segment Cut

2. Retrieving
   - Set Blurring
   - Set Sensitivity
   - Set Internal Weight Cut
   - Set External Weight Cut
   - Set Segment Size

If you want, you can save the NeuralNet Filter parameter settings in a batch file.

12.3 Short-Search, Long-Search, and File-Search

The NeuralNet Filter commands are divided into three types: Short, Long, and File. We introduced File-Search earlier. In this chapter, we will introduce Short-Search, and the next chapter will introduce Long-Search.

The Short-Search uses directory input. The Short-Search will not go to sub-directories. **The limit for Short-Search is 1,000 images**. All images to be searched must be in one directory, the search directory. All images in the sub-directories of the search directory will not be included in "Short-Search".

There is no technical limit for the Long-Search. **Long-Search can search millions of images.** In the Long-Search, the search directory can have many sub-directories. In this version, the default number is 3,000 sub-directories. All sub-directories must be only one level deep, i.e. the sub-directory cannot have other sub-directories. All images to be searched must be in the sub-directories. Each sub-directory can have up to 1,000 images.

If your search directory has 3,000 sub-directories and each sub-directory has 1,000 images, then you can search 3,000,000 images.

There is no limit for the File-Search. **File-Search can search any number of directories with any number of images.** File-Search Does require the additional work of preparing the input file. The input file lists one image per line.

12.4 ImageFinder Operations for Short-Search

The Search procedure is:

- Training;
- Retrieving.

In a typical search, you will set some parameters and leave other parameters with default values. The Image Processing Filters and the Reduction Filter are important for the NeuralNet Filters. For the NeuralNet Filter, not all parameters are equal:

- **The most important parameters for training are AOI, Segment Cut, and Symmetry.**
- **The most important parameters for searching are Blurring, Sensitivity.**

Step 0 Preprocessing:

Choose the three image processing filters where the sample objects will stand out;
Choose the three image processing filters where the black area is as small as possible, as long as it covers the key-segment(s).

Example. Choose "Light Background 128".

Figure 12.1 ".\Uspto\IMAGE036.JPG".

Step 1. Sample Image.

Example.
To select ".\Uspto\IMAGE036.JPG", click the "**Key Segment**" button; then choose the file.

Step 2. Training.

2.1 **Set Focus: Select a Segment**.

The simplest way is to click the Segment button and set it to "AutoSeg 10". If no segment is chosen, the whole image will be used. Use image segments for searching similar images. Only use the whole image for exact matches.

<u>There are two situations where you should create a new sample image out of a sample segment:</u>

- You repeatedly use an image segment;
- The image segment is not a rectangle; say a polygon.

2.2 **Symmetry**:

Your options are:
No symmetry;
Translation symmetry;
Rotation symmetry;
Scaling symmetry, Oblique symmetry; and
Rotation and Scaling symmetries.

2.3 Set Segment Cut.

2.4 Click the "NeuralNet/NeuralNet Train" command:

By repeating step 1 and step 2, you can train the software with as many image segments as you wish, provided the memory used is less than your RAM. **Use the Training button for the first segment; use the Retraining button for the second, third, ... , segments.**

Step 3. Search Directory.

Example. To select: ".\Uspto\, click the "**Search Dir**" button; **then click any file in** ".\Uspto"**.**

Step 4. Search.

The most important parameters for searching are Blurring and Sensitivity. In a typical search, you will set these parameters and leave other parameters with the default values.

4.1 Sensitivity

When the default setting yields no results, increase Sensitivity;
When the default setting yields too many results, decrease Sensitivity.

4.2 Blurring

When a search yields no results, increase Blurring;
When a search yields too many results, decrease Blurring.

4.3 Internal / External Weight Cut

To list only those retrieved images with weights greater than a certain value, you can set the "External Weight Cut" or "Internal Weight Cut".

4.5 Click the "NeuralNet/1:N Search" command.

Step 5. Results.

See the results in the web page. You might need to click the "Refresh" button. The results are not sorted. **If you want to sort the results, click** the "NeuralNet/Sort" command.

Finally, if you want to save the search results, click the "Batch/Save" command; the batch file will be generated in the text area and saved into a file. You can save up to 5 batch files by selecting 1 of 5 files to save. To recall a file, use a "Batch/Open" commands. If you want to save the code to your own file, highlight the code, hit "Ctrl/C", then go to Window's notepad and hit "Ctrl/V" to generate the batch file.

12.5 ImageFinder Operations for Short-Search (Advanced)

For the advanced users, there are additional options to increase the matching accuracy. This section adds more options to the last section.

Step 0 Preprocessing

To make the optimal selection, you can experiment with different combinations of Edge Filters and Threshold Filters.

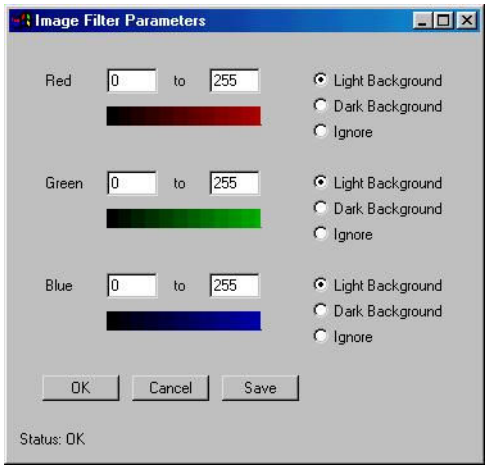


Figure 12.2 Nine Filter Parameters.

There are two special Threshold Filters: "Average" and "Customized". These Threshold Filters provide you more control of the image preprocessing before the images enter the NeuralNet Filter**.** Each color has 3 variables: 2 variables for range and 1 variable for type. The 9 Filter Variables are:

Red Range: [r1, r2];
Red Type: Light Background /Dark Background / Ignore;
Green Range: [g1, g2];
Green Type: Light Background /Dark Background/ Ignore;
Blue Range: [b1, b2];
Blue Type: Light Background /Dark Background/ Ignore.

After setting the variables, click the Save button, and go back to the **ImageFinder** to see the training image. If the background filter is not satisfied, set the parameters again and click the Save button.

To explain what these filters are, we have to dig into technical details, which is beyond the scope of this menu. We encourage you to experiment: select a key image and try each filter.

Step 2. Training.

Set Translation Type
Set Rotation Type

Set Scaling Type
Set Border Cut

Step 3. NA

Step 4. Search.

Set Image Type
"Bi-level 1" (Integration) search will produce a higher weight than a "Bi-level 2" (Maximum) search. Similarly, a "Color 1" search will produce a higher weight than a "Color 2" search.

Set Large/Small Size
To search large segments, use "L Segment" (Large Segment).
To search small segments, use "S Segment" (Small Segment).

Step 5. Results.

Select text or html output.

12.6 Trade Mark Retrieval

The images used in this example are from:
FY 1999 USPTO Annual Report,
http://www.uspto.gov/web/offices/com/annual/1999/

In this section, we will identify 5 trademarks. In particular, we will try to demonstrate the symmetry parameter of the NeuralNet Filter. Symmetry means objects in images have been changed, such as moved to a different place (Translation Symmetry), or enlarged (Scaling Symmetry), or rotated (Rotation Symmetry). The first two examples demonstrate Rotation symmetry, the next two examples demonstrate Scaling symmetry, and the last example demonstrates combined Rotation and Scaling symmetries. All examples have Translation symmetry.

12.6.1 United Way - Rotation Symmetry

There are two ways to run this example:

- Batch
- Manual

The Batch Run takes only two clicks:

Click “Example/Neural Net/United Way - R”
Click “Batch/Run”.

The Manual Run requires a few more clicks:

Input:

Training: .\uspto\image036.jpg
Search: .\uspto\

Parameters

Edge Filter: None
Threshold Filter: Light Background 192
NeuralNet Filter:
Symmetry: Rotation Symmetry
Blurring = 18
Sensitivity = 25
Internal Cut = 40

Operation

- Click the “Key Segment” button and select “.\uspto\image036.jpg”;
- Click “Search Dir” button and select “.\uspto\”;
- Set the Threshold Filter to Light Background 192;
- Click the NeuralNet Filter Parameter button and set:
  Symmetry: Rotation Symmetry
  Blurring = 18
  Sensitivity = 25
  Internal Cut = 40
- Click “NeuralNet/Train” button to train the filter;
- Click “NeuralNet/1:N Search” button to make a search;
- The result is in a web page;
- If you want to sort, click the “Sort” button.

Results

I036_r10.jpg 104064
I036_r20.jpg 85824
I036_r30.jpg 115328
I036_r40.jpg 77632
I036_r50.jpg 70208
I036_r60.jpg 96384
I036_r70.jpg 98176
I036_r80.jpg 109312
I036_r90.jpg 91520
IMAGE036.JPG 128000000

Summary

# Images	= 126
# To be retrieved	= 10
# Retrieved Correctly	= 10
# Missed	= 0
Hit Ratio	**= 100%**

Here Hit Ratio is the number of correctly retrieved images divided by the number of retrieved images. In this particular case, **Hit Ratio = 100%** = 10/10.

12.6.2 Tabasco - Rotation Symmetry

There are two ways to run this example:

- Batch
- Manual

The Batch Run takes only two clicks:

Click "Example/Neural Net/Tabasco - R"
Click "Batch/Run".

The Manual Run requires a few more clicks:

Input:

Training: .\uspto\image026.jpg
Search: .\uspto\

Parameters

Edge Filter: Sobel 1;
Threshold Filter: Dark 128;
NeuralNet Filter Parameter:
Symmetry = Rotation
Blurring = 30
Sensitivity = 80
Internal Cut = 50
ExternalCut = 60000

Operation

- Click the "Key Segment" button and select ".\uspto\image036.jpg";
- Click "Search Dir" button and select ".\uspto\";
- Set parameters as specified above;
- Click "NeuralNet/Train" button to train the filter;
- Click "NeuralNet/1:N Search" button to make a search;
- The result is in a web page;
- If you want to sort, click the "Sort" button.

Results

I026_r10.jpg 78848
I026_r20.jpg 72832
I026_r30.jpg 71104
I026_r40.jpg 70016
I026_r50.jpg 72192
I026_r60.jpg 68992
I026_r70.jpg 69120
I026_r80.jpg 75072
I026_r90.jpg 102976

IMAGE026.JPG 128000000

Summary

# Images	= 126
# To be retrieved	= 10
# Retrieved correctly	= 10
# Missed	= 0
Hit Ratio	**= 100%**

12.6.3 Mr. Potato - Scaling Symmetry

There are two ways to run this example:

- Batch
- Manual

The Batch Run takes only two clicks:

Click "Example/Neural Net/ Mr.Potato -S"
Click "Batch/Run".

The Manual Run requires a few more clicks:

Input:

Training: .\uspto\image043.jpg
Search: .\uspto\

Parameters

Threshold Filter: Light Background 192

NeuralNet Filter Parameters:

Scaling Symmetry
Blurring = 9
Sensitivity = 18
InternalCut = 50 %
ExternalCut = 100000

Results

i042_s110.jpg 120128
i042_s120.jpg 3375000
i042_s130.jpg 114496
i042_s140.jpg 102976
I042_S50.JPG 123712
I042_S60.JPG 408000
I042_S70.JPG 609375
I042_S80.JPG 126784
I042_S90.JPG 122176
IMAGE038.JPG 106880
IMAGE042.JPG 128000000

Summary

# Images	= 126
# To be retrieved	= 10
# Retrieved	= 11
# Retrieved Correctly	= 10
# Missed	= 0
Hit Ratio	= 10/11

12.6.4 Monopoly - Scaling Symmetry

There are two ways to run this example:

- Batch
- Manual

The Batch Run takes only two clicks:

Click "Example/Neural Net/Manopoly -S"
Click "Batch/Run".

The Manual Run requires a few more clicks:

Input:

Training: .\uspto\image046.jpg
Search: .\uspto\

Parameters
Reduction Filter: Int/Max
NeuralNet Filter Parameter:
Blurring = 3
Sensitivity = 23
Image Type = Color 2
Internal Cut = 40
External Cut = 1000

Results
I46_S105.JPG 1220
I46_S110.JPG 1220
I46_S115.JPG 1182
I46_S120.JPG 1104
I46_S80.JPG 2042
I46_S85.JPG 1080
I46_S90.JPG 1041
I46_S95.JPG 1563
IMAGE046.JPG 1280000000

Summary

# Images	= 126
# To be retrieved	= 9
# Retrieved Correctly	= 9
# Missed	= 0
Hit ratio	**= 100%**

12.6.5 Chemical Compound

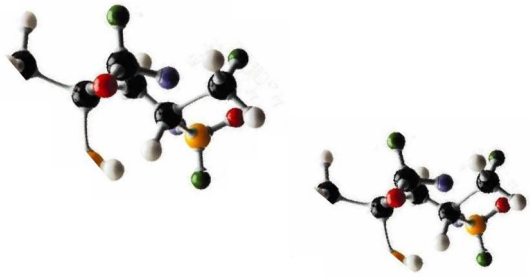
There are two ways to run this example:
- Batch
- Manual

The Batch Run takes only two clicks:

Click “Example/Neural Net/ Compound - RS”
Click “Batch/Run”.

The Manual Run requires a few more clicks:

Input:
Training: .\uspto\i82_s80.jpg
Search: .\uspto\



Parameters

Threshold Filter: Light Background 192
Reduction Filter: Int/Max
NeuralNet Filter Parameters:
Blurring = 25
Sensitivity = 22
Symmetry = Rotation
InternalCut = 25 %
ExternalCut = 80000

Results
56_90_25.JPG 80384
82_110_300.jpg 81280
82_110_320.jpg 82688
82_110_340.jpg 80320
82_80_40.JPG 84032
82_80_50.JPG 80640
82_80_60.JPG 86400
82_90_110.jpg 108288
82_90_120.jpg 103616
82_90_130.jpg 97664
I042_S70.JPG 92992
I82_S110.JPG 88384
I82_S80.JPG 128000000
I82_S90.JPG 112320
image004_t1.jpg 91456
image004_t4.jpg 92352
IMAGE038.JPG 82880
IMAGE082.JPG 89024
IMAGE104.JPG 81344

Summary

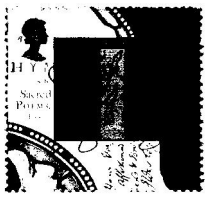
# Images	= 126
# To be retrieved	= 13
# Retrieved Correctly	= 13
# Missed	= 0
Hit Ratio	= 13/19

12.7 Stamp Recognition

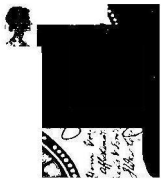
The images used in this section are in the directory ".\stamp\". In this section, we try to identify 2 stamps. Rather than use an existing image to search, we will focus on building a sample image for matching.

12.7.1 Example 1

The first example retrieves images like the following:



We will build a sample image as follows:



There are two ways to run this example:

- Batch
- Manual

The Batch Run takes only two clicks:

Click "Example/Neural Net/Stamp 1"
Click "Batch/Run".

The Manual Run requires a few more clicks:

Input:

Training: .\stamp\class1.jpg
Search: .\stamp\

Parameters

NeuralNet Filter Parameters:

Blurring = 8
Sensitivity = 45
InternalCut = 40 %

Results

CLASS1_4.JPG 46208
CLASS1_1.JPG 41344
class1_10.jpg 15488
CLASS1_2.JPG 45120
CLASS1_3.JPG 16896
CLASS1.JPG 128000000
CLASS1_5.JPG 56064
CLASS1_6.JPG 45568
CLASS1_7.JPG 41984
CLASS1_8.JPG 46336
CLASS1_9.JPG 30400

Summary

# Images	= 104
# To be retrieved	= 11
# Retrieved Correctly	= 11
# Missed	= 0
Hit Ratio	**= 100%**

12.7.2 Example 2

The second example retrieves images like the following:

We will build a sample image as follows:

There are two ways to run this example:

- Batch
- Manual

The Batch Run takes only two clicks:

Click “Example/Neural Net/Stamp 2”
Click “Batch/Run”.

The Manual Run requires a few more clicks:

Input:

Training: .\stamp\class7.jpg
Search: .\stamp\

Parameters

NeuralNet Filter Parameters:

Blurring = 9
Sensitivity = 40
InternalCut = 70 %

Results

CLASS7.JPG 128000000
CLASS7_1.JPG 62144
class7_10.jpg 36480
CLASS7_2.JPG 31360
CLASS7_3.JPG 27328
CLASS7_4.JPG 34560
CLASS7_5.JPG 27136
CLASS7_6.JPG 30720
CLASS7_7.JPG 56448
CLASS7_8.JPG 40704
CLASS7_9.JPG 47744

Summary

# Images	= 104
# To be retrieved	= 11
# Retrieved Correctly	= 11
# Missed	= 0
Hit Ratio	**= 100%**

13. NeuralNet Filter Long-Search

The limit for Short-Search is 1,000 images. There is no technical limit for the Long-Search, except your hardware limits. **Long-Search can search millions of images.**

The difference between Long-Search and Short-Search is the Short-Search does not look into the sub-directories and the Long-Search does look into the sub-directories.

13.1 Long-Search File Structure

The file structure looks like this:

SearchDir
- dumyfile.jpg
- Dir0000
  - Dir0000_000.jpg
  - Dir0000_001.jpg
  - Dir0000_002.jpg
  - …
  - Dir0000_999.jpg
- Dir0001
  - Dir0001_000.jpg
  - Dir0001_001.jpg
  - Dir0001_002.jpg
  - …
  - Dir0001_999.jpg
- Dir0002
  - Dir0002_000.jpg
  - Dir0002_001.jpg
  - Dir0002_002.jpg
  - …
  - Dir0002_999.jpg

In the Long-Search, the search directory can have many sub-directories. In this version, the default number is 3,000 sub-directories. All images to be searched must be in the sub-directories. Each sub-directory can have up to 1,000 images. If your search directory has 3,000 sub-directories and each sub-directory has 1,000 images, then you can search up to 3,000,000 images. **Please leave one file in the search directory.** (In the above structure, it is "dumyfile.jpg") The purpose of this file is for the **ImageFinder** to select the directory when you use the Browse button.

All sub-directories must only be one level deep. If a sub-directory, x, has another sub-directory, y, then the images in sub-directory, y, will not be searched. If you must go more than 1 level deep, you have to use the File-Search.

When operating the **ImageFinder**, the only difference between Short-Search and Long-Search is to click a different command. All the parameters used are the same as the File-Search and Short-Search.

13.2 Trademark Example

Now we will introduce a Trademark Search Example. The directory is ".\Uspto72\", which has about 3,000 images.

Problem Definition:

Job: A trademark attorney wants to make sure the trademark is new.

Problem: The trademark in currently indexed by text. Beyond the text search, the visual search is currently very labor and time intensive.

Solution: Attrasoft **ImageFinder** will exclude 99.9% of the irrelevant images in visual search, which translates into tremendous efficiency and productivity gains.

Figure 13.1 The key image, ./uspto72/ 72158547.gif.

Input

Key:
./uspto72/uspto_72_018/72158547.gif;
Search dir: ./uspto72/.

Parameters

NeuralNet Filter Parameters:
Blurring = 10
Sensitivity = 45
Internal Weight Cut = 30
External Cut = 300000

Operation:

- Click the "Key Segment" button and enter "./uspto72/uspto_72_018/72158547.gif";
- Click the "NeuralNet/Long-Search Directory" command and enter a search directory, E:/uspto72/;
- Set the NeuralNet parameters and set:
  Blurring = 10
  Sensitivity = 45
  Internal Weight Cut = 30
  External Cut = 300000
- Click the "NeuralNet/Train" button to train the filter;
- Click the "NeuralNet/1:N Long-Search" button to make a Long-Search;
- The result is in a web page;
- If you want to sort, click the "Sort" button. The first 200 items in the sorted list will also be printed in the text area.

Results:
Total Images: 4000
#Images Retrieved: 5
Percentage Retrieved: 0.1%

C:/…/uspto72/uspto_72_018/72158547.gif 128000000
C:/…/uspto72/uspto_72_033/72246506.gif 812500
C:/…/uspto72/uspto_72_029/72222807.gif 504000
C:/…/uspto72/uspto_72_025/72197083.gif 440000
C:/…/uspto72/uspto_72_025/72197801.gif 320000

72246506.gif 812500

72197083.gif 440000

Figure 13.2 Two Matched Images.

13.3 Keyword-Search vs Content-Based Search

ImageFinder provides the Content-Based Search. The **ImageFinder** is further divided into:

- Feature Space Recognition Alone (BioFilter, Neural Filter);
- Feature Space and Input Space Recognition (BioFilter, Neural Filter, NeuralNet Filter File-Search);

- Input Space Recognition Alone (NeuralNet Filter, Short-Search and Long-Search).

If you need both Keyword-Search and Content-Based Search, you will need to combine the Attrasoft **ImageFinder** with a database-management-system (DBMS) software, like Microsoft Access. A DBMS application saves the meta-data, which describes the images with key words.

An imagebase or image database is an image file cabinet. An imagebase is a set of tables. A table is a set of records. A record consists of a set of fields. A field contains text or links to images. Obviously, you can insert, update, delete, and retrieve images. The most important feature of an imagebase is to retrieve images efficiently.

You can use the DBMS software for the following functions:

- Insert
- Update
- Delete
- Keyword Retrieval

You can use the **ImageFinder** for:

- Content-Based Image Retrieval.

14. NeuralNet Filter Parameter Advisor

NeuralNet Filter has more parameters than the BioFilter and Neural Filter. In this chapter, we will study the Parameter Advisor, which will help you to select values for the parameters. We will use a Car-License-Plate Identification Problem as an example.

14.1 Parameter Advisor

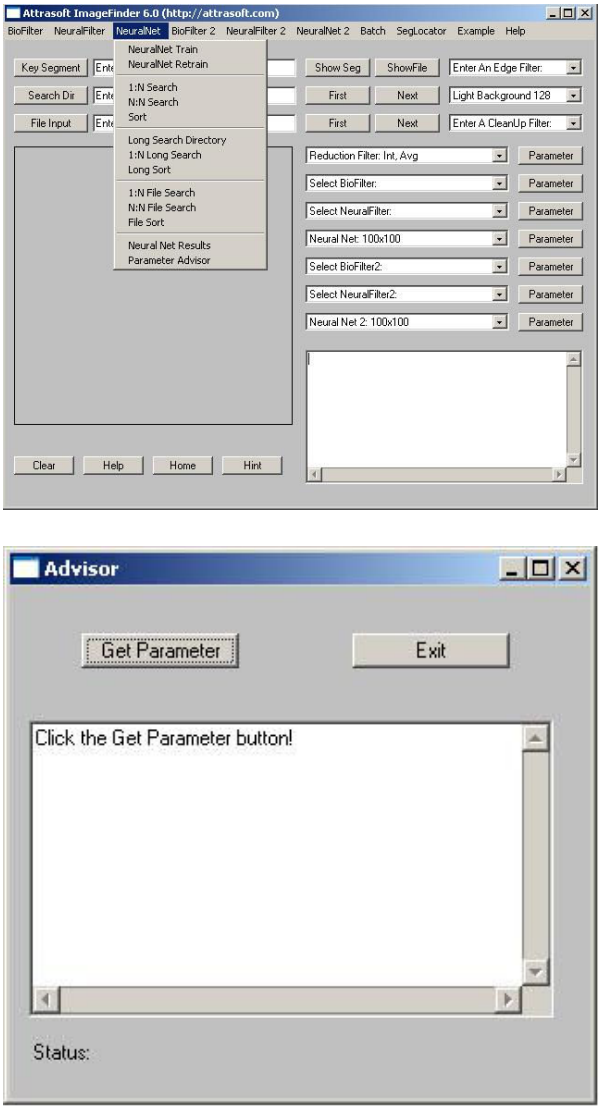


Figure 14.1 Parameter Advisor.

The data used in this section is in the "license1\" directory. The Parameter Advisor helps you to choose a set of important parameters for the NeuralNet Filter. The Parameter Advisor will make recommendations for:

- Segment Cut
- Blurring
- Sensitivity
- Internal Weight Cut

You will still need to set the other parameters, but they are relatively easy to set. To use the Parameter Advisor, you must:

- Put all similar images in a directory;
- Set the parameters very high to make sure that the **ImageFinder** will retrieve the matching images.

We will start with the following parameters:

Segment Cut = 0
Blurring = 50
Internal Cut = 95
Sensitivity = 95

The reason is: the Advisor will only look at the parameter-range up to a certain point. For example, if you set the Blurring = 20, then the Advisor will only look at the range of Blurring from 0 to 20; if you set the Blurring = 50, then the Advisor will only look at the range of Blurring from 0 to 50;

The data used in this example is given in the following figure:

Figure 14.2 All images in the first class.

The following example will show you how to use the Parameter Advisor. You can duplicate the set up of this example by several clicks:

- Click the "Example/NeuralNet/Parameter Advisor" to open the parameters for this example;
- Click "Batch/Load" to load the parameters;
- Click "NeuralNet/Parameter Advisor" to open the Parameter Advisor;
- In the Parameter Advisor, click "Get Parameter" button; this will initiate the NeuralNet Filter to train and search;
- Close the **ImageFinder** output. The recommendation will be in the text area of the Parameter Advisor;
- Close the Parameter Advisor.

In the following, we will show you the details of running this example.

The first two clicks load the parameters; alternatively, you can do this manually. The data used is in the ".\license1\" directory. The steps to do it manually are:

1. Data directory is "license1\". See these images with Windows Explorer.

   761SBY~7.JPG
   761SBY~2.JPG
   761SBY~3.JPG
   761SBY~4.JPG
   761SBY~6.JPG
   761SBY~1.JPG
   761SBY~1.JPG

2. Set image, license1\761SBY~1.JPG, as a sample image (Click Key Segment button).

3. Set directory, license1\, as the search directory (Click Search Dir button).

4. Threshold: Average Filter.
Threshold Filter Parameter:
   Red: Ignore
   Green: Ignore
   Blue: 0 – 100; Light Background.

5. Reduction Filter: Real Max
Reduction Filter Parameter
   Border Cut = 2

6. NeuralNet Filter Parameter:
   Symmetry: Rotation
   Segment Size: "S Segmant" (Small)
   R(rotation) Type: Type 1
   Sample Segment: 50 50 200 200

Set the following parameter to as large as possible to make sure that the **ImageFinder** will retrieve the matching images:

   Segment Cut = 0
   Blurring = 90
   Internal Cut = 90
   Sensitivity = 90

You can do all of the above by:

- Click the "Example/NeuralNet/Parameter Advisor";
- Click the "Batch/Load".

To get a recommendation, now:

- Click the "NeuralNet/Parameter Advisor" to open the Parameter Advisor;
- In the Parameter Advisor, click the "Get Parameter" button; this will initiate the NeuralNet Filter to train and search;

you will get:

761SBY~4.JPG 428892
761SBY~2.JPG 59250
761SBY~3.JPG 535000
761SBY~1.JPG 128000000
761SBY~6.JPG 518000
761SBY~7.JPG 14000000

The recommendation is:

Segment Cut: Very High
Blurring: 23 28
Internal Weight Cut: 16 to 21
Sensitivity: 9 to 14

Remember the recommendation is based on the average of the minimum requirement. The recommendation is generally on the lower side. In this example, we are trying to use one set of the parameters which will fit all of the license plate images, so we will choose the parameter a bit higher than the recommendation.

After a few rounds of testing, this is the set of parameters:

Reduction: Real Max
Blurring: 20
Sensitivity: 25
Internal Cut: 22

14.2 License Plate Recognition

The data used in this section is in the ".\license" directory. Use the Windows Explorer to see the images. This example has 50 images from 9 different car plates. The following are some of the examples:

Figure 14.3 The first image in each class.

The operation has two steps:

1. Click "Example/NeuralNet/License Plates" to get the batch code;
2. Click "Batch/Run" Button.

47 out of 50 are identified. There are three errors, 3345~1.jpg, 3349~5.jpg and 3349~9.jpg, which gives

Identification Rate = 94% = 3/50.
Error Rate = 6%.

15. Biometrics

Biometrics refers to Facial, Fingerprint, Voice, Palm Print, Iris Recognition, … In this chapter, we will study Face Recognition and Fingerprint Recognition. The example of Face Recognition is given in chapter 3. We will use Fingerprints as illustrations to show you how to build an image recognition solution using the **ImageFinder.**

15.1 Attrasoft Facial Recognition Classifications

When dealing with "Facial Recognition", words mean different things to different people. We will first define what Facial Recognition is in terms of 5 categories.

Attrasoft divides Facial Recognition into 5 categories:

Class 1: Photo ID Recognition

Class 1 Face Recognition is Photo ID (Passport, Driver's license, ...) Recognition, which is semi-automatic:

- Before a passport is issued, a picture of the passport is taken and stored in a database, which has facial images and/or fingerprints;
- The computer will compare (match or deny) the stored original passport images and the newly captured passport image via the passport number as the person goes through security;
- Airport staff is responsible for comparing the passport photo with the passport holder.

Class 1 Face Recognition appears to be somewhat "low tech", until you consider the following facts:

- **Class 1 Face Verification Rate is 100%.** All other face recognition is not even close to 100% verification rate.
- No face recognition system, which addresses face recognition beyond Class 2, has ever been successful.

Class 2: Highly Intrusive Live Image

In Class 2 Facial Recognition, the captured live image is taken by placing their chin on a chin plate while their picture is taken; their picture is then compared to the stored image taken in a similar manner. This method is considered intrusive and highly restrictive. Fingerprint, palm print, and iris recognition fall into this category.

Class 3: Low Intrusive Live Image

Class 3 Face Recognition has the following features:

- Non-intrusive, no physical contact is made;
- The person is asked to stand in a box area and is asked to look forward.

Class 4: Very Low Intrusive Live Image

Class 4 Face Recognition has the following features:

- Non-intrusive, no physical contact is made;
- The person is asked to walk through a narrow path.

Class 5: No Restriction

Class 5 Facial Recognition has no restrictions:

- The person can walk through the security area and their identity can be verified or admission denied via video camera.

Attrasoft **FaceFinder** will only address Class 1 and Class 2 Face Recognitions.

Generally speaking, the more restrictive the images are when taken, the higher the

identification rates will be. There are various ways to improve facial recognition rates. For example, verify a passenger with two live pictures is better than one picture.

The Face Recognition example has been addressed in the **FaceFinder** chapter.

15.2 How to Build an Image Recognition Solution with the ImageFinder

The procedure for building a solution with the **ImageFinder** is fairly predictable. At a minimum, you will need to go through the following steps:

- Data
- Image Processing
- BioFilter
- Neural Filter
- NeuralNet Filter
- Improvement

Data

Data is further divided into a development set and a test set (blind set). The blind set is usually not available for the developers. A typical development set has 1000 images and a typical test set has 1000 images. Each image in a set compares with the whole set, so 1000 images will generate 1,000,000 comparisons.

Often a special type of images require a special image-processing filter, which the **ImageFinder** may or may not have it. If the **ImageFinder** does not have it, the pre-processed image can be presented to the **ImageFinder**. This will be the case in our fingerprint example.

BioFilter

The BioFilter provides an option called Unsupervised Learning, i.e. you can start to identify the images basically without doing anything. This should quickly give you an idea how hard the problem is. If the Unsupervised Learning basically identifies most of images correctly, then you have an easy problem; on the other hand, if the Unsupervised Learning does not work at all, you have a hard problem.

To use the Unsupervised Learning:

- Click "Search Dir button" and select a directory containing the images;
- Click "BioFilter/Scan Images – Directory Input" to convert images into templates.
- Click "BioFilter/BioFilter N:N Match (Untrained)" to see the results.

The next step is to train the BioFilter. You have to prepare the match.txt file, which lists the matching pairs. If you have done the above steps, you can continue to do the following:

- Click "BioFilter/BioFiloter Train (match.txt required)";
- Click "BioFilter/BioFilter N:N Match (Trained)" to see the results.

Neural Filter

Neural Filter will significantly improve the BioFilter results. The advantage of this filter is that it is very fast; the speed is 100,000 comparisons per second or more.

NeuralNet Filter

NeuralNet Filter will significantly improve the Neural Filter results. The NeuralNet Filter is much slower than the Neural Filter.

Improvement

Working on the **ImageFinder** parameters can make improvements on the Identification Rates. The idea is that each class has its own parameters rather than all classes share a set of common parameters. There are three vectors that this step deals with:

- Pixel Array;

- Template;
- Parameter Vectors.

The parameter vectors are those numbers in the batch file, which set the parameters of the **ImageFinder**. Usually the parameter vector is fixed for a given problem. It will increase the recognition rates significantly if the parameters can vary for each class of images. In practice, a batch file is associated with each class of images rather than all classes.

Unlike the BioFilter and Neural Filter, the selection of a parameter vector for the Neural Net from a set of matching images is not done automatically in this version. As we have seen in the last chapter, the Parameter Advisor can make a recommendation for the parameters, but a person has to check the validity of the recommendations.

The rest of this chapter will be organized according to the above procedure, with a section addressing each step:

- Data
- Image Processing
- BioFilter
- Neural Filter
- NeuralNet Filter
- Improvement

15.3 Fingerprint Data

Fingerprint Recognition is equivalent to Class 2 Recognition discussed in the last section, i.e. they are intrusive.

There are 56 images from 8 different persons. All fingerprint images are preprocessed with the edge filters already. Each image must be recognized 1 time and must be rejected 7 times when being compared with the other 7 different fingerprints. So there are 56 positive recognitions and 56*7 = 392 negative recognitions, giving a total of 448 tests.

The fingerprint images used in this example are very, very noisy and this makes recognition very hard.

Figure 15.1 First image of 4 persons.

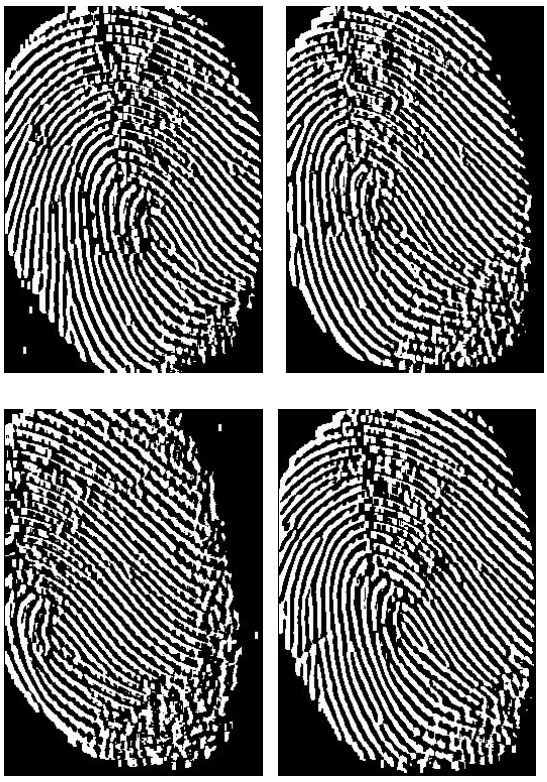


Figure 15.2 All images for person 5.

AS discussed in the last section, these images are preprocessed.

15.4 Image Processing

The images are already processed. We will set:

Threshold Filter: Dark Background 96;
Reduction Filter Parameter:

- Border Cut: 5
- Segment Cut: 8

This completes image processing.

15.5 BioFilter

We now converting an image in the input space into a vector in the feature space:

- Click "Search Dir" button and select any image in the ".\Finger\" directory to select this directory;
- Click "BioFilter/Scan Images – Directory Input" to convert.

We now make a preliminary investigation of the data set via unsupervised learning:

- Click the "Parameter" button next to the BioFilter and set the "BioFilter Scale" to 50;
- Click "BioFilter/BioFilter N:N Match (Untrained)" to see the results.

The result is:

101_1B.GIF
101_1B.GIF
101_2B.GIF
101_3B.GIF
101_4B.GIF
101_8B.GIF
103_3B.GIF
103_4B.GIF
103_6B.GIF
103_7B.GIF
103_8B.GIF
104_5B.GIF
107_1.GIF
107_3.GIF
107_4.GIF
107_5.GIF
107_7.GIF

101_2B.GIF
101_1B.GIF
101_2B.GIF
101_3B.GIF
101_4B.GIF
101_8B.GIF
102_1B.GIF
103_3B.GIF
103_4B.GIF
103_6B.GIF
103_7B.GIF
103_8B.GIF
104_1B.GIF
104_2B.GIF
104_5B.GIF
104_6B.GIF
104_8B.GIF
107_1.GIF
107_3.GIF
107_4.GIF
107_5.GIF
107_7.GIF

101_3B.GIF
101_1B.GIF
101_2B.GIF
101_3B.GIF
101_4B.GIF
101_8B.GIF
104_1B.GIF
104_5B.GIF
107_1.GIF
107_4.GIF

101_4B.GIF
101_1B.GIF
101_3B.GIF
101_4B.GIF

101_8B.GIF

…

The first image in a block is the input and the rest in the block are output. Some images are classified, but there are also images that cannot be classified. We will move to the next step, train the BioFilter.

Training requires a training file, match.txt. This file is already prepared; go to the **ImageFinder** home directory and open "finger_match.txt" to see the training file. To train the BioFilter:

- Save ".\Finger_match.txt" to ".\match.txt" (overwrite the existing file);
- Click the "Parameter" button next to the BioFilter and set the "BioFilter Scale" to 10;
- Click "BioFilter\BioFiloter Train (match.txt required)";
- Click "BioFilter/BioFilter N:N Match (Trained)" to see the results.

There is no improvement, so we will need to go to the Next Step, Neural Filter.

15.6 Neural Filter

The Neural Filter requires the same training file, match.txt, as the BioFilter. Since we have done this step for BioFilter, we are ready to match now:

- Click "NeuralFilter\NeuralFiloter Train (match.txt required)";
- Click "NeuralFilter/NeuralFilter N:N Match" to see the results.

The result is:

101_1B.GIF
101_1B.GIF
101_2B.GIF
101_3B.GIF
101_7B.GIF
101_8B.GIF
103_6B.GIF

101_2B.GIF
101_1B.GIF
101_2B.GIF
101_3B.GIF
103_6B.GIF
104_4B.GIF
107_1.GIF
107_2.GIF
107_3.GIF
107_4.GIF
107_5.GIF

101_3B.GIF
101_1B.GIF
101_2B.GIF
101_3B.GIF
101_4B.GIF
101_7B.GIF
101_8B.GIF

101_4B.GIF
101_3B.GIF
101_4B.GIF
101_5B.GIF
101_7B.GIF
101_8B.GIF

…

The first image in a block is the input and the rest in the block are output. More images are classified compared to the BioFilter, but there are also images that cannot be classified. We will move to the next step, train the Neural Net.

15.7 NeuralNet Filter

We will continue the example from the last section. We will set the NeuralNet Parameters:

Segment: 90 90 120 120

Sensitivity: 75
Blurring: 15
Cutoff: 23900
Segment Size: Small Segment
Internal Weight Cut: 15

The output of the last section is b1.txt. This will be the input of this section:

- Click the "File Input" button and select b1.txt;
- Click "NeuralNet/1:N File-Search".

A quick way to get this done is:

- Click "Example/NeuralNet/Fingerprint";
- Click "Batch/Run".

The result is:

```
Training: 101_1B.GIF
101_1B.GIF   32000000
101_2B.GIF   25484
101_3B.GIF   30452
--------------
Training: 101_2B.GIF
101_1B.GIF   27140
101_2B.GIF   32000000
101_3B.GIF   26744
--------------
Training: 101_3B.GIF
101_1B.GIF   31064
101_2B.GIF   29300
101_3B.GIF   32000000
101_4B.GIF   28652
101_7B.GIF   28508
101_8B.GIF   28760
…
```

There are 56 images from 8 different persons. Each image must be recognized 1 time and must be rejected 7 times when being compared with the other 7 different fingerprints. So there are 56 positive recognitions and 56*7 = 392 negative recognitions, giving a total of 448 tests.

Errors:

102_1B: 1 false acceptance;
103_7B: 3 false acceptances when 103 pretend to be 102, 104, and 106;
105_2B: 1 false acceptance when 105 pretend to be 106;
105_7B: 1 false acceptance when 105 pretend to be 102;
107_3B: 1 false acceptance;
107_7B: 1 false acceptance;
109_2B: 2 false acceptances when 109 pretend to be 102 and 106;
109_2B: 1 false acceptance;

102_8B: 1 false rejection;
107_2B: 1 false rejection;

Together, there are 2 errors in positive recognition and 12 errors in negative recognition.

Rates:

Total Number of Tests = 448

Positive Recognition = 96.4 % = 54/56.
Negative Recognition = 96.9 % = 380/392 (percentage of time Software recognizes you as using the wrong ID).
False Rejection = 3.7 % = 2/56. (percentage of time software will reject the right person).
False Acceptance = 3.06 % = 12/392. (percentage of time software will identify the wrong person as being the right person).

This result is not very good, so we move to the next step, improvement.

15.8 Improvement

Working on the **ImageFinder** parameters can make improvements on the Identification Rates. The idea is each class has its own parameters rather than all classes share a set of common parameters. It will increase the recognition rates

significantly if the parameters can vary for each class of images. In practice, a batch file is associated with each class of images rather than all classes.

Unlike the BioFilter and Neural Filter, the selection of a parameter vector for the Neural Net from a set of matching images is not done automatically in this version.

In the following example, we will choose 4 parameters, which can change from fingerprint to fingerprint. There are four parameters: Segment Cut, Border Cut from the Reduction Filter, Internal Cut, and External Cut (Threshold) from the NeuralNet Filter. After a few rounds of experiments, the parameters are:

Images	Parameters
101	(8,5,14,24500)
102	(8,2,14,25000)
103,4	(12,8,13,25000)
105,6,9	(12,5,13,25000)
107	(10,6,13,24500)

Basically, each class has its own parameters. To run this example:

- Click "Example/NeuralNet/Fingerprint - 1";
- Click "Batch/Run" to get results for 101;
- Click "Example/NeuralNet/Fingerprint - 2";
- Click "Batch/Run" to get results for 102;
- Click "Example/NeuralNet/Fingerprint - 34";
- Click "Batch/Run" to get results for 103, 104;
- Click "Example/NeuralNet/Fingerprint - 7";
- Click "Batch/Run" to get results for 107;
- Click "Example/NeuralNet/Fingerprint - 569";
- Click "Batch/Run" to get results for 105, 106, 109.

There is 1 false rejection for the 448 tests. The results for 101 are:

```
Training: 101_1B.GIF
101_1B.GIF    32000000
101_2B.GIF    25484
101_3B.GIF    30452
------------------------------
Training: 101_2B.GIF
101_1B.GIF    27140
101_2B.GIF    32000000
101_3B.GIF    26744
------------------------------
Training: 101_3B.GIF
101_1B.GIF    31064
101_2B.GIF    29300
101_3B.GIF    32000000
101_4B.GIF    28652
101_7B.GIF    28508
101_8B.GIF    28760
------------------------------
Training: 101_4B.GIF
101_3B.GIF    28544
101_4B.GIF    32000000
101_5B.GIF    27860
101_8B.GIF    25016
------------------------------
Training: 101_5B.GIF
101_4B.GIF    26456
101_5B.GIF    32000000
101_6B.GIF    26492
101_7B.GIF    27752
------------------------------
Training: 101_6B.GIF
101_5B.GIF    28076
101_6B.GIF    32000000
101_7B.GIF    26996
------------------------------
Training: 101_7B.GIF
101_1B.GIF    26996
101_3B.GIF    27500
101_4B.GIF    26492
101_5B.GIF    27536
101_6B.GIF    25880
101_7B.GIF    32000000
```

101_8B.GIF 27608

Training: 101_8B.GIF
101_1B.GIF 25484
101_3B.GIF 31568
101_4B.GIF 25700
101_7B.GIF 28544
101_8B.GIF 32000000

Rates:

Total Number of Tests = 448

Positive Recognition = 98.2 %
= 55/56.

Negative Recognition = 100 %
= 392/392 (percentage of time Software recognizes you as using the wrong ID).

False Rejection = 1.8 %
= 1/56 (percentage of time software will reject the right person).

False Acceptance = 0 %
= 0/392 (percentage of time software will identify the wrong person as being the right person).

16. SegLocator

In this chapter, we will introduce **Segment Locator** or **SegLocator**. The **SegLocator** is a specialized NeuralNet filter, which will locate a segment and print their coordinates.

The **SegLocator** works exactly the same as the **ImageFinder**; however the **SegLocator** only supports 1:N matching in the directory input mode, i.e. the **SegLocator** does not support file input and Long-Search modes. The available commands are:

- SegLocator/Training
- SegLocator/1:N Match
- SegLocator/Batch Run

The "SegLocator/Batch Run" only supports batch code 1027, which is a consecutive execution of the following two commands:

- SegLocator/Training
- SegLocator/1:N Match

The **SegLocator output** is the **ImageFinder** output plus the coordinates of the segment located in the images. Note: Each image has 9 logos.

Note: Each image contains 9 logos.

Figure 16.1 The data consists of 4 images.

16.1 Coordinate System

The data for this chapter consists of the four images below:

There are 9 trademarks randomly placed in the four images. We want the **SegLocator** not only to recognize the image segment, but also locate the position.

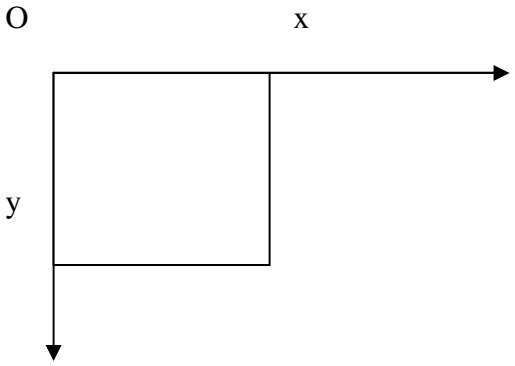


Figure 16.2 Segment Specification.

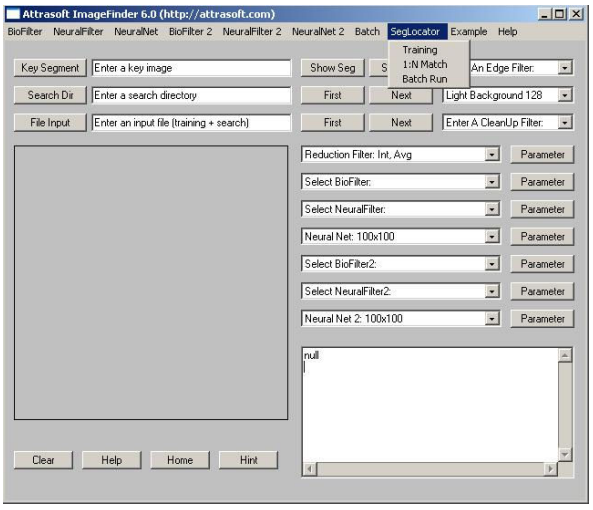


Figure 16.3 SegLocator.

The image segment is specified by 5 numbers:

(x, y, w, h, r)

where:

- x is the x-coordinate of the upper left corner;
- y is the y-coordinate of the upper left corner;
- w is the x-length of the segment (width);
- h is the y-length of the segment (height);
  and,
- r is the rotation angle.

The default Representation is 100x100. The x coordinate runs from 0 to 100 and the y coordinate runs from 0 to 100. (This depends on the NeuralNet filter selected. If you choose a 50x50 NeuralNet filter instead of 100x100, then the x coordinate runs from 0 to 50 and the y coordinate runs from 0 to 50.) The origin of the coordinate system is at the upper left corner. The x-axis runs from left to right and the y-axis runs from top to bottom.

To find a segment in an image, you can use:

- SegLocator/Training followed by SegLocator/1:N Match; or
- SegLocator/Batch Run.

Note that the image area in the **ImageFinder** is 300x300. You can find the location of a segment in an image by viewing it in the **ImageFinder**, but remember in the **ImageFinder**, the image area is 300x300, while the output specification is 100x100.

16.2 Six Examples

1. The United Way.

You can duplicate this example by two clicks:

- Click "Example/SegLocator/United Way";
- Click "SegLocator/Batch Run".

If you want to set the parameter manually,

Input
Key Segment = image002

NeuralNet Parameter
Training Segment 98 16 98 81

The segment picked is 98 16 98 81. This is in 300x300 scale because the **ImageFinder** display area is 300 x 300 pixels. The output is 100x100 scale; and (98 16 98 81) in 300x300 scale will convert to (33, 5, 33, 27) in 100x100 scale. The output is:

image002.jpg 86976
34 7 26 20 0
image004.jpg 86528
2 4 26 20 0
image006.jpg 84608
63 33 26 20 0
image008.jpg 82368
8 10 26 20 0

In all 4 cases, the "United Way" segments are correctly identified.

The output for the first image is (34 7 26 20) and the original segment is (33, 5, 33, 27). The difference is from the fact that the **SegLocator** locates the image segment without a blank area, while the original training segment, (33, 5, 33, 27), contains a blank area.

2. Monopoly.

You can duplicate this example by two clicks:

- Click "Example/SegLocator/Monopoly";
- Click "SegLocator/Batch Run".

If you want to set the parameter manually,

Input
Key Segment = image002

NeuralNet Parameter
Training Segment 85 117 110 68

The segment picked, in 300x300 scale, is 85 117 110 68, which translates to 100x100 scale as: (28, 39, 37, 23). The output is:

image002.jpg 78784
31 42 29 8 0
image004.jpg 78336
35 74 29 8 0
image006.jpg 78080
34 77 29 8 0
image008.jpg 76480
7 77 29 8 0

In all 4 cases, the "Monopoly" segments are correctly identified.

The output for the first image is (31 42 29 8) and the original segment is (28, 39, 37, 23). The difference is from the fact that the **SegLocator** locates the image segment without a blank area, while the original training segment contains a blank area.

3. Mr. Potato.

You can duplicate this example by two clicks:

- Click "Example/SegLocator/Mr. Potato";
- Click "SegLocator/Batch Run".

If you want to set the parameter manually,

Input
Key Segment = image002

NeuralNet Parameter
Training Segment is 206 17 74 88

The segment picked, in 300x300 scale, is 206 17 74 88. The output is:

image002.jpg 79552
73 9 12 20 0
image004.jpg 78080
15 68 12 20 0
image006.jpg 78336
40 41 12 20 0
image008.jpg 75584

48 11 12 20 0

In all 4 cases, the "Mr. Potato" segments are correctly identified.

4. AAA.

You can duplicate this example by two clicks:

- Click "Example/SegLocator/Mr. Potato";
- Click "SegLocator/Batch Run".

If you want to set the parameter manually,

Input
Key Segment = image002

NeuralNet Parameter
Training Segment is 109 194 74 75

The output is:

image002.jpg 79872
37 66 19 14 0
image004.jpg 79296
68 41 19 14 0
image006.jpg 77376
7 12 19 14 0
image008.jpg 76288
39 43 19 14 0

In all 4 cases, the "AAA" segments are correctly identified.

5. Ford.

You can duplicate this example by two clicks:

- Click "Example/SegLocator/ Ford";
- Click "SegLocator/Batch Run".

If you want to set the parameter manually,

Input
Key Segment = image002

NeuralNet Parameter
Training Segment is 195 202 92 60

The output is:

image002.jpg 78976
62 70 22 8 0
image004.jpg 78912
66 14 22 8 0
image006.jpg 77888
40 16 22 8 0
image008.jpg 75904
7 47 22 8 0

In all 4 cases, the "Ford" segments are correctly identified.

6. Soup.

You can duplicate this example by two clicks:

- Click "Example/SegLocator/ Soup";
- Click "SegLocator/Batch Run".

If you want to set the parameter manually,

Input
Key Segment = image002
Image processing
Threshold File = Customized filter:
Red 30 - 127 Light Background
Green: Ignore
Blue: Ignore

NeuralNet Parameter
Training Segment is 18 14 53 69
Blurring = 2
Segment Size = S(Small) Segment

The output is:

image002.jpg 2400000
10 7 8 13 0
image004.jpg 48000
50 46 8 13 0
image006.jpg 60000
18 69 8 13 0
image008.jpg 48000
73 20 8 13 0

In all 4 cases, the "Soup" segments are correctly identified.

16.3 Segment Output

The output of the **SegLocator** is within an area of 100x100. Interpretation of the **SegLocator**'s output depends on:

- Image Size;
- NeuralNet Filter;
- Reduction Filter.

Assume:

- An image is 256x384;
- Neural Net is 100x100; and
- The Reduction Filter is Integer.

Under the Integer Reduction Filters, the new width is 256/4 = 64, and the new height is 384/4 = 96. Assume the **SegLocator** identifies a segment in (x, y, w, h) = (10, 10, 20, 20), then in terms of pixel value, it means 4 * (10, 10, 20, 20) = (40, 40, 80, 80).

Assume:

- An image is 256x384;
- Neural Net is 100x100; and
- The Reduction Filter is Real.

Under the Real Reduction Filters, the new width is 256/3.8 = 67, and the new height is 384/3.8 = 100. Assume the **SegLocator** identifies a segment in (x, y, w, h) = (10, 10, 20, 20), then in terms of pixel value, it means 3.8 * (10, 10, 20, 20) = (38, 38, 76, 76).

17. Customized Point-Locator

The **Point-Locator** is a special version of the **Segment-Locator**; it locates points by locating the segment first and specifies points based on the position of the located segment.

This chapter introduces a **Customized Point-Locator**, in which the training segment is already designed and encoded into the software. You do not need to train the software. You will run the software with the "Batch/Run" command.

17.1 Locating Eye and Nose

You can run this example with three clicks:

- Click "Example/SegLocator/Point Locator: Feret 20";
- Click "Batch/Run";
- Click "Example/SegLocator/Point Locator: Results".

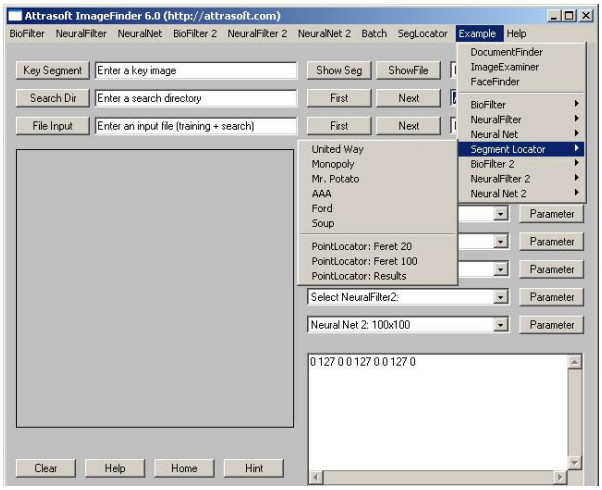


Figure 17.1 Customized Point-Locator.

In this example, we will locate the eyes and nose in a face. This customized module will work only for the following images:

1. Ferret Database Face Image, 256x384 pixels;
2. Only front view images can be used.

The first 20 images in the Feret database will be used in this example.

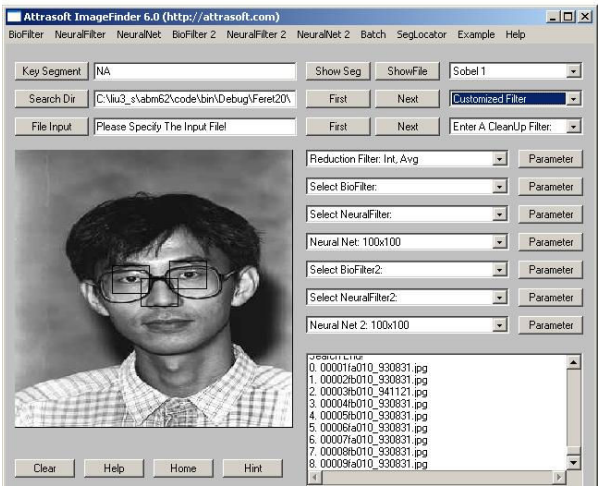


Figure 17.2 Visual results of **Customized Point-Locator.**

We will now explain this example. The first click, "Example/SegLocator/Point Locator: Feret 20", will load the batch file. The second click, "Batch/Run", will execute this batch file. Note the execution code 1101. Each customized version has its own batch code. The **Point Locator** will:

1. Locate the Face;
2. Locate the left eye, the right eye, and nose ROI (Region Of Interest);

After the **Point-Locator** locates the left eye segment, the right eye segment, and the nose segment, it will compute their coordinates based on the located segments. The training images are hard coded into the **Customized Point-Locator**, therefore, there is no need for training. The results will look like this:

0. 00007fa010_930831.jpg
   Left eye: 98.0 165.0
   Right: 159.0 161.0
   Nose: 127 199
1. 00002fb010_930831.jpg

Left eye: 107.0 166.0
Right: 158.0 157.0
Nose: 126 184
2. 00003fb010_941121.jpg
Left eye: 89.0 192.0
Right: 153.0 196.0
Nose: 119 229
…

The third click, "Example/SegLocator/Point Locator: Results", will show the results visually; the **Point-Locator** will draw a 40x30 square on left eye, the right eye, and the nose. The **Point- Locator** will present an animation, which runs through all images. This will last 80 seconds, 4 seconds per image.

17.2 Feret 100

You can run this example with three clicks:

- Click "Example/SegLocator/Point Locator: Feret 100";
- Click "Batch/Run";
- Click "Example/SegLocator/Point Locator: Results".

This example will use images in the ".\Feret100\" folder. There are 100 images in this folder. Other than the numbers, this example is similar to the previous example.

17. 3 Run Your Data

You can use other Feret images by using a different folder. Here is the operation:

- Click "Example/SegLocator/Point Locator: Feret 20" to open the batch file;
- Click "Batch/Load" to load the parameters;
- Click "Search Dir" button to select the folder containing your images;
- Click "Batch/Save" to create the batch file; make sure the execution code (second line) is 1101;
- Click "Batch/Run" to run your data;
- Click "Example/SegLocator/Point Locator: Results" to see the results visually.

17.4 Analysis

Now we will compare the output of the **Point-Locator** for the "Feret 20" example with human work. The following are two tables for the left eye and the right eye. The images are 256x384 pixels in size. The units for all measurements are pixels. Each table has the following columns:

Image Number
Human Result: x1
Human Result: y1
Point-Locator: x2
Point-Locator: y2
Gap = |x2-x1| + |y2-y1|

Left Eye

	x1	y1	x2	y2	
1	98	165	101.0	164.0	4.0
2	106	165	107.0	166.0	2.0
3	88	191	89.0	192.0	2.0
4	92	171	91.0	172.0	2.0
5	102	150	103.0	150.0	1.0
6	112	175	113.0	177.0	3.0
7	95	163	98.0	165.0	5.0
8	91	170	93.0	171.0	3.0
9	106	179	107.0	180.0	2.0
10	124	185	126.0	188.0	5.0
11	112	158	112.0	157.0	1.0
12*	112	159	112.0	171.0	12.0
13	112	180	110.0	181.0	3.0
14	121	170	121.0	165.0	5.0
15	107	184	108.0	185.0	2.0
16	104	164	105.0	166.0	3.0
17	123	183	123.0	184.0	1.0
18	96	176	94.0	177.0	3.0
19	104	158	106.0	160.0	4.0
20	119	150	120.0	151.0	2.0
sum					65.0

Here * means error. There are 40 measurements total (20 pairs of (x,y)). The total gap is 65, and the average measurement Gap = 65/40= 1.6. There is one error; Error Rate = 2.5% = 1/40.

Right Eye

	x1	y1	x2	y2	
1*	161	163	149.0	165.0	14.0
2	158	160	158.0	157.0	3.0
3	152	194	153.0	196.0	3.0
4	149	167	147.0	167.0	2.0
5	158	145	159.0	145.0	1.0
6	170	173	168.0	173.0	2.0
7	160	159	159.0	161.0	3.0
8	148	161	148.0	162.0	1.0
9	160	173	160.0	174.0	1.0
10	176	186	178.0	188.0	5.0
11	168	160	164.0	161.0	5.0
12	169	171	171.0	173.0	4.0
13	168	181	171.0	183.0	5.0
14	181	174	178.0	174.0	3.0
15	160	180	158.0	180.0	2.0
16	150	163	151.0	164.0	2.0
17	177	187	179.0	188.0	3.0
18	142	176	143.0	176.0	1.0
19	165	162	167.0	163.0	3.0
20	161	147	162.0	149.0	3.0
sum					65.0

Average measurement Gap = 65/40 = 1.6
Error Rate = 2.5% = 1/40.

Other than a few percentages of errors, the human reading and the **Point-Locator** readings are very close.

18. BioFilter II

Until now, we have been using one-layer Neural Network Architecture:

Image Preprocessing
    Edge Filters;
    Threshold Filters; and
    Clean Up Filters.
Normalization
    Reduction Filter.
Feature Recognition
    BioFilter;
    NeuralFilter.
Pixel Level Recognition
    ABM Filter.

We call this **Basic Architecture**. When you go beyond the basic architecture, the options are limitless.

We will focus our discussion on a specific application: to compare two basically identical images and identify the minor differences. For example, in two satellite images, there is a car in one image, but not in the other. Potential applications are:

    Satellite Image Recognition
    Quality Control
    Product Label Recognition
    …

Our approach is to divide the image into small sections and use our basic architecture for each smaller section of the image. This will generate the following additional layers:

    BioFilter II;
    NeuralFilter II;
    ABM Filter II.

In this multi-layered Neural Network Architecture, Layer 1 (basic architecture) serves as a filter for later layers (i.e., to decide quickly whether an image can pass or not). It simply decides whether two images are similar with potentially minor differences, or whether these are two completely different images.

Throughout this chapter, we will focus on a Label Verification problem.

18.1 Input

An input file or an input directory enters data into the **BioFilter II**. An input file is a text file, which looks like this:

C:\abc1\efg1\image0001.jpg
C:\abc2\efg2\image0002jpg
…

This file lists one image per line. Please do not add a blank space to the end of the line. The first line in the file is the newly captured image, which will be matched against the previously stored images specified by the rest of the line. The number of the previously stored images is arbitrary; for example, in this input file:

C:\newlycaptured\L12063.jpg
C:\masterdata1\L12063.jpg
C:\masterdata2\L12063.jpg
C:\masterdata3\L12063.jpg

The newly captured image, C:\newlycaptured\L12063.jpg, will be matched against three previously stored images.

Multiple matches cannot be entered in a single input file; each match must have its own file. In the case of directory input, the first image will be considered as the "newly captured image" and the rest of the images will be considered "previously stored images". Because it is hard to control the image scanning order, the file input is preferred over the directory input.

18.2 Data

The label verification problem is to make sure the new product images match the existing images. The requirements are to compare a newly captured image against previously stored images and determine:

- **Match or No Match?**
- **If not, where is the error?**

This section presents two pairs of images used in this chapter. The first pair of images match, and the second pair does not match.

18.2.1 Good Match

Good Match verifies that the newly captured image matches the existing images. Figure 18.1 shows a Good Match.

Figure 18.1 Good Match.

18.2.2 Bad Match

Bad Match rejects the newly captured image because it does not match the existing images. The following figure is an example:

Figure 18.2 Bad Match.

The errors are:

Error Number	Difference	Coordinate
1	0g 4%	21
2	Canned vegetable and Fruits for good nutrition Try a variety of key food's Vs. Fruits for good nutrition Try a variety of key food's Canned vegetable and	22, 32
3	Extra word "Cranberry" 13	

The images are 200 pixels per inch. The coordinate 21 means this: starting from upper left corner, go 2 inches to the right, 1 inch down and look at the 1 inch square box.

18.3 Templates

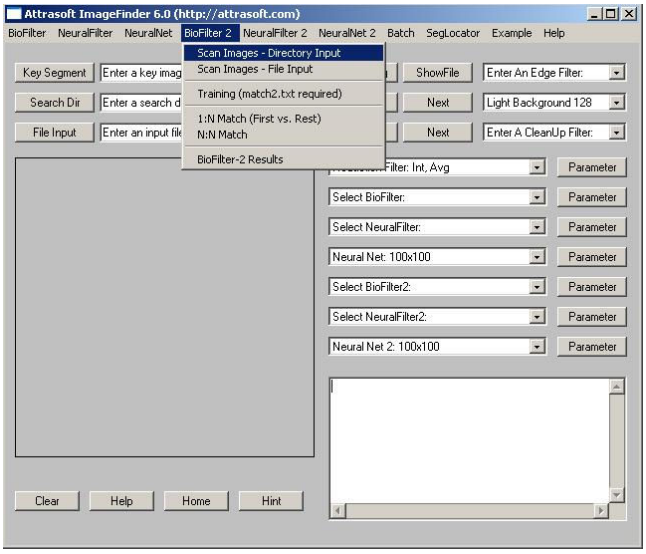


Figure 18.3 BioFilter2 Menu.

BioFilter II is exactly the same as **BioFilter**, except it operates on a section of an image, instead of whole image. It will convert a section of an image into a template. In this section, we will see both File Input and Directory Input.

18.3.1 File Input

1. Click the "File Input" button and open "biofilter2_ex1_input.txt; you will see:

9961400239\9961400239 1001.jpg
9961400239\9961400239 1002.jpg

2. Click "BioFilter 2/Scan Images – File Input" to convert images into template records.

3. Open d1.txt to see the results.

18.3.2 Directory Input

1. Click the "Search Dir" button and select 9961400239\9961400239 1001.jpg.

2. Click "BioFilter 2/Scan Images – Directory Input" to convert images into template records.

3. Open d1.txt to see the results.

18.4 Training

Unlike the **BioFilter**, which supports unsupervised learning, the **BioFilter II** must be trained. Training teaches the **BioFilter II** what to look for.

Each filter is trained differently. For the **BioFilter II**, training requires two files, d1.txt and match2.txt:

- D1.txt is the record file, which contains many records. Each image is converted into multiple records. A record represents features of an image segment in a feature space.
- Match2.txt is a list of matching pairs. This file will teach the **ImageFinder** who will match with whom.

These two files for training are fixed; you cannot change the names of these two files. You obtain d1.txt automatically, as you saw in the last section of this chapter. You have to prepare match2.txt for each problem.

The match2.txt looks like this:

```
1591
1       9961400239 1001_00 9961400239 1002_00
2       9961400239 1001_01 9961400239 1002_01
3       9961400239 1001_02 9961400239 1002_02
4       9961400239 1001_03 9961400239 1002_03
5       9961400239 1001_04 9961400239 1002_04
…
```

Line 1 is the number of matches in this file. This match file indicates images 9961400239 1001, section 00, will match with image 9961400239 1002, section 00. Here section (0,0) is the upper left corner. Each line has the following format:

Number, tab, filename1, tab, filename1, **tab.**

Note:

- **You must have a tab at the end of each line;**
- **The file names do not contain ".jpg".**

There are two common errors:

- The last Tab is missing;
- The number of rows is less than the first number in the file.

Once you get the two files prepared, click "BioFilter 2\Train (match2.txt required)" to train the **BioFilter II**. After training, you can use these commands:

"BioFilter 2/1:N Match (First vs. Rest)"
"BioFilter 2/N:N Match"

The 1:N or N:N Match is based on the d1.txt file. In 1:N Matching, the first image in d1.txt will match the rest of the images. In N:N Matching, each image in d1.txt will match against the rest of the images.

18.5 Training the BioFilter II

To continue the label recognition example, we must prepare d1.txt and match2.txt file for training now. The data used for training consists of 50 matching pairs.

The data is in " \Biofilter2_good".

You must prepare d1.txt in piecemeal for 2 reasons:

(1) The **BioFilter II** will use the first image in the input file or a directory to determine the region of interest. Because the images are different in size, you have to convert each image separately.

(2) It will take a long time to convert images into **BioFilter II** templates if many images are used, because the **BioFilter II** keeps all images in RAM.

All examples in this chapter use the same setting. You can click "Example/BioFilter 2/Label Match Setting" or any example under "Example/BioFilter 2" to open the parameters and then, click "Batch/Load" command to load. If you want to work manually, here are the parameters:

Edge Filters: Sobel 1
Threshold Filters: Dark Background 128
Clean Up Filter: Small
BioFilter: CL
BioFilter II: CL1

For example, to generate d1.txt for images in folder ".\9961400239\",

- Click the "Search Dir" button and select \9961400239.
- Click "BioFilter 2/Scan Images – Directory Input" to convert images into template records.

We have prepared the d1.txt for you. Open biofilter2_d1.txt and save it to d1.txt.

The second training file is match2.txt, which specifies who should match with whom. This file is already prepared for you. Go to the **ImageFinder** folder. (The default folder is C:\program files\Attrasoft\ImageFinder 6.0\"and open the file, biofilter2_match2.txt. Save it to match2.txt (overwrite the existing file). Now the training file is prepared.

To train the **BioFilter II**, click "BioFilter 2/Train (match2.txt required)". Now the **BioFilter II** is trained for the label recognition problem. You can save the trained results by clicking "Batch/Save" to save the trained **BioFilter II**.

18.6 1:N and N:N Matching Example 1

1:N Matching compares the first image in the template file, d1.txt, with the rest of the images in the file; the d1.txt, in turn, is generated by the input file or input directory. The images used in this example can be seen in Figure 18.1.

The data used in this example will be in biofilter2_ex1_input.txt. Operations for 1:N Matching:

- Click "Example/BioFilter 2/Label Match Setting" to open the parameters;
- Click "Batch/Load" command to load parameters. At this point, the **BioFilter II** is also trained.
- Click the "File Input" button and open "biofilter2_ex1_input.txt; you will see:

  9961400239\9961400239 1001.jpg
  9961400239\9961400239 1002.jpg

- Click "BioFilter 2/Scan Images – File Input" to convert images into template records.
- Click "BioFilter 2/1:N Match (First vs. Rest)" to make a 1:N match.

The results look like this:

C:\…\9961400239\9961400239 1001
0 0
0 1
Match!
0 2
Match!
…
3 9
Match!
Image Match

The **BioFilter II** divides the images into smaller sections. In this example, the sections are 00, 01, … 39. Here section (0,0) is the upper left corner. The output shows the matching result in each section. The above example shows the "newly captured image", 9961400239\9961400239 1001.jpg, matches the "previously stored image", 9961400239\9961400239 1002.jpg, in all sections. We now continue the example:

- Click "BioFilter 2/N:N Match" to make a N:N match.

The results are:

C:\…\9961400239 1001
0 0
Match!
0 1
Match!
0 2
Match!
…
3 9
Match!
Image Match!

C:\…\9961400239 1002
0 0
Match!
0 1
Match!
0 2
Match!
…
3 9
Match!
Image Match!

These results show 9961400239 1001 matches 9961400239 1002 and 9961400239 1002 matches 9961400239 1001.

18.7 1:N and N:N Matching Example 2

In this section, we will present an example, which does not match. The input file is

biofilter2_ex2_input.txt.

The operations are:

- Click "Example/BioFilter 2/Label Match Setting" to open the parameters;
- Click "Batch/Load" command to load parameters. At this point, the **BioFilter II** is also trained.
- Click the "File Input" button and open "biofilter2_ex2_input.txt; you will see:

biofilter2_good10/l01016key.jpg
biofilter2_bad10/l01016key.jpg

- Click "BioFilter 2/Scan Images – File Input" to convert images into template records.
- Click "BioFilter 2/1:N Match (First vs. Rest)" to make a 1:N match.

The results are:

C:\…\biofilter2_good10/l01016key
0 0
Match!
…
1 3
No Match!
…
3 3
No Match!
…
6 0
No Match!
…
Image Does Not Match!

18.8 Two-layer Neural Net Architecture

In the last two sections, we present a Matching example and a "No Match" example, using the **BioFilter II** alone. The examples in section use the two-layered Neural Network Architecture:

BioFilter
BioFilter II

The objective is to compare two basically identical images and identify the minor differences. Two different images do not fall into this category. The **BioFilter** will serve as a filter to get rid off different images and the **BioFilter II** will identify the minor differences.

The three examples used in this section are 1:N Match, N:N Match in section 6, and 1:N Match in section 7. This process will take many steps; the batch job is perfect for this type of work.

The operation for the first example is:

Click "Example/BioFilter 2/Label example 1 (1:N)";
Click "Batch/Run".

The operation for the second example is:

Click "Example/BioFilter 2/Label example 2 (N:N)";
Click "Batch/Run".

The operation for the third example is:

Click "Example/BioFilter 2/Label example 3 (1:N)";
Click "Batch/Run".

The results of these runs should match results earlier.

19. Neural Filter II

The **BioFilter II** will find about 80% of the unmatched images and the **Neural Filter II** will find an additional 19%, leaving only about 1% of the unmatched images for the **ABM Filter II**.

This chapter will demonstrate how **Neural Filter II** works using the same examples of the last chapter. Again, the problem is to compare the newly capture images with several previously stored images. These images are basically identical with some minor differences. The data is entered via an input file or input directory. The first image is the newly captured and the rest are previously stored images. These images are converted into templates in the last chapter. The matching is based on these templates.

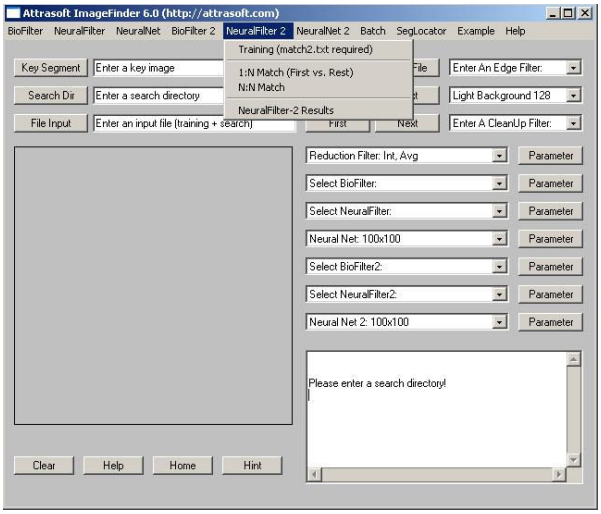


Figure 19.1 NeuralFilter II Menu.

Operating **Neural Filter II** is exactly the same as the **BioFilter II**. The **Neural Filter II** is more accurate than the **BioFilter II**.

19.1 Overview

The **Neural Filter II** matching will be done in several steps:

- Initialization
- Converting Images to Records if you have not done so in the **BioFilter II**
- Training
- Template Matching

Let us look at each phase.

Initialization

Initialization sets the **ImageFinder** parameters.

When **Converting Images to Records** if you have not done so in the **BioFilter II**, use these two commands:

BioFilter 2/Scan Images – File Input

BioFilter 2/Scan Images – Directory Input

Training

Neural Filter II training requires the same input files as the BioFilter II**.** Training uses the data collected in advance to teach the Neural Filter II how to match**. Training requires two files, d1.txt and match2.txt:**

- D1.txt is the record file, which contains many records. Each image is converted into several records. A record represents features of a section of an image in a feature space.
- Match2.txt is a list of matching pairs. This file will teach the **ImageFinder** who will match with whom.

Template Matching

Both the **BioFilter II** and **Neural Filter II** will do the template matching.

19.2 Training

Training teaches the **Neural Filter II** what to look for. Each filter is trained differently.

If you have done the **BioFilter II** training in the last chapter, all you have to do is to click

"NeuralFilter 2\Train (match2.txt required)" to train the **Neural Filter II**.

Both the **BioFilter II** training and **Neural Filter II** training, require two files, d1.txt and match2.txt:

- D1.txt is the record file, which contains many records. Each image is converted into several records. A record represents features of a section of an image in a feature space.
- Match2.txt is a list of matching pairs. This file will teach the **ImageFinder** who will match with whom.

These two files for training are fixed; you cannot change the names of these two files. You obtain d1.txt automatically, as you run the following commands:

BioFilter 2/Scan Images – File Input

BioFilter 2/Scan Images – Directory Input

You have to prepare match2.txt for each problem, which has the same format as match.txt. Please read the last chapter to see the format of this file.

The data used for training consists of 50 matching pairs. The data is in \Biofilter2_good. You must prepare d1.txt in piecemeal for 2 reasons:

- The **BioFilter II** will use the first image in the input file or a directory to determine the region of interest. Because the images are different in size, you have to convert each image separately.
- It will take a long time to convert images into **BioFilter II** templates if many images are used, because the BioFilter II keeps all images in RAM.

For example, to generate d1.txt for images in folder ".\9961400239\",

- Click the "Search Dir" button and select \9961400239.
- Click "BioFilter 2/Scan Images – Directory Input" to convert images into template records.

All examples in this chapter use the same setting. You can click "Example/Neural Filter 2/Setting" to open the parameters and then, click "Batch/Load" command to load. If you want to work manually, here are the parameters:

Edge Filters: Sobel 1
Threshold Filters: Dark Background 128
Clean Up Filter: Small
BioFilter: CL
BioFilter II: CL1

We have prepared the d1.txt for you. Open biofilter2_d1.txt and save it to d1.txt.

The second training file is match2.txt, which specifies who should match with whom. This file is already prepared for you. Go to the **ImageFinder** folder. (The default folder is C:\program files\Attrasoft\ImageFinder 6.0\"and open the file, biofilter2_match2.txt. Save it to match2.txt (overwrite the existing file). Now the training file is prepared.

To train the **Neural Filter II**, click "Neural Filter 2/Train (match2.txt required)". Now the **Neural Filter II** is trained for the label recognition problem. You can save the trained results by clicking "Batch/Save".

19.3 1:N and N:N Matching Example 1

1:N Matching compares the first image in the template file, d1.txt, with the rest of the images in the file; the d1.txt, in turn, is generated by the input file or input directory.

The examples here are the same as the last chapter. The images used in the examples can be seen from the last chapter.

The data used in this example will be in biofilter2_ex1_input.txt. Operations for 1:N Matching:

- Click "Example/Neural Filter 2/Setting" to open the parameters;
- Click "Batch/Load" command to load parameters. At this point, the **Neural Filter II** is also trained.
- Click the "File Input" button and open "biofilter2_ex1_input.txt; you will see:

9961400239\9961400239 1001.jpg
9961400239\9961400239 1002.jpg

- Click "BioFilter 2/Scan Images – File Input" to convert images into template records.
- Click "Neural Filter 2/1:N Match (First vs. Rest)" to make a 1:N match.

The results look like this:

C:\…\9961400239\9961400239 1001
0 0
0 1
Match!
0 2
Match!
…
3 9
Match!
Image Match!

Like the **BioFilter II**, the **Neural Filter II** divides the images into smaller sections. In this example, the sections are 00, 01, … 39. Here section (0,0) is the upper left corner. The output shows the matching result in each section. The above example shows the "newly captured image", 9961400239\9961400239 1001.jpg, matches the "previously stored image", 9961400239\9961400239 1002.jpg, in all sections. We now continue the example:

- Click "Neural Filter 2/N:N Match" to make a N:N match.

The results are:

C:\…\9961400239 1001
0 0
Match!
0 1
Match!
0 2
Match!
…
3 9
Match!
Image Match!

C:\…\9961400239 1002
0 0
Match!
0 1
Match!
0 2
Match!
…
3 9
Match!
Image Match!

This results show 9961400239 1001 matches 9961400239 1002, and 9961400239 1002 matches 9961400239 1001.

19.4 1:N Matching Example 2

In this section, we will present an example, which does **Not Match.** The input file is biofilter2_ex2_input.txt. The operations are:

- Click "Example/BioFilter 2/Label Match Setting" to open the parameters;
- Click "Batch/Load" command to load parameters. At this point, the **BioFilter II** is also trained.
- Click the "File Input" button and open "biofilter2_ex2_input.txt; you will see:

biofilter2_good10/l01016key.jpg

biofilter2_bad10/l01016key.jpg

- Click "BioFilter 2/Scan Images – File Input" to convert images into template records.
- Click "BioFilter 2/1:N Match (First vs. Rest)" to make a 1:N match.

The results are:

C:\…\biofilter2_good10/l01016key
…
1 3
No Match!
..
2 2
No Match!
2 3
No Match!
…
3 3
No Match!
…
4 3
No Match!
Image Does Not Match!

This result is different from the **BioFilter II** result of the last chapter, which is:

C:\…\biofilter2_good10/l01016key
0 0
Match!
…
1 3
No Match!
…
3 3
No Match!
…
6 0
No Match!
…
Image Does Not Match!

19.5 Two-layer Neural Net Architecture

In the last two sections, we presented a Matching example and a Non-Matching example, using the **Neural Filter II** alone. The examples in section use the two-layered Neural Network Architecture:

BioFilter
Neural Filter II.

The objective is to compare two basically identical images, and identify the minor differences. Two different images do not fall into this category. The **BioFilter** will serve to get rid off different images and the **Neural Filter II** will identify the minor differences.

The three examples used in this section are 1:N Match, N:N Match in section 3, and 1:N Match in section 4. This process will take many steps; the batch job is perfect for this type of work.

The operation for the first example is:

- Click "Example/NeuralFilter 2/Label example 1 (1:N)";
- Click "Batch/Run".

Two files will be opened: b1.txt is the result of the **BioFilter** and e1.txt is the results of **Neural Filter II**. The operation for the second example is:

Click "Example/NeuralFilter 2/Label example 2 (N:N)";
Click "Batch/Run".

The operation for the third example is:

Click "Example/NeuralFilter 2/Label example 3 (1:N)";
Click "Batch/Run".

The results of these runs should match earlier results.

20. NeuralNet Filter II

The **BioFilter II** will find approximately 80% of the unmatched images and the **Neural Filter II** will find an additional 19%, leaving only about 1% of the unmatched images for the **Neural Net Filter II** or **ABM Filter II**.

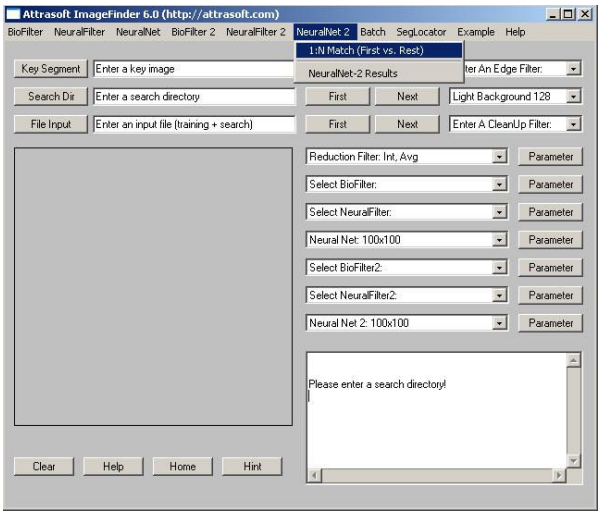


Figure 20.1 NeuralNet II Menu.

This chapter will demonstrate how **Neural Net Filter II** or **ABM Filter II** works using the same examples of the last chapter, label verification. Again, the problem is to compare the newly captured images with several previously stored images. These images are basically identical with some minor differences. The data used by the filter is entered via **input file** only. The first image in the input file is the newly captured image and the rest are previously stored images.

The Neural Net Filter II or ABM Filter II operates on top of the Neural Filter II of the last chapter.

20.1 Overview

The **Neural Net Filter II** matching will be done in several steps:

- Initialization
- Training and 1:N Matching

Let us look at each phase.

Initialization

Initialization sets the **ImageFinder** parameters.

Training and 1:N Matching

ABM Filter II operates on top of the Neural Filter II of the last chapter. After the **Neural Net Filter II** computation is complete, click "NeuralNet 2/1:N Match (First vs. Rest)" to make training and 1:N Matching. The data is entered via **input file** only. The first image in the input file is the training image and the rest of the images are compared with the training image(s).

20.2 Positive Match Examples

There are three positive matching examples and three negative matching examples. All of the examples can be reached by menu item "Example/NeuralNet 2".

Let's go over the first positive matching example. Click "Example/NeuralNet 2/Match 1" to open the batch code for the first example. The batch code is:

```
[ImageFinder 6.0]
        executionCode=1039
[Input]
        trainFileName=.\biofilter2_bad10\358
2600598.jpg
        searchDirName=NA
        fileInputName=neuralnet2_ex1_input.
txt
…
```

The training image is .\biofilter2_bad10\3582600598.jpg, which is displayed below:

Figure 20.2 Training image.

The input file is neuralnet2_ex1_input.txt, which is:

biofilter2_good10\3582600598 1.jpg
biofilter2_good10\3582600598 2.jpg

The stored image is displayed below:

Figure 20.3 Stored Image.

Now click "Batch/Run", the software will go through 4 matching filters:

BioFilter I
BioFilter II
Neural Filter II
Neural Net Filter II.

You will get 4 output files, one for each filter:

BioFilter I: b1.txt
BioFilter II: e1.txt
Neural Filter II: e1.txt
Neural Net Filter II: f1.txt.

B1.txt will show you the training image matched both images in the input file. BioFilter identifies whether they are similar images or not. After that, the image is divided into 40 smaller segments, with coordinates 00, 01, … 39. Each segment is 1 square inch. BioFilter II and Neural Filter II will make feature space recognition on these square inch

images and the Neural Net Filter II will make input space recognition. In this example, all four filters reported positive identification.

There are two more positive match examples. **Example 2** is similar to Example 1, with all 4 filters reporting positive matches. **Example 3** contains some errors: the first three filters reported positive matches and the Neural Net II filter reported one mismatch in one of the squares, representing a false alarm.

20.3 Negative Match Examples

Let's go over the first Negative Matching Example. Click "Example/NeuralNet 2/No Match 1" to open the batch code for this example. The batch code is:

```
[ImageFinder 6.0]
        executionCode=1039
[Input]
        trainFileName=.\biofilter2_bad10\l330
63rk-f.jpg
        searchDirName=NA
        fileInputName=neuralnet2_ex4_input.
txt
…
```

The training images is =.\biofilter2_bad10\l33063rk-f.jpg, which is displayed below:

The input file is neuralnet2_ex4_input.txt, which is:

biofilter2_good10/l33063rk-f.jpg

The image looks like this:

The errors are:

```
l33063rk-f.jpg
Error 1:
        100 PRUNE
        vs
        100% PRUNE
        Coordinates: 01 02 12

Error 2:
        %
        vs
        U
        Coordinates: 22 32
```

In this example, the first three filters passed the images and the NeuralNet II filter caught the error at 02, which is blank in one image and has 10 in the other.

21. API

This chapter introduces the standard **Application Programming Interface (API)** of the **ImageFinder** for software developers.

The neural network level library is called **PolyApplet** and the image recognition level library is called **TransApplet.** You must order the **ImageFinder** libraries separately.

Very often a customized **ImageFinder** will be ordered before ordering the **ImageFinder** library as a proof-of-concept test, based on a specific set of images.

21.1 ImageFinder Internal Structures

ImageFinder adopts a layered approach to image recognition. An image recognition application is roughly divided into:

Level 5: User Interface;
Level 4: Data Management;
Level 3: Image-Matching Layer;
Level 2: Scanner, Camera, and their drivers;
Level 1: PC with Windows.

The reason for the above approach is the **ImageFinder** library will be integrated into a database application. The Stand-alone software is roughly divided into 3 layers:

Level 3: User Interface;
Level 2: Image-Matching Layer;
Level 1: PC with Windows.

For the **ImageFinder**, the Image-Matching Layer is divided into:

Image Preprocessing
Normalization
Feature Recognition
Pixel Recognition
Sub-Pixel Recognition

Each of the sub-layers is further divided into filters:

Image Preprocessing
    Edge Filters;
    Threshold Filters; and
    Clean Up Filters.
Normalization
    Reduction Filter.
Feature Recognition
    BioFilter;
    Neural Filter.
Pixel Level Recognition
    NeuralNet Filter or ABM Filter.
Multi-layered Pixel Recognition
    BioFilter 2;
    NeuralFilter 2;
    ABM Filter 2.
…

The API is basically organized accordingly. Each section will have three sub-sections:

- Filters
- Parameters
- Commands

The Filter section sets and gets the filter; the Parameter section sets and gets the parameters, and the command section executes the commands.

21.2 Software Structure

The software structure is:

Main class (TransApplet6x):
    Image Preprocessing Class
    Normalization Class
    Feature Recognition Class
    Pixel Recognition Class
    Sub-Pixel Recognition Class

Each of these classes is further divided into several classes; for example,

Image Preprocessing Class
- Edge Filter class
- Threshold Filter class
- Clean Up Filter class

Again, each of these classes is further divided into several classes; for example,

Edge Filter Class
- Sobel 1 class
- Sobel 2 class
- …

21.3 Standard API

The software was originally developed under Microsoft Visual Studio 6.0, then later converted into J#.Net. Therefore, developers must have both .Net Framework and J# Redistributable package installed. The following specification still has the Java flavor:

```
Public class TransApplet6x
{

//Input

void setKeySegment (String s);
String getKeySegment ();
boolean getKeySegmentStatus();

void setSerachDir (String s);
String getSerachDir ();
boolean getSerachDirStatus();

void setFileInput (String s);
String getFileInput ();
boolean getFileInputStatus();


//Image Preprocessing
//1. Filters Section
void setEdgeFilters(int);
int getEdgeFilters();

void setThresholdFilters (int);
int getThresholdFilters ();

void setCleanUpFilter (int);
int getCleanUpFilter ();

//2. Parameter Section
//color: R1, R2, R3, G1, G2, G3, B1, B2, B3
void setR1(int);
int getR1();
…

//3. Command Section: NA

//Normalization
//1. Filter section
void setReductionFilter (int);
int getReductionFilter ();

//2. Parameter Section
void setSegmentCut(int);
int getSegmentCut(i);

void setSizeCut(int);
int getSizeCut();

void setBorderCut(int);
int getBorderCut();

void setLookAtX(int);
int getLookAtX();

void setLookAtY(int);
int getLookAtY();

void setLookAtXLength(int);
int getLookAtXLength();

void setLookAtYLength(int);
int getLookAtYLength();

//3. Command Section: NA

//Feature Recognition
//BioFilter
//1. Filter Section

void set BioFilter (int);
```

```
int getBioFilter();

//2. Parameter section

void setBioFilterPercent(int);
int getBioFilterPercent();

void setBioFilterMode(int);
int getBioFilterMode();

void setBioFilterFinal(int);
int getBioFilterFinal();

void setBioFilterCutOff(int);
int getBioFilterCutOff();

//3. Command Section

boolean scanImageDir();
boolean scanImageFile();

boolean bioFilterTrain();

Boolean bioFilterNNMatchUntrained();
Boolean bioFilter1NMatchUntrained();

Boolean bioFilterNNMatch();
Boolean bioFilter1NMatch();

Boolean bioFilterCheck();

String getBioFilterResults(int);

boolean scanImageDir(int);
boolean scanImageFile(int);

//Neural Filter
//1.  Filter section
void setNeuralFilter(int);
int getNeuralFilter();

//2. Parameter section
void setNeuralFilterPercent (int);
get getNeuralFilterPercent ();

void setNeuralFilterMode (int);
get getNeuralFilterMode ();

void setNeuralFilterSize (int);
get getNeuralFilterSize ();

void setNeuralFilterFinal (int);
int getNeuralFilterFinal ();

void setNeuralFilterCutOff (int);
int getNeuralFilterCutOff ();

//3. Command Section
boolean neuralFilterTrain();

Boolean neuralFilterNNMatch();
Boolean neuralFilter1NMatch();
Boolean neuralFilter1N1Match();

Boolean neuralFilterCheck();


String getNeuralFilterResults(int);

Boolean neuralFilterNMMatch(String a1,
String a2, String b2);

//Pixel Level Recognition
//ABM Filter or NeuralNet Filter
//1. Filter Section
void setNeuralNet(int);
int getNeuralNet();

//2. Parameter Section

void setSegmentX (int);
int getSegmentX ();

void setSegmentY(int);
int setSegmentY();

void setSegmentXlength(int);
int getSegmentXlength();

void setSegmentYLength(int);
int getSegmentYLength();

void setSymmetry(int);
int getSymmetry();

void setRotationType(int);
int getRotationType();
```

```
void setTranslationType(int);
int getTranslationType();

void setScalingType(int);
int getScalingType();

void setSensitivity(int);
int getSensitivity();

void setBlurring(int);
int setBlurring();

void setInternalWeightCut(int);
get setInternalWeightCut();

void setExternalWeightCut(int);
int setExternalWeightCut();

void setSegmentSize(int);
int getSegmentSize();

void setImageType(int);
int getImageType();

void setFileDisplayType(int);
int setFileDisplayType();

void setAutoSegment(int);
int getAutoSegment();

void setNeuralNetMode(int);
int setNeuralNetMode();

//3. Command Section

boolean NeuralNetTrain();
boolean NeuralNetRetrain();

Boolean NeuralNet1NSearch();
Boolean NeuralNetNNSearch();
Boolean NeuralNetSort();

Boolean NeuralNet1NSearchLong();
Boolean NeuralNetSortLong();
Void getLongSearchDir(String s);
String getLongSearchDir();

Boolean NeuralNet1NSearchFile();
Boolean NeuralNetNNSearchFile();
Boolean NeuralNetSortFile();

String getNeuralNetResults(int);
… …

}
```

After the execution, the results go to the files. You can process the output by reading and processing the output files from the **TransApplet**.

21.4 Example: Unsupervised Learning

Assume we want to match an image, C:\abc\input.jpg, with the images in C:\efg:

```
TransApplet60 x = new TransApplet60 ();

Boolean unsupervisedLearning_1_to_N ()
{
x.setKeySegment ("c:\abc\input.jpg");
if ( ! getKeySegmentStatus() )
        return false;

x.setSearchDir ("c:\efg");
if ( ! getSearchDirStatus() )
       return false;

return x.bioFilter1NMatchUntrained();

}
```

The results will be in b1.txt.

22. ImageFinder for Dos

ImageFinder for Dos is a quick way to implement an image recognition application. An image recognition application is roughly divided into:

Level 5: User Interface;
Level 4: Data Management;
Level 3: Image-Matching Layer;
Level 2: Scanner, Camera, and their drivers;
Level 1: PC with Windows.

The quick and dirty way to implement a system for a feasibility study can be done very quickly as follows:

Level 1: Find a computer.
Level 2: Select an off-the-shelf camera or scanner and connect it to your computer. Install the software driver.
Level 3: Purchase the **ImageFinder for Dos**.
Level 4: Create a few directories and store both newly captured images and the previously stored images there.
Level 5. Write a GUI, which captures images by issuing keystroke to the camera or scanner driver and saves the images to a directory. Issue a dos command to call the **ImageFinder for Dos** for image recognition. Finally, process the output files of the **ImageFinder for Dos.**

Attrasoft has an example for the Logitech camera and an example for the Fujitsu scanner. The source codes for Level 5 (Graphical User Interface) for both systems are available for licensed users of the **ImageFinder for Dos**. The **ImageFinder for Dos** has to be purchased separately.

The ImageFinder for Dos consists of an exe file, "attrasoft60dos.exe" and several text files. Copy all files from the "**ImageFinder for Dos**" CD to a directory and run with the command "attrasoft60Dos".

To use the dos version, you must use the **ImageFinder** for Windows**.** The basic command is:

C:\…>imagefinder60dos 1

This is equivalent to the following in the Window version:

- Click Batch/Open
- Click Batch/Run
- Do not show **ImageFinder** GUI.

Basically, you are running a Window's version without the **ImageFinder** Graphical User Interface. The parameters for your problem are all in the batch file, abm60.txt.

When you write your application, you will write a Level 5 GUI, most likely using Visual Basic. Your GUI will prepare the input file, abm60.txt; then you run the dos version; finally you will process the output files of the **ImageFinder for Dos**.

22.1 Introduction

There are several off-the-shelf components in the **ImageFinder** family:

- **ImageFinder for Windows;**
- **ImageFinder for Dos;**
- **TransApplet;**
- **PolyApplet.**

Attrasoft Component-Object structure consists of three layers:

- Application Layer
- Presentation Layer
- Neural Network Layer

The **PolyApplet** is the Neural Network Layer developer tool, the **TransApplet** is the Presentation Layer developer tool, and the

ImageFinder for Dos is the Application Layer developer tool.

ImageFinder for Dos is the **ImageFinder for Windows** without the Graphical User Interface (GUI). Other than that, these two are exactly the same.

The Dos version has many advantages over the Windows version:

- The Dos version can start as a Windows version;
- You can run multiple tasks in a single Dos batch file; and
- You can have programming control over the **ImageFinder**.

You should become familiar with the **ImageFinder** for Windows first. There are two commands in the **ImageFinder** for Windows:

- Batch/Open
- Batch/Run

ImageFinder for Dos allows you to run these commands from the Dos prompt, from a Dos batch file, or from a Visual Basic program, … . This allows developers to quickly integrate the **ImageFinder** into their applications.

22.2 Batch Files

The Batch files specify the **ImageFinder** setting. The **ImageFinder** for Windows uses 5 files, abm60.txt, abm60_1.txt, abm60_2.txt, … and abm60_5.txt. When you use the **ImageFinder** for Windows and obtain satisfactory results, you can click the following button to save your results in 1 of 5 files:

Batch/Save
Batch/Save 2
Batch/Save 3
Batch/Save 4
Batch/Save 5

Later, to duplicate your earlier results saved in the first file, click:

- Batch/Open;
- Batch/Run.

To duplicate your earlier results saved in the second file, click

- Batch/Open 2;
- Batch/Run.

…

You should never try to write these files yourself. Instead, use the **ImageFinder** for Windows to find the correct setting and click the "Save" button to obtain the codes.

22.3 Batch Commands

There are five possible commands:

C:\…>imagefinder60dos 1
C:\…>imagefinder60dos 2
C:\…>imagefinder60dos 3
C:\…>imagefinder60dos 4
C:\…>imagefinder60dos 5

corresponding to 5 different batch files. In addition, you can use the following command for the Windows version:

C:\…>imagefinder60dos x 1

Here x = 1, 2,3, 4, or 5. This command will run as the dos version; at the end, it will bring back the Window GUI. This is often necessary for debugging.

23. Reference Manual

This manual is divided into the following sections:

1. Input
2. Image Processing
3. Normalization
4. BioFilter
5. Neural Filter
6. Neural Net Filter
7. Batch
8. BioFilter 2
9. Neural Filter 2
10. Neural Net Filter 2
11. SegLocator
12. Miscellaneous
13. Examples

Each section is further divided into 3 parts:

- Filter Selection
- Menu Items
- Parameters

Filter Selection uses the Drop Down List. Menu Items use the menu bar. Parameters are set by using the "Parameter" button to open a new window.

23.1 Input

Key-Segment Button

Use the "Key-Segment" button to specify a key. For example, click the button, go to directory, "C:\images\", and then click "attrasoft.jpg". The selected image will be shown in the **ImageFinder** to indicate the specification is successful and the key-image is ready for training or retraining.

Search Dir Button

Use the "Search Dir" button to specify a search-directory. To select a search-directory, go to the directory; **then click any file in the directory.** The first image in the search-directory will be shown in the **ImageFinder** to indicate the specification is successful and the search-directory is active for retrieval.

File Input Button

Use the "File Input" button to specify a search-file.

Show Seg Button

Use the "Show Seg" button to display the selected training image.

ShowFile Button

Use the "ShowFile" button to display the selected search-file.

First, Next Button for Search Dir

Use the "First" button to display the first image in the search-directory. Use the "Next" button to display the next image in the search-directory.

First, Next Button for Search-File

Use the "First" button to display the first image in the search-file. Use the "Next" button to display the next image in the search-file.

23.2 Image Processing

Edge Filter Drop Down List

Use the "Edge Filter Drop Down List" to select an Edge Filter. The Edge Filter is an optional filter. The Edge Filters attempt to exaggerate the main feature(s) a user is looking for. The Edge Filters usually require a dark threshold filter. The Edge Filters extract and enhance edges & contours in an image by expressing intensity differences (gradients) between neighboring pixels as an intensity value. The basic variables are the differences between the top and bottom rows; the differences between the left and right columns; and the differences between the center point and its neighbors.

The batch codes for the Edge Filters are:

Code	Meaning
0	No Edge Filter
1	Sobel 1 (Prewitt)
2	Sobel 2 (Sobel)
3	Sobel 3
4	Sobel 4
5	Gradient
6	Gradient, 45°
7	Sobel 1, 45°
8	Sobel 1, - 45°
9	Laplacian 4
10	CD 11
11	FD 11
12	FD 9
13	FD 7
14	Laplacian 5
15	Laplacian 8
16	Laplacian 9
17	Laplacian 16
18	Laplacian17

Threshold Filter Drop Down List

Use the "Threshold Filter Drop Down List" button to set the Threshold Filter. The Threshold Filters attempt to suppress the background. The Threshold Filter is not optional; if you do not set the Threshold Filter, the default filter will be used.

Choose an Edge Filter and a Threshold Filter where the sample objects will stand out, otherwise change the filter. If you do not have a filter, a customized filter has to be built. DO NOT make too many things stand out, i.e. as long as the area of interest stands out, the rest should show as little as possible.

Once you make a selection, the objects in the training images are black and the background is white. **You should make the black area as small as possible, as long as it covers the key-segment(s). Otherwise, switch to a different filter.**

CleanUp Filter Drop Down List

Use the "CleanUp Filter Drop Down List" to select a Clean Up Filter. The CleanUp Filters will smooth the resulting image to reduce recognition error. The CleanUp Filter is an optional filter.

23.3 Normalization

Reduction Filter Drop Down List

Use the "Reduction Filter Drop Down List" to select a Reduction Filter. When reducing images, a scaling factor can be introduced easily. Although scaling symmetry can compensate for this scaling factor, the scaling symmetry is expensive. The Internal Reduction parameter is introduced to avoid unnecessary scaling.

There are several ways to reduce images:

- Integer;
- Real; or
- All images are reduced by a same amount (Customized Version Only).

Integer Reduction

Images are reduced by an integer factor to maximally fit 100x100 without distortion. For example, a 350x230 image will be reduced to 87x57.

Real Reduction

Images are reduced by a real number to maximally fit 100x100 without distortion. For example, a 350x230 image will be reduced to 100x65.

All

All training images and images in the search-directory are

reduced by the same integer to fit 100x100 without distortion.

Within each type of reduction, there are 3 more settings:

- Avg: AOI (Area Of Interest) is half black and half white;
- Max: AOI is mostly white, or the black areas are thin lines; or
- Min: AOI is mostly black.

Reduction Filter Parameter / Segment-Cut Button

Use the "Segment-Cut" button to shape the segment considered by the **ImageFinder**. The Segment-Cut parameter ranges from 0 to 12. This parameter deals with the edges of segments in the images. The larger this parameter is, the smaller the segment the **ImageFinder** will use. The possible settings of this parameter in the user interface are: 0, 1, 2, ..,and 12. To set the parameter, keep clicking the button.

Reduction Filter Parameter/Size Cut Button

Use the "Size-Cut" button to limit the dimensions of images to be searched. In some applications, the users only want to search images of certain dimensions and ignore other images.

The dimension setting ranges from 0 to 9. To set the parameter, keep clicking the "Dimension" button; the setting will switch from one to the next each time you click the button.

If the setting is 0, this parameter will be ignored. If the parameter is 1, then the longest edge of the image to be considered must be at least 100, but less than 199. If the parameter is 2, then the longest edge of the image to be considered must be at least 200, but less than 299, …

Reduction Filter Parameter / Border Cut

Use the "Border-Cut" button to ignore the sections of the image near the borders. The Border-Cut parameter ranges from 0 (no cut) to 9 (18% border cut). The possible settings in the user interface are: 0, 1, 2, ..,and 9. Assume an image is (0,0; 1,1); setting Border-Cut to 1 means the **ImageFinder** will look at the section (0.02, 0.02; 0.98, 0.98); setting Border-Cut to 2 means the **ImageFinder** will look at the section (0.04, 0.04; 0.96, 0.96); … . To set the parameter, keep clicking the button.

Reduction Filter Parameter / Look-At Area

The "Look-At Area" is the area the **ImageFinder** will use. A 100 x 100 window specifies a whole image. In the Integer-Reduction, the actual area can be less than 100x100. The Look-At Area is specified by 4 numbers:

(x, y, w, h)

(x, y) are the coordinates of the upper-left corner and (w, h) are the width and height of the Look-At Window.

To use this Look-At Window, enter values for (x, y, w, h) in the four textboxes. Note that the image display area in the **ImageFinder** is 300x300, therefore, the training segment is specified within a 300x300 area. The Look-At Window is 100x100. The default value is (0,0,0,0), meaning the Look-At area setting is ignored.

23.4 BioFilter

BioFilter Drop Down List

Use the "BioFilter Drop Down List" to select a BioFilter. The main role of the BioFilter is (1) to make an assessment of data via unsupervised learning; and (2) to eliminate 80% of the mismatches.

"BioFilter/Scan Images - Directory Input" Menu Item

Use "BioFilter/Scan Images - Directory Input" menu item to convert images into records. The BioFilter is one of two filters in Feature Space image recognition (Image recognition is divided into Feature Space recognition and Input Space recognition. Feature Space recognition operates on signatures of images.) To convert images into templates:

- Click "Search Dir" button to specify the search-directory.
- Click menu item "BioFilter/Scan Images - Directory Input" to convert images to records. You should see the **ImageFinder** scan through the images at this point.

"BioFilter/Scan Images - File Input" Menu Item

Use "BioFilter/Scan Images - File Input" menu item to convert images into records. The BioFilter is one of two filters in Feature Space image recognition (Image recognition is divided into Feature Space recognition and Input Space recognition.
Feature Space recognition operates on signatures of images.) To convert images into templates:

- Click the "Input File" button to specify the search-file.
- Click menu item "BioFilter/Scan Images - File Input" to convert images to records. You should see the **ImageFinder** scan through the images at this point.

"BioFilter\Train (match.txt required)" Menu Item

Use "BioFilter\Train (match.txt required)" menu item to train the BioFilter. Training uses the data collected in advance to teach the BioFilter how to match. Training requires two files, a1.txt and match.txt:

- A1.txt is the record file, which contains many records. Each image is converted into a record. A record represents features of an image in a Feature Space.
- Match.txt is a list of matching pairs. This file will teach the **ImageFinder** who will match with whom.

"BioFilter/BioFilter 1:N Match (Untrained)" Menu Item

Use "BioFilter/BioFilter 1:N Match (untrained)" menu item to make an untrained 1:N matching. 1:N Matching compares one key image with the images in a search-directory or search-file; the key image is specified in the "Key Segment" textbox or selected by the "Key Segment" button. 1:N matching requires the images in the search-directory or search-file being converted into templates in advance. To make an 1:N matching:

- Click "Key Segment" button, and select an image;
- Click "BioFilter/BioFilter 1:N Match (untrained)".

The result is in file, b1.txt, which will be opened at the end of computation.

"BioFilter/BioFilter N:N Match (Untrained)" Menu Item

Use "BioFilter/BioFilter N:N Match (Untrained)" menu item to make an untrained N:N matching. N: N compares each image, specified in the search-directory or search-file, with every image in the search-directory or search-file. N:N matching requires the images in the search-directory or search-file being converted into templates in advance. The result is in file, b1.txt, which will be opened at the end of computation.

"BioFilter/BioFilter 1:N Match" Menu Item

Use "BioFilter/BioFilter 1:N Match" menu item to make a 1:N matching. 1:N Matching compares one key image with the images in a search-directory or search-file; the key image is specified in the "Key Segment" textbox or selected by the "Key Segment" button. 1:N matching requires (1) the images in the search-directory or search-file being converted into templates in advance; and (2) the

BioFilter being trained. To make an 1:N matching:

- Click "Key Segment" button, and select an image;
- Click "BioFilter/BioFilter 1:N Match".

The result is in file, b1.txt, which will be opened at the end of computation.

"BioFilter/BioFilter N:N Match" Menu Item

Use "BioFilter/BioFilter N:N Match" menu item to make a N:N matching. N: N Matching compares each image, specified in the search- directory or search-file, with every image in the search-directory or search-file. N:N matching requires (1) the images in the search-directory or search-file being converted into templates in advance; and (2) the BioFilter being trained. The result is in file, b1.txt, which will be opened at the end of computation.

"BioFilter/BioFilter Results" Menu Item

Use "BioFilter/BioFilter Results" menu item to open b1.txt, the file containing the last matching result.

"BioFilter/Check (b1_matchlist.txt required)" Menu Item

Use "BioFilter/Check (b1_matchlist.txt required)" menu item to check the matching results. If this is a test run (i.e., you know the correct answers), you can see the matching results in seconds. To test the results in b1.txt, you must prepare B1_matchlist.txt file, which indicates the matching pairs. Once b1.txt and B1_matchlist.txt are prepared, the command will let you know your matching results in seconds.

"BioFilter/Option" Menu Item

Use "BioFilter/Option" menu item to open a sub-menu, which allows you to store the templates files into b2.txt, b3.txt, b4.txt, and b5.txt.

"BioFilter/Option/Scan Images - Directory Input (a2.txt)" Menu Item

Similar to "BioFilter/Scan Images - Directory Input" menu item but saves the results to a2.txt.

"BioFilter/Option/Scan Images - Directory Input (a3.txt)" Menu Item

Similar to "BioFilter/Scan Images - Directory Input" menu item but saves the results to a3.txt.

"BioFilter/Option/Scan Images - Directory Input (a4.txt)" Menu Item

Similar to "BioFilter/Scan Images - Directory Input" menu item but saves the results to a4.txt.

"BioFilter/Option/Scan Images - Directory Input (a5.txt)" Menu Item

Similar to "BioFilter/Scan Images - Directory Input" menu item but saves the results to a5.txt.

"BioFilter/Option/Scan Images - File Input (a2.txt)" Menu Item

Similar to "BioFilter/Scan Images - File Input" menu item but saves the results to a2.txt.

"BioFilter/Option/Scan Images - File Input (a3.txt)" Menu Item

Similar to "BioFilter/Scan Images - File Input" menu item but saves the results to a3.txt.

"BioFilter/Option/Scan Images - File Input (a4.txt)" Menu Item

Similar to "BioFilter/Scan Images - File Input" menu item but saves the results to a4.txt.

"BioFilter/Option/Scan Images - File Input (a5.txt)" Menu Item

Similar to "BioFilter/Scan Images - File Input" menu item but saves the results to a5.txt.

BioFilter Parameter/Bio-Filter Scale TextBox

Use Bio-Filter Scale textbox to control the amount of output. This parameter ranges from 0 to 100. The larger this number is, the more matches you will get. To set this parameter, enter a number between 0 and 100 to the text box.

BioFilter Parameter/Bio-Filter Mode Button

Use “Bio-Filter Mode” button to determine whether later filters will use this filter. Since BioFilter is one of the recognition filters, you may or may not use this filter. This parameter decides whether the BioFilter will be used.

This parameter has three values:

Untrained
Trained
Bypass

- The “Untrained” setting will do an image matching without training.
- The “Trained” setting requires the BioFilter be trained first.
- The “Bypass” setting will by pass this filter.

To set the parameter, keep clicking the button; the setting will switch from one to the next each time you click this button.

BioFilter Parameter/Output Button

Use “Output” button to decide whether you want to display the score or not. To determine if one image "matches" another image, they must be compared using a unique algorithm. Generally, the result of this comparison is a "score", indicating the degree to which a match exists. This score is then compared to a pre-set threshold to determine whether or not to declare a match.

The Output parameter has two settings:

No Scores
Scores

If the BioFilter is an intermediate step, this score will not show up in the output file. The “No Scores” setting (default setting) will not show the scores in the Output file. If the BioFilter is the only filter used in the matching, then you can show the score in the Output file by selecting the “Scores” setting. To set the parameter, keep clicking the button; the setting will switch from one to the next each time you click the “Blurring” button.

BioFilter Parameter/BioFilter Threshold TextBox

Use Threshold TextBox to set the BioFilter threshold. The result of image comparison is a "score", indicating the degree to which a match exists. This score is then compared to a pre-set Threshold to determine whether or not to declare a match. This parameter sets the threshold.

To decide what threshold to use, you should make a test run first and look at the scores. Matching images have higher scores; unmatched images have lower scores. Select a threshold to separate these two groups. There will be a few images in the middle, representing both groups. Under these circumstances, the threshold selection depends on your applications. To set the Threshold parameter, enter a number into the Threshold text box.

23.5 Neural Filter

Neural Filter Drop Down List

Use the “Neural Filter Drop Down List” to select a Neural Filter. The Neural Filter is the main matching filter for the Feature Space.

“NeuralFilter\Train (match.txt required)” Menu Item

Use “NeuralFilter\Train (match.txt required)” menu item to train the Neural Filter. Training uses the data collected in advance to teach the Neural Filter how to match. Training requires two files, a1.txt and match.txt:

- A1.txt is the record file, which contains many records. Each image is converted into a record. A record represents features of an image in a Feature Space.

- Match.txt is a list of matching pairs. This file will teach the **ImageFinder** who will match with whom.

"NeuralFilter/NeuralFilter 1:N Match" Menu Item

Use "NeuralFilter/NeuralFilter 1:N Match" menu item to make a 1:N Matching. 1:N Matching compares one key image with the images in a search-directory or search-file; the key image is specified in the "Key Segment" textbox or selected by the "Key Segment" button. 1:N Matching requires (1) the images in the search-directory or search-file being converted into templates in advance; and (2) the NeuralFilter being trained. To make an 1:N matching:

- Click "Key Segment" button, and select an image;
- Click "NeuralFilter/NeuralFilter 1:N Match".

The result is in file, b1.txt, which will be opened at the end of computation.

"NeuralFilter/NeuralFilter N:N Match" Menu Item, and
"NeuralFilter/Query Set + Target Set/a1 + a1 ==> b1" Menu Item

Use "NeuralFilter/NeuralFilter N:N Match" menu item to make an Untrained N:N Matching. N:N Matching compares each image, specified in the search-directory or search-file, with every image in the search-directory or search-file. N:N Matching requires (1) the images in the search-directory or search-file being converted into templates in advance; and (2) the NeuralFilter being trained. The result is in file, b1.txt, which will be opened at the end of computation.

"NeuralFilter/NeuralFilter N:(N-1) Match" Menu Item

Use "NeuralFilter/NeuralFilter N:(N-1) Match" menu item to make an N: (N-1) Matching. Let a and b be two images; N:N Matching is {aa, ab, ba, bb} and the N : (N-1) Matching is {ab}. The N: N Matching has N * N comparisons; and the N : (N-1) Matching has N * (N-1)/2 comparisons. The purpose of N : (N-1) Matching is to reduce the number of comparisons. N:(N-1) Matching requires (1) the images in the search-directory or search- file being converted into templates in advance; and (2) the NeuralFilter being trained. The result is in file, b1.txt, which will be opened at the end of computation.

"NeuralFilter/NeuralFilter Results" Menu Item

Use "NeuralFilter/NeuralFilter Results" menu item to open b1.txt, the file containing the last matching result.

"NeuralFilter/Check (b1_matchlist.txt required)" Menu Item

Use "NeuralFilter/Check (b1_matchlist.txt required)" menu item to check the matching results. If this is a test run (i.e., you know the correct answers), you can see the matching results in seconds. To test the results in b1.txt, you must prepare B1_matchlist.txt file, which indicates the matching pairs. Once b1.txt and B1_matchlist.txt are prepared, the command will let you know your matching results in seconds.

"NeuralFilter/Query Set + Target Set" Menu Item

Use "NeuralFilter/Query Set + Target Set" menu item to open a sub-menu. These menu items allow you to compare images in file with images in another file.

"NeuralFilter/ Query Set + Target Set/a1 + a2 ==> b2" Menu Item

Similar to "NeuralFilter/NeuralFilter N:N Match" menu item, except that "NeuralFilter/NeuralFilter N:N Match" menu item makes the following matching: a1 + a1 ==> b1 (images in a1.txt matches against images in a1.txt and the results are in b1.txt), this command makes the following match: a1 + a2 ==> b2 (images in a1.txt matches against images in a2.txt and the results are in b2.txt).

"NeuralFilter/ Query Set + Target Set/a1 + a3 ==> b3" Menu Item

Similar to "NeuralFilter/NeuralFilter N:N Match" menu item, except that "NeuralFilter/NeuralFilter N:N Match" menu item makes the following matching: a1 + a1 ==> b1 (images in a1.txt matches against images in a1.txt and the results are in b1.txt), this command makes the following match: a1 + a3 ==> b3 (images in a1.txt matches against images in a3.txt and the results are in b3.txt).

"NeuralFilter/ Query Set + Target Set/a1 + a4 ==> b4" Menu Item

Similar to "NeuralFilter/NeuralFilter N:N Match" menu item, except that "NeuralFilter/NeuralFilter N:N Match" menu item makes the following matching: a1 + a1 ==> b1(images in a1.txt matches against images in a1.txt and the results are in b1.txt), this command makes the following match: a1 + a4 ==> b4 (images in a1.txt matches against images in a4.txt and the results are in b4.txt).

"NeuralFilter/ Query Set + Target Set/a1 + a5 ==> b5" Menu Item

Similar to "NeuralFilter/NeuralFilter N:N Match" menu item, except that "NeuralFilter/NeuralFilter N:N Match" menu item makes the following matching: a1 + a1 ==> b1(images in a1.txt matches against images in a1.txt and the results are in b1.txt), this command makes the following match: a1 + a5 ==> b5 (images in a1.txt matches against images in a5.txt and the results are in b5.txt).

NeuralFilter Parameter/Neural-Filter Scale TextBox

Use Neural-Filter Scale textbox to control the amount of output. This parameter ranges from 0 to 100. The larger this number is, the more matches you will get. To set this parameter, enter a number between 0 and 100 to the text box.

NeuralFilter Parameter/Neural-Filter Mode Button

Use the "Neural-Filter Mode" button to determine whether this filter will be used by later filters. Since NeuralFilter is one of the recognition filters, you may or may not use this filter. This parameter decides whether the NeuralFilter will be used.

This parameter has two values:

Trained
Bypass

- The "Trained" setting requires the NeuralFilter being used by later filters.
- The "Bypass" setting will by-pass this filter.

To set the parameter, keep clicking the button; the setting will switch from one to the next each time you click this button.

NeuralFilter Parameter/Neural-Filter Opening Button

Use the "Neural-Filter Opening" button to control the "Openness" of this filter. This parameter has the following settings:

- Very Large
- Large
- Normal
- Small
- Very Small

A large opening will allow more matches in the results.

NeuralFilter Parameter/Output Button

Use the "Output" button to decide whether you want to display the score or not. To determine if one image "matches" another image, they must be compared using a unique algorithm. Generally, the result of this comparison is a "score", indicating the degree to which a match exists. This score is then compared to a pre-set threshold to determine whether or not to declare a match.

The Output parameter has two settings:

No Scores
Scores

If the NeuralFilter is an intermediate step, this score will not show up in the output file. The "No Scores" setting (default setting) will not show the scores in the output file. If the NeuralFilter is the only filter used in the matching, then you can show the score in the output file by selecting the "Scores" setting. To set the parameter, keep clicking the button; the setting will switch from one to the next each time you click the "Blurring" button.

NeuralFilter Parameter/NeuralFilter Threshold TextBox

Use Threshold TextBox to set the NeuralFilter threshold. The result of image comparison is a "score", indicating the degree to which a match exists. This score is then compared to a pre-set Threshold to determine whether or not to declare a match. This parameter sets the threshold.

To decide what threshold to use, you should make a test run first and look at the scores. Matching images have higher scores; unmatched images have lower scores. Select a threshold to separate these two groups. There will be a few images in the middle, representing both groups. Under these circumstances, the threshold selection depends on your application. To set the Threshold parameter, enter a number into the Threshold text box.

23.6 NeuralNet Filter

Neural Net Filter Drop Down List

Use the "Neural Net Filter Drop Down List" to set the Neural Net Filter. The default filter is 100x100. All images are scaled down by an integer amount. For example, 640x480 will be scaled down 7 times to 91x68. You need to understand this; so when you translate, or rotate an image, you will make sure no additional scaling factors are introduced.

The search speed crucially depends on the Neural Net Filter. For example, if the 50x50 filter is used, then the underlying neural net size is reduced by a factor of 4; and the neural computation speed will be increased by a factor of 16.

The filters available are:

- 100x100 (Most Accurate)
- 90x90
- 80x80
- 70x70
- 60x60
- 50x50 (Least Accurate)

Let the speed of 100x100 representation be a base, then the overall speed for:

- 90x90 representation is 1 times faster;
- 80x80 representation is 1.6 times faster;
- 70x70 representation is 2.7 times faster;
- 60x60 representation is 5 times faster; and
- 50x50 representation is 10 times faster.

NeuralNet/NeuralNet Train Menu Item

Use the "NeuralNet Train" menu item to train the software to learn the active key (what image to look for). These commands will first delete all old training and start to train the software from the beginning. After clicking a button, wait 1 second. If everything is O.K., a message "Training End!" will be printed in the Status Text Area.

NeuralNet/NeuralNet Retrain Menu Item

Use the "Retrain" menu item to retrain the software to learn additional keys. If several keys are used for training, the first learning uses **training** and all the subsequent learning uses **retraining**. After clicking the button, wait 1 second. If everything is O.K., a message "Retraining End!" will be printed in the Status Text Area.

NeuralNet/1:N Search Menu Item

Use the "1:N Search" menu item to retrieve images in the search-directory. You must train the software first before searching. After clicking the button, wait until a web page is opened. If everything is O.K., a message "Retrieval End!" will be printed in the text area, then a web page with a list of retrieved images will be opened. Whenever you click a file name in the opened web page, the retrieved image will be shown. You might need to click the "Refresh" button.

Note that, next to each image, an integer, called **similarity**, is printed. The larger the number is, the more similarity between the training image(s) and the retrieved images.

NeuralNet/N:N Search Menu Item

Use the "N:N Search" menu item to match each image in the search-directory against all other images in the directory. After clicking the button, wait until a web page is opened. If everything is O.K., a message "Retrieval End!" will be printed in the text area, then a web page with a list of retrieved images will be opened. Whenever you click a file name in the opened web page, the retrieved image will be shown. You might need to click the "Refresh" button. In the output file, each image in the search-directory has a block; the first line in a block is the input and the rest of the lines in a block is the output.

Note that, next to each image, an integer, called **similarity**, is printed. The larger the number is, the more similarity between the training image(s) and the retrieved images.

NeuralNet/Sort Menu Item

Use the "Sort" menu item to sort the retrieved 1:N Matching results according to their weights. You might need to click the "Refresh" button. For N:N Matching, only the last block is sorted.

NeuralNet/Long-Search Directory Menu Item

Use the “Long-Search” Directory menu item to specify a search-directory. To select a search-directory, go to the directory; **then click any file in the directory.** The first image in the search-directory will be shown in the **ImageFinder** to indicate the specification is successful and the search-directory is active for retrieval.

NeuralNet/1:N Long-Search Menu Item

Use the "1:N Long-Search" menu item to retrieve images in the sub-directories of the search-directory. You must train the software first before searching. After clicking the button, wait until a web page is opened. If everything is O.K., a message "Retrieval End!" will be printed in the text area, then a web page with a list of retrieved images will be opened. Whenever you click a file name in the opened web page, the retrieved image will be shown. You might need to click the "Refresh" button.

Note that, next to each image, an integer, called **similarity**, is printed. The larger the number is, the more similarity between the training image(s) and the retrieved images.

NeuralNet/Long-Sort Menu Item

Use the "Long-Sort" menu item to sort the retrieved Long-Search results according to their weights. You might need to click the "Refresh" button.

NeuralNet/1:N File-Search Menu Item

Use the "1:N File-Search" menu item to retrieve images in the search-file. You must train the software first before searching. After clicking the button, wait until a web page is opened. If everything is O.K., a message "Retrieval End!" will be printed in the text area, then a web page with a list of retrieved images will be opened. Whenever you click a file name in the opened web page, the retrieved image will be shown. You might need to click the "Refresh" button.

Note that, next to each image, an integer, called **similarity**, is printed. The larger the number is, the more similarity between the training image(s) and the retrieved images.

NeuralNet/N:N File-Search Menu Item

Use the "N:N File-Search" menu item to match each image in the search-file against all other images in the file. After clicking the button, wait until a web page is opened. If everything is O.K., a message "Retrieval End!" will be printed in the text area, then a web page with a list of retrieved images will be opened. In the output file, each image in the search-file has a block; the first line in a block is the input and the rest of the lines in a block is the output.

Note that, next to each image, an integer, called **similarity**, is printed. The larger the number is, the more similarity between the training image(s) and the retrieved images.

NeuralNet/File Sort Menu Item

Use the "File Sort" menu item to sort the retrieved 1:N File Matching results according to their weights. You might need to click the "Refresh" button. For N:N Matching, only the last block is sorted.

NeuralNet/NeuralNet Results Menu Item

Use "NeuralNet Results" menu item to open the last matching result.

NeuralNet/NeuralNet Advisor Menu Item

Use the "NeuralNet Advisor" menu item to get the values of the following parameters: Segment-Cut, Blurring, Internal Weight Cut, and Sensitivity. The **ImageFinder** has a 'Parameter Advisor" to help you to locate the general range of these parameters. To use the Advisor, you must already know how to search.

Procedure:

1. Assume a sample image is sample.jpg. Put the sample image and all matched images in a directory, say, c:\image\.
2. Enter the sample image, sample.jpg, and the search-directory, c:\image\ into the **ImageFinder**.
3. Set the parameters rather large to make sure all the matched images are retrieved by the **ImageFinder,** say:
   Segment Cut = 0
   Blurring = 50
   Shape Cut = 90
   Internal Weight Cut = 90
   Sensitivity = 90
4. Click the Advisor Menu Item.
5. Click the "Get Parameter" button in the Advisor window; you will see the recommended parameter settings.

The recommended values represent an average, so it might not fit some of your images, but it gives you a basic idea of the general intervals of these parameters.

NeuralNet/Parameter/Symmetry

Use the "Symmetry" button to set the symmetry. The symmetry settings are:

- No symmetry;
- Translation symmetry;
- Rotation symmetry;

- Scaling symmetry; and
- Rotation and Scaling symmetry.

The default setting is "Tran", the Translation symmetry. To set the symmetry, keep clicking the button; the setting will switch from one to the next each time you click the button.

NeuralNet/Parameter/Translation Type Button

Use "Translation Type" button to select the accuracy of the translation symmetry. The Translation Type settings (and their codes) are:

- Most Accurate (0);
- Accurate (1); and
- Least (2).

To set the Translation Type, keep clicking the "T Type" button; the setting will switch from one to the next each time you click the button. The default setting is 0, the most accurate setting.

NeuralNet/Parameter/Scaling Type Button

Use "Scaling Type" button to select the accuracy of the scaling symmetry. The Scaling Type settings (and their codes) are:

- Least Accurate (0);
- Accurate (1);
- Accurate (2); and
- Most Accurate (3).

To set the Scaling Type, keep clicking the "S Type" button; the setting will switch from one to the next each time you click the button. The default setting is 0, the least accurate setting.

NeuralNet/Parameter/Rotation Type Button

Use the R Type (Rotation Type) buttons to set the Rotation Types. The settings are:

- 360° rotation (0);
- -5° to 5° rotation (1);
- -10° to 10° rotation (2);
- 360° rotation, accurate (3);
- 360° rotation, more accurate (4);
- 360° rotation, most accurate (5).

Other settings can be ordered in a Customized Version.

To set the Rotation Type, keep clicking "R Type" button; the setting will switch from one to the next each time you click the button. The default setting is 360° rotation (0).

NeuralNet/Parameter/Training Segment Text Boxes

Use Training Segment Text Boxes to set the training segment. To enter a segment, first select an image. The selected image will be shown in the **ImageFinder**, 300 by 300 in size, to indicate the specification is successful. The key-segment is specified by 4 integers: the upper-left corner (x, y) and the length and height (w, h) of the segment. Once the segment specification is successful, a black box will cover the selected area. If the selected area is not what you want, just re-select the area again.

NeuralNet/Parameter/Blurring Text Box

Use Blurring Text Box to control the amount of output. "0%"-Blurring means the exact match. When the "Blurring" is increased, you will get more and more similar images. As the Blurring goes higher, the speed will be slower. The Blurring settings range from 0 – 50. To set the Blurring, enter a number between 0 and 50 to the text box. The default setting is 5%.

NeuralNet/Parameter/Sensitivity Text Box

Use Sensitivity Text Box to adjust search segment size. The Sensitivity parameter

ranges from 0 (least sensitive) to 100 (most sensitive).

- To search small segment(s), use a high sensitivity search.
- To search large segment(s), use low sensitivity search.
- The higher the parameter is set, the more results you will get.

To set the Sensitivity, enter a number between 0 and 100 the text box. The default setting is 50.

NeuralNet/Parameter/External Weight Cut (ExternalCut) Text Box

Use the External Cut Text Box to eliminate those retrieved images with the weights below the External Cut. This parameter is also called Threshold. To set the External Cut, enter a number to the text box. The default setting is 0. In general, it is better to give no answer than a wrong answer. Assume you are searching images and all similar images have weights ranging from 1,000 to 10,000. It is possible that some other images will pop up with weights ranging from 10 to 100. To eliminate these images, you can set the External Cut to 1,000.

NeuralNet/Parameter/Internal Weight Cut (Internal Cut) Text Box

The Internal Cut plays the similar role as the External Cut. There are two differences between these two cuts:

- The Internal Cut ranges from 0 to 100; and the ExternalCut can be any number;
- The Internal Cut stops the images from coming out; whereas, the External Cut can bring the eliminated images back if you set the External Cut to 0. You might need to see the eliminated images sometimes for the purpose of adjusting parameters.

To set the Internal Cut, enter a number between 0 and 100 to the text box. The default setting is 100.

NeuralNet/Parameter/Segment Size Button

Use the “Segment Size” button to select segment size. The default setting is "L Segment".

- To search large segments, use "L Segment" (Large Segment). For example, if a sample segment is one quarter of the sample image, it is a large segment
- To search small segments, use "S Segment" (Small Segment). If the segment is 1/25 of the sample image, it is a small segment.

To set the segment size, keep clicking the Size button; the setting will switch from one to the next each time you click the button.

Currently, "S Segment" only supports translation symmetry. If you need rotation and scaling symmetry, please use "L Segment". Additional symmetry can be added very quickly in a Customized Version.

NeuralNet/Parameter/Image Type Button

There are BW and Color images. For each of them, there are “sum-search”, “maximum-search”, and “average-search”. This generates 6 image types:

- Bi-level 1 (0)
- Bi-level 2 (1)
- Bi-level 3 (2)
- Color 1 (3)
- Color 2 (4)
- Color 3 (5)

"Bi-level 1” is like an integration of a function f(x); "Bi-level 2” is like a maximum value of f(x); and "Bi-level 3” is the average of the above two.

"Bi-level 1" search will produce a higher weight than a "Bi-level 2" search. "Bi-level 3" search is in the middle. Similarly, a "Color 1" search

will produce a higher weight than a "Color 2" search. "Color 3" is in the middle.

To set the image type, keep clicking the "Image Type" button; the setting will switch from one to the next each time you click the "Image Type" button.

NeuralNet/Parameter/File Display Type Button

Use "File Display Type" button to set the output file type. The options are text file and html file.

NeuralNet/Parameter/AutoSegment Button

Use "AutoSegment" button to select a training segment automatically. The options are:

Manual Segment
AutoSegment 10
AutoSegment 20
AutoSegment 30

"Manual Segment" setting requires you to select a training segment using the 4 Training Segment Text Boxes introduced earlier in this section. To select a segment by computer, do not use "Manual Segment" setting. To set this parameter, keep clicking the button; the setting will switch from one to the next each time you click the button.

The "AutoSegment 10" setting will select a larger segment than "AutoSegment 20" setting, which in turn, will select a larger segment than the "AutoSegment 30" setting.

NeuralNet/Parameter/Mode Button

Use Neural Net Mode button to determine whether this filter will be used by later filters. Since the Neural Net filter is one of the recognition filters, you may or may not use this filter. This parameter decides whether the BioFilter will be used.

This parameter has two values:

Trained
Bypass

The "Trained" setting requires that later filters use the NeuralNet filter. The "Bypass" setting will by-pass this filter.

To set the parameter, keep clicking the button; the setting will switch from one to the next each time you click this button.

23.7 Batch Commands

Batch/Set Execution Code Menu Item

Use "Set Execution Code" menu item to set the Set Execution Code. There are many commands in the **ImageFinder.** Each command has an integer for identification. This integer is called Batch Execution Code. This number is used before you save your batch code.

Batch/Save Menu Item

Use the "Save" menu item to save the last **ImageFinder** setting in batch code. The batch code is saved to a file, abm60.txt.

Batch/Save 2 Menu Item

Use the "Save 2" menu item to save the last **ImageFinder** setting in batch code. The batch code is saved to a file, abm60_2.txt.

Batch/Save 3 Menu Item

Use the "Save 3" menu item to save the last **ImageFinder** setting in batch code. The batch code is saved to a file, abm60_3.txt.

Batch/Save 4 Menu Item

Use the "Save 4" menu item to save the last **ImageFinder** setting in batch code. The batch code is saved to a file, abm60_4.txt.

Batch/Save 5 Menu Item

Use the "Save 5" menu item to save the last **ImageFinder** setting in batch code. The batch code is saved to a file, abm60_5.txt.

Batch/Open Menu Item

Use the "Open" menu item to open the batch code file saved by the Batch/Save command.

Batch/Open 2 Menu Item

Use the "Open 2" menu item to open the batch code file saved by the "Batch/Save 2" command.

Batch/Open 3 Menu Item

Use the "Open 3" menu item to open the batch code file saved by the "Batch/Save 4" command.

Batch/Open 4 Menu Item

Use the "Open 4" menu item to open the batch code file saved by the "Batch/Save 4" command.

Batch/Open 5 Menu Item

Use the "Open 5" menu item to open the batch code file saved by the "Batch/Save 5" command.

Batch/Notes

Click the "Notes" menu item to create an online note so you can remember which batch code is for which problem.

Batch/Run

Use the "Run" menu item to execute the batch code in the display area.

Batch/Load

Use the "Load" menu item to load the parameters specified by the batch code in the display area. The load command will not execute the batch code.

23.8 BioFilter II

BioFilter II Drop Down List

Use the "BioFilter II Drop Down List" to select a BioFilter-II. BioFilter I matches the whole image, while BioFilter-II matches a part of an image. The main role of the BioFilter-II is to eliminate 80% of the mismatches.

"BioFilter 2/Scan Images - Directory Input" Menu Item

Use "BioFilter 2/Scan Images - Directory Input" menu item to convert images into records. The BioFilter-II is one of two filters operating on image segments in feature space image recognition (Image recognition is divided into feature space recognition and input space recognition. The feature space recognition operates on signatures of images.) To convert images into templates:

- Click "Search Dir" button to specify the search-directory.
- Click menu item "BioFilter 2/Scan Images - Directory Input" to convert images to records. You should see the **ImageFinder** scan through the images at this point.

"BioFilter 2/Scan Images - File Input" Menu Item

Use "BioFilter 2/Scan Images - File Input" menu item to convert images into records. The BioFilter II is one of two filters operating on image segments in feature space images recognition (Image recognition is divided into feature space recognition and input space recognition. The feature space recognition operates on signatures of images.) To convert images into templates:

- Click "Input File" button to specify the search-file.
- Click menu item "BioFilter 2/Scan Images - File Input" to convert images to records. You should see the **ImageFinder** scan through the images at this point.

"BioFilter 2\Train (match2.txt required)" Menu Item

Use "BioFilter 2\Train (match2.txt required)" menu item to train the BioFilter II. Training uses the data collected in advance to teach the BioFilter II how to match. Training requires two files, d1.txt and match2.txt:

- D1.txt is the record file, which contains many records. Each image is converted into a set of records. A record represents features of an image segment in a feature space.
- Match2.txt is a list of matching pairs. This file will teach the **ImageFinder** who will match with whom.

"BioFilter 2/1:N Match (First vs. Rest)" Menu Item

Use "BioFilter2/1:N Match (First vs. Rest)" menu item to make a 1:N matching. 1:N Matching compares one key image with the images in a search-directory or search-file; the key image is specified in the "Key Segment" textbox or selected by the "Key Segment" button. 1:N matching requires (1) the images in the search- directory or search-file being converted into templates in advance; and (2) the BioFilter II being trained. To make an 1:N matching:

- Click the "Key Segment" button, and select an image;
- Click "BioFilter 2/1:N Match (First vs. Rest)".

The result is in file, e1.txt, which will be opened at the end of computation.

"BioFilter/N:N Match" Menu Item

Use "BioFilter/N:N Match" menu item to make an N:N matching. N: N compares each image, specified in the search-directory or search-file, with every image in the search-directory or search-file. N:N matching requires (1) the images in the search-directory or search-file being converted into templates in advance; and (2) the BioFilter II being trained. The result is in file, e1.txt, which will be opened at the end of computation.

"BioFilter 2/BioFilter-2 Results" Menu Item

Use "BioFilter 2/BioFilter-2 Results" menu item to open e1.txt, the file containing the last matching result.

Parameters

Please see the BioFilter section.

23.9 Neural Filter II

Neural Filter II Drop Down List

Use the "Neural Filter II Drop Down List" to select a Neural Filter-II. The Neural Filter-II is the main matching filter operating on image segments for the Feature Space.

"NeuralFilter 2\Train (match2.txt required)" Menu Item

Use "NeuralFilter 2\Train (match2.txt required)" menu item to train the Neural Filter. Training uses the data collected in advance to teach the Neural Filter how to match. Training requires two files, d1.txt and match2.txt:

- D1.txt is the record file, which contains many records. Each image is converted into a set of records. A record represents features of an image segment in a feature space.
- Match2.txt is a list of matching pairs. This file will teach the **ImageFinder** who will match with whom.

"NeuralFilter 2/1:N Match (First vs. Rest)" Menu Item

Use "NeuralFilter 2/1:N Match (First vs. Rest)" menu item to make a 1:N matching. 1:N Matching compares one key image with the images in a search-directory or search-file; the key image is

specified in the "Key Segment" textbox or selected by the "Key Segment" button. 1:N matching requires (1) the images in the search- directory or search-file being converted into templates in advance; and (2) the NeuralFilter II being trained. To make an 1:N matching:

- Click "Key Segment" button, and select an image;
- Click "NeuralFilter 2/1:N Match (First vs. Rest)".

The result is in file, e1.txt, which will be opened at the end of computation.

"NeuralFilter 2/ N:N Match" Menu Item

Use "NeuralFilter 2/N:N Match" menu item to make an untrained N:N matching. N:N compares each image, specified in the search- directory or search-file, with every image in the search-directory or search-file. N:N matching requires (1) the images in the search- directory or search-file being converted into templates in advance; and (2) the NeuralFilter II being trained. The result is in file, e1.txt, which will be opened at the end of computation.

"NeuralFilter 2/NeuralFilter-2 Results" Menu Item

Use "NeuralFilter 2/NeuralFilter-2 Results" menu item to open e1.txt, the file containing the last matching result.

Parameters

Please see the Neural Filter section.

23.10 NeuralNet II Filter

Neural Net II Filter Drop Down List

Use the "Neural Net II Filter Drop Down List" to set the Neural Net II Filter. The default filter is 100x100. All images are scaled down by an integer amount. For example, 640x480 will be scaled down 7 times to 91x68.

- You need to understand it; so when you translate, or rotate an image, you will make sure no additional scaling factors are introduced.
- **The search speed crucially depends on this filter.** For example, if the 50x50 filter is used, then the underlying neural net size is reduced by a factor of 4; and the neural computation speed will be increased by a factor of 16.

The filters available are:

- 100x100 (Most Accurate)
- 90x90
- 80x80
- 70x70
- 60x60
- 50x50 (Least Accurate)

Let the speed of 100x100 representation be a base, then the overall speed for:

- 90x90 representation is 1 times faster;
- 80x80 representation is 1.6 times faster;
- 70x70 representation is 2.7 times faster;
- 60x60 representation is 5 times faster; and
- 50x50 representation is 10 times faster.

NeuralNet 2/1:N Match (First vs. Rest) Menu Item

Use the "1:N Match (First vs. Rest)" menu item to retrieve images in the search-file. This command will train the software using the first image in the input file. After clicking the button, wait until a result file is opened.

NeuralNet 2/NeuralNet-2 Results Menu Item

Use "NeuralNet-2 Results" menu item to open the last matching result.

Parameters

Please see the Neural Net Filter section.

23.11 Segment-Locator

SegLocator/Train Menu Item

Use "Train" menu item to train the **Segment-Locator.**

SegLocator/1:N Match Menu Item

Use "1:N Match" menu item to make a 1:N match for the **Segment-Locator** using directory input.

SegLocator/Batch Run Menu Item

Use "Batch Run" menu item to make a batch run based on the batch code in the text area. **The only valid Execution Code is 1027,** therefore, you must use 1027, meaning 1:N match using the Neural Net Filter and directory input.

23.12 Other Objects In the Page

Hints Button, Help/Hint Menu Item

Use the "Hints" button or "Help/Help" menu item to see the basic procedure.

Help Button, Help/Help Menu Item

Use the "Help" button or "Help/Help" menu item to display the html version of this document.

Clear Button, Help/Clear Window Menu Item

Use the "Clear" button or "Help/Clear Window" menu item to clear the text area.

Home Button, Help/Home Menu Item

Go to http://attrasoft.com

Image Display Area

Use the "Image Area" to see the image(s). This area is 300 by 300 in size. If an image is chosen for specifying a segment, the image will be displayed in this area.

Text Area

Use the "Text Area" to see the current status of your execution, results, error messages, and suggestions.

Help/About Menu Items

Display email, web site, and version number.

23.13 Examples

Example/BioFilter

Use this menu item to open batch codes for the following examples:

N:N Match, Untrained, Label Template
N:N Match, Trained, Label Template
1:N Match, Untrained, Label Template
1:N Match, Trained, Label Template
1:N Match, Trained, Label File
N:N Match, Trained, Label File
1:N Match, Trained, Label Directory
N:N Match, Trained, Label Directory

Example/NeuralFilter

Use this menu item to open batch codes for the following examples:

N:N Match, Label Template
N:N-1 Match, Label Template
1:N Match, Label Template
N:N Match, Label File
N:N-1 Match, Label File
1:N Match, Label File
N:N Match, Label Directory
N:N-1 Match, Label Directory
1:N Match, Label Directory
N:N Match, Logo Template

Example/Neural Net

Use this menu item to open batch codes for the following examples:

1:N Match, Logo, File
N:N Match, Logo, File
United Way - R

Tabasco - R
Mr. Potato - S
Monopoly - S
Compound - RS
Stamp 1
Stamp 2
Long-Search
License Plates
Fingerprint
Fingerprint - 1
Fingerprint - 2
Fingerprint - 34
Fingerprint - 7
Fingerprint - 569

Example/Segment Locator

Use this menu item to open batch codes for the following examples:

United Way
Monopoly
Mr. Potato
AAA
Ford
Soup
PointLocator: Feret 20
PointLocator: Feret 100
PointLocator: Results

Example/BioFilter 2

Use this menu item to open batch codes for the following examples:

Label Match Setting
Label Example 1 (1:N)
Label Example 2 (N:N)
Label Example 3 (1:N)

Example/NeuralFilter 2

Use this menu item to open batch codes for the following examples:

Setting
Example 1 (1:N)
Example 2 (N:N)
Example 3 (1:N)

Example/Neural Net 2

Use this menu item to open batch codes for the following examples:

Match 1
Match 2
Match 3
No Match 1
No Match 2
No Match 3

24. ImageFinder Support Service Packages

Attrasoft's **ImageFinder Services** are designed to accelerate a company's path to deploy Image Recognition Solutions. Our structured Service Offerings help you to develop your products/services with the **ImageFinder** as a component.

24.1 What is Support Service?

ImageFinder Support Service Packages are designed to help a company to understand the process of integrating Image Recognition technology into their product/service. From a large set of possible paths, a company must plan the most realistic execution to ensure the success of the project.

The focus of Support Service is to follow the right development path and shorten the learning curve for developers. Based on dozens of past projects, Attrasoft's development experience will specify the required work and map the shortest path to the system integration; most importantly, Attrasoft Services might prevent you from following a wrong path; thus save you from unnecessary waste of resources, even failure of the project.

System Integration with the **ImageFinder** as a component has two basic approaches:

- **ImageFinder for Dos** (Dos Command Line version of the **ImageFinder**)
- Attrasoft **TransApplet** (Library)

The **ImageFinder** Support Service is divided into several levels:

Level	Cost	#Images
0	NA	NA
1	$1,000	100
2	$5,000	1,000
3	$10,000	1,000
4	$20,000	1,000
5	TBA	1,000

24.2 What is a Feasibility Study?

Attrasoft Image Recognition technology offers a suite of products (Off-the-Shelf **ImageFinder**, **ImageFinder for Dos, TransApplet** (Library version of the **ImageFinder**), **Customized-ImageFinder, Customized-Dos Version**, **Customized-TransApplet**) that enables the success of projects a company will develop with Attrasoft components.

A Feasibility Study is very valuable because experience from many past projects will be naturally deployed into your project. The very first question we encountered, and you will be asked when you justify your project to your company, is 'Can this be done?' Very often, the answer is more than a simple "Yes"; a feasibility study will be necessary to respond this question, which is further divided into a Preliminary Assessment with 100 images and a Feasibility Study with 1,000 images.

24.3 ImageFinder Services

With **ImageFinder** Level 0 Service (that comes with the purchase of the **ImageFinder**), you can:

- Get Help for Installing and Running the software, such as how to obtain Microsoft .Net Framework and "J# .Net Redistributable".
- Get Help on how to operate the software by running examples in the **ImageFinder** and trying some of your own examples.

- Evaluate the **ImageFinder** and justify a product/service development opportunity to your organization based on the Preliminary Evaluation, if such a project is not already in place. The Preliminary Evaluation is usually based on a few of your own images.
- Select a Service Package based on your need.

With **ImageFinder** Level 1 Service ($1,000), you can:

- Set an implementation roadmap based on dozens of past Attrasoft projects. Attrasoft's development experience will specify the required work and map the shortest path to the System Integration; most importantly, Attrasoft Services might prevent you from following a wrong path.
- Develop a Miniature Proof-of-Concept Demo using the **ImageFinder** for your internal justification purposes based on 100 images. The 100 images should result in 10,000 comparisons in an N: N Matching; thus promote confidence and gather internal support from your company.
- Define necessary performance requirements as defined by several variables in the **ImageFinder**.

With **ImageFinder** Level 2 Service ($5,000), in addition to Level 1 Service, you can:

- Get Help to finalize an implementation road map as more information is gathered;
- Develop a Proof-of-Concept (Feasibility) Demo using the **ImageFinder**, which will specify all of the parameters to be used in the final product/services. 1,000 images will be used in the Feasibility Study, resulting in 1,000,000 comparisons in an N: N Matching.

With **ImageFinder** Level 3 Service ($10,000 per year), in addition to Level 2 Service, you can:

- Get **Consultation Help** from Attrasoft **throughout your entire Development Phase**, from device setup, data collection, layered approach, **up to the System Integration,** ensuring smooth development.

In all of the above cases, the Off-the-Shelf **ImageFinder** is used.

However, from time to time because projects using Image Recognition are Pioneer Initiatives, it is sometimes necessary to make minor modifications to the **ImageFinder**; sometimes, a major effort will be necessary for the Proof-of-Concept Study.

With **ImageFinder** Level 4 Service ($20,000), in addition to Level 3 Service, you can:

- Get a **Customized ImageFinder** with Minor Programming Efforts (<500 non-recurring engineering hours) for one or more of the following reasons:

- Reducing the Operation Complexity via Attrasoft tuning the Parameters to one specific image type;
- Speed Optimization;
- Internal Structure Optimization;
- Data Input Optimization via preprocessed data rather raw data;
- Graphical User Interface Customization;
- Neural Network Module and structure Customization;
- Memory Optimization (For some problems, the RAM consumption can be reduced by 80%);
- Database Interface;
- New Image Preprocessing Filters;
- Programming Library;
- Specific Symmetries or Combination of Symmetries;
- Fine Tuning of the Neural Parameters;
- Digital Image Database (Combine **ImageFinder** with Database);

- Image Formats other than jpg and gif;
- Internet Image Search Engines;
- Multi-Layers of Image Matching;
- Web Interface (solutions that will provide users with a searchable database using a Web Interface);
- Other Specific Needs.

With **ImageFinder** Level 5 Service (TBA), in addition to Level 4 Service, you can:

- Get a **Customized ImageFinder** that requires Major Programming Efforts.

Chapter 25. Readme.txt

25.1 Software Requirement

Software Requirements:

(1) Windows .Net Framework 1.1.
(2) J# .Net Redistributable.
(3) Internet Explorer.

(1) To get the latest version .Net Framework 1.1, use the Internet Explorer, then click "Tools\Windows Update".

(2) To get J# .Net Redistributable, either get it from this Microsoft site directly: http://www.microsoft.com/downloads/details.aspx?familyid=E3CF70A9-84CA-4FEA-9E7D-7D674D2C7CA1&displaylang=en or get it from the CD with the following path: CD:\vjredist.exe. Please install it by double clicking it.

25.2 Install the Software

1. If you have not done so, go to Internet Explorer, then click "Tools\Windows Update" to download Windows .Net Framework 1.1.

2. Click CD:\vjredist.exe to install Microsoft J# .Net Redistributable.

3.Click CD:\setup.exe to install the **ImageFinder**.

25.3 Image Recognition

The **ImageFinder** is an Image Recognition software. At a minimum, you will need the following steps:

- Image Processing
- Feature Space Recognition
- Input Space Recognition (Pixel Recognition)

These steps can be combined into a single batch click.

Image Processing sets up 3 image filters: Edge Filters, Threshold Filters, and Clean Up Filters.

Feature Space Recognition trains two filters, the BioFilter and NeuralFilter, and uses these filters for 1:N and N:N Image Matching.

Input Space Recognition trains the Neural Net Filter and makes 1:N or N:N Matching at the pixel level. The output of the Feature Space Recognition is usually the input of the Neural Net Filter.

25.3.1 Image Processing

If you are not familiar with Image Processing, please try the following:

Setting 1:

> Edge Filters: Sobel 1 (or Sobel 2)
> Threshold Filters: Dark Background 128

Setting 2:

> Edge Filters: None ("Enter An Edge Filter")
> Threshold Filters: Light Background 128

Other settings can be adopted by trial and error. You should choose an image pre-processing procedure **where the sample objects stand out, otherwise change the options**.

25.3.2 BioFilter Templates

BioFilter matching will have a few steps:

- Initialization
- Converting Images to Records

- Training
- Template Matching
- Results.

Initialization sets the **ImageFinder** parameters. To start with, simply use the default setting.

To converting Images to Records in a feature space, there are two steps:

- Entering Data: click "Search Dir" and select any file in the input directory;
- Records: click menu item "BioFilter/Scan Images - Directory Input".

Similar steps can be performed for file input.

25.3.3 BioFilter Training and Matching

The main task of training is to prepare a file match.txt, which is a list of matching pairs. Click "BioFilter\Train (match.txt required)" to train the BioFilter.

For BioFilter N:N Matching: click "BioFilter/BioFilter N:N Match (Trained)". For BioFilter 1:N Matching: click "BioFilter/BioFilter 1:N Match (Trained)".

The result file contains many blocks: each image has a block. Line 1 in each block is the input and the rest of the lines are output.

25.3.4 Neural Filters

The Neural Filter runs parallel to the BioFilter.

25.3.5 NeuralNet Filter

Neural Network matching will have a few steps:

- Initialization
- Training
- Matching

Initialization sets the **ImageFinder** parameters. To start with, simply use the default setting. Training and matching take the following steps:

1. Enter key-segments into the **ImageFinder** (keys are used to teach this software what to look for);
2. Click the NeuralNet/Train to teach the software what to look for.
3. Save all the images you want to look through into a directory (search-directory) and enter it into the software;
4. Click a NeuralNet/Search --- the computer is now looking through the images.
5. The output is a web page, which is a list of names and weights (scores):
    - The weight of an image is related to the characteristics you are looking for (the weight is similar to an Internet search engine score);
    - Click the link of each image and an image will pop up on the screen.

25.3.6 Batch File

Batch commands allow you to save your setting and to execute your problem in a few clicks:

Click Batch/Save to save your setting; later click Batch/Load to load your setting and click Batch/Run to run your problem.