

# Lifespan Machine Software 2.0 : Model Building Tutorial

Nicholas Stroustrup. September 8<sup>th</sup>, 2020

nicholas.stroustrup@crg.eu  
Centre for Genomic Regulation  
Barcelona, Spain

## Contents

Making new worm detection models .....	1
Generating training set images to annotate .....	1
Annotating training set images .....	2
Building a large set of training set images .....	3
Training an SVM model file .....	4
Understanding your SVM model results .....	8

*Note: The annotated data set used to make published worm detection models are available for download, from Nick's group at the CRG. This data set makes a good starting point to which you can add your own annotated images. In this way, you can build upon others' efforts to build high-quality worm detection models.*

## Making new worm detection models.

### Generating training set images to annotate

The lifespan machine worm detection models are built from image data collected from scanners and then annotated by hand. As covered elsewhere in the documentation, the first steps of image analysis are 1) schedule and run an experiment 2) generate an image mask and apply it to your collected images 3) Run a series of image processing steps: "Median Filter", "Threshold", and "Worm Detection". There is an additional step not performed in standard analysis, which is "Add Detected Objects to Training Set". If you select and run this step (*Fig. 1*), you will generate a "training\_set" image that you can annotate by hand and use to train a new worm detection model.

**The Lifespan Machine**  
 Create New Processing Job [Back]

**Job Target(s):**  
 2018\_10\_31\_thermotolerance[chewie\_a [0]][Include Only Censored Regions]

**Schedule a Job for Individual Images**

Submission time: (Never)

Priority: Standard ▼

**Processing Tasks**

Only "Median Filter", "Threshold" and "Worm Detection" are necessary.

- ☐ Median Filter
- ☐ Intensity Stretch
- ☐ Threshold
- ☐ Worm Detection
- ☐ Worm Detection (Vis)
- ☒ Add Detected Objects to Training Set
- ☐ Compress Unprocessed Image for archiving
- ☐ Movement Paths Visualization
- ☐ Movement Paths Vis with Mortality Overlay

Save Job Delete Job Pause Job

**Schedule a Job For an Entire Region**

Submission time: (Never)

Priority: Standard ▼

Analyze Worm Movement

Save Job Delete Job Pause Job

**Update Region Information**

Only "Time at which animals had 0 age" is needed for analysis

Strain: ☐ TJ1060

Time at which animals had 0 Age: ☐ 13 : 0 / 10 / 31

Culturing Temperature: ☐

Experiment Temperature: ☐

Food Source: ☐ HT115

Experiment Conditions: ☐

Strain Condition 1: ☐ EV

**Figure 1—You can generate a “training set” image from any plate using the website interface.**

Training images are written to a single directory for each experiment:

/your\_long\_term\_storage\_directory/partition\_000/your\_experiment\_name/training\_set

A training set image is generated for every image collected, so you can potentially generate a very large number of images. In addition to the training set image, two extra files are generated, an “xml” file that includes metadata describing the training set image, and a “csv.gz” file that contains image quantification data for the image. You will never need to manually inspect the xml or csv.gz files.

### Annotating training set images

You can open and annotate a training set image using standard image program (Fig 2). However, the image data will look somewhat strange as the image data is encoded to include different information in each color channel. The blue channel contains the original grayscale image, and the red and blue channels contain segmentation information about what parts of the image are marked as foreground by the thresholding algorithm. In the training\_set file, the annotation for each worm is stored in the box next to each object. Objects marked with a white dot are indicated as “worms”. Objects without a white dot are “not worms”. Objects marked with a blue dot are marked “problematic / multiple worms”. Problematic objects are not included in training set generation.

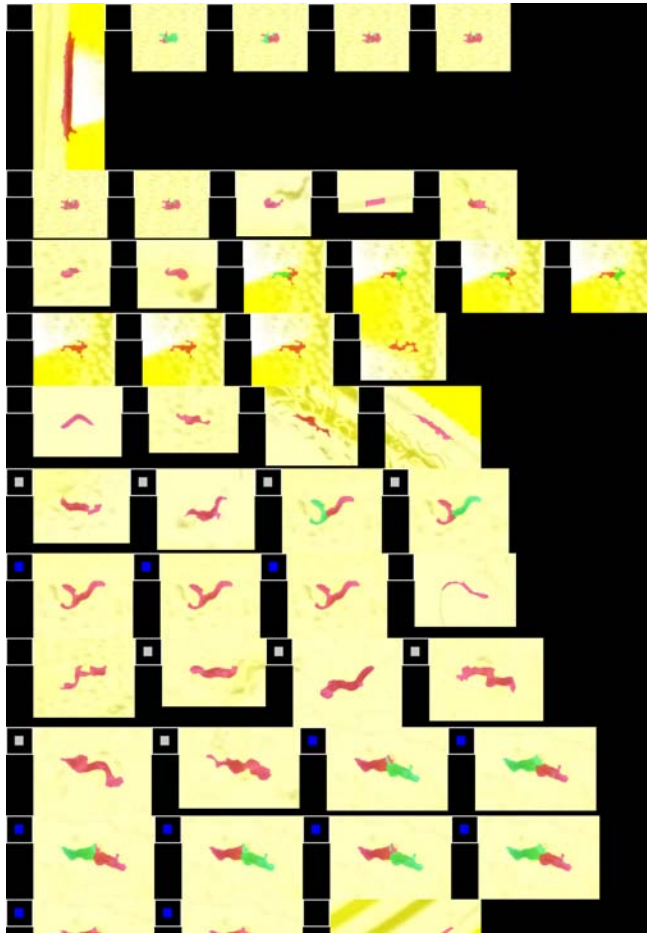


Figure 2—a training set image opened with a standard image viewing tool.

The easiest way to annotate these images is to load them using the worm browser. Choose the “menu item Calibration/Worm Detection/Annotate Worm Detection Training set”. The worm browser will then ask you to choose a training image that you want to annotate. If you select this directory, the worm browser will load the image in and break it down into easy-to-annotate strips—very similar to the “worm storyboards” that described elsewhere in the documentation. Unlike storyboards, you cannot click on images to get videos of the object in question. Since each object may contain several potential objects, the specific object being annotated is highlighted in red. Each object can be annotated as “A worm” (no outline) “Not a worm” (white outline) and “Problematic / multiple worms” (blue outlines). “Worms” and “not worms” are used to generate training files, whereas “problematic / multiple worms” are ignored entirely.

Using the arrow keys on your keyboard or “a” and “w” keys, you should annotate all worms in each strip of the image. When you are done, you can hit “Save” and your annotations will be stored in the image file.

### Building a large set of training set images

You want your training set to contain a diverse set of objects. It is important to include a diverse set of animals (different genotypes, ages) housed in variety of different conditions (placed different plates,


located on different scanners, measured in different experiments). It is not so important to include many images from the same plate, as there will be a high redundancy among them (as the images will all contain pictures of the same objects imaged at different times). Generally, the more diversity you can include in a data set, the better the model file you will get.

One strategy that works well is to create a new directory, empty, to hold all your training set images. Then, go through your experiments' saved training\_set images (see "Generating training set images to annotate") and copy a few images from each plate into your new directory. Remember to copy not just each training set image, but the associated metadata file (it has the same filename as the image but instead of ".tif" it has an ".xml" file extension) Do this for a couple different experiments. When you have a diverse set, go through using the worm browser and annotate each image.

## Training an SVM model file

When you have a data set annotated, you are ready to generate a new worm detection model.

1. First, using the worm browser, you need to generate a SVM training file. Select the option "Calibration/Worm Detection/Process annotated images to produce SVM Training Data". The worm browser will ask you to select the directory containing your training set images. When you do this, the worm browser will process all these files and generate a set of training data located in a new subdirectory of your training set directory, called "analysis".
2. To help you understand your training data, a few files will be written to this "analysis" directory. First, you will find a file called "worm\_stats.html" which you can load in any web browser. This file contains a series of "mug shot" photos of each object, along with all the quantitative metrics calculated for the worm.



Pixel Area	1933
Rectangular Width	61
Rectangular Height	77
Rectangular Diagonal	98.2344
Spine Length	118.85
Distance between ends	45.2598
Average Width	13.8985
Maximum Width	18.0176
Minimum Width	11.4017
Variance in Width	5.52388
Width at Center	22.7103
Width at Front	15.5259
Width at Rear	10.5134
SPLength / Pixel Area	0.0614845
SPLength / Rect Width	1.94835
SPLength / Rect Height	1.5435
SSPLength / Rect Diagonal	1.20986
SPLength / Maximum Width	6.59631
SPLength / Average Width	8.55128
SPLength / Width at Front	0.683648
SPLength / Width at Rear	0.462935
Ratio of end widths	1.47677
Average Curvature	0.306979

**Figure 3—the worm\_stats file generated, where you can inspect all the quantitative metrics calculated for each object in your training set.**

Another file "annotation\_summary.html" simply sorts all objects according to whether they are "worms" "non-worms" or "problematic / multiple". With all the objects juxtaposed in one place, you can quickly identify mistakes in your annotations. In most web browsers, you can right click on any mis-annotated image and say "view image", which will then allow you to see the image filename in the browser bar. This filename tells you the filename of the training\_set image that contains this mis-annotated object. Knowing the filename, you can go back to the worm

browser, correct the annotation, and then re-run the “Calibration/Worm Detection/Process annotated images to produce SVM Training Data” command in step one of this section. A final file “classifier\_comp.html” has a section for each quantitative metric, which is listed next to each object. This allows you to browse through objects and understand them in respect to a specific metric.

3. To build a SVM model, a variety of different training sets will be written to the subdirectory “analysis/training\_sets” of your training set folder. Each training set contains data from the same objects, but summarized using a different subset of quantitative features. For example, the training files “all\_train.txt”, “all\_test.txt”, “all\_range.txt” include all quantitative features describing objects. Feature selection is an important step in SVM model generation, but we have already done this for you. The best performing subset is called “high\_24\_plus\_fscore\_features”, which we recommend. Of course, for new applications, different feature subsets may also perform well, and if you like you can compare the performance of different feature subsets by re-running steps 5 through X on different feature sets.
4. The libSVM library is used to generate SVM model files used for worm detection. The easiest way to do this is to use the R programming language port of the libSVM library. To facilitate this, an r script is included in the lifespan machine source repository. This script is located in the repository directory lifespan \scripts\r\ns\_worm\_detection\_svm\_model\_generator.r , viewable on github at the url:

[https://github.com/nstroustrup/lifespan/blob/flow/scripts/r/ns\\_worm\\_detection\\_svm\\_model\\_generator.r](https://github.com/nstroustrup/lifespan/blob/flow/scripts/r/ns_worm_detection_svm_model_generator.r)

To use this r script, you need to specify directories of your analysis files. Two lines are important:

base\_analysis\_directory =

"X:\\microscopy\\lifespan\_machine\\2019\_10=worm\_detection\_training\_set\\analysis"

should be set to the location of your training set directory.

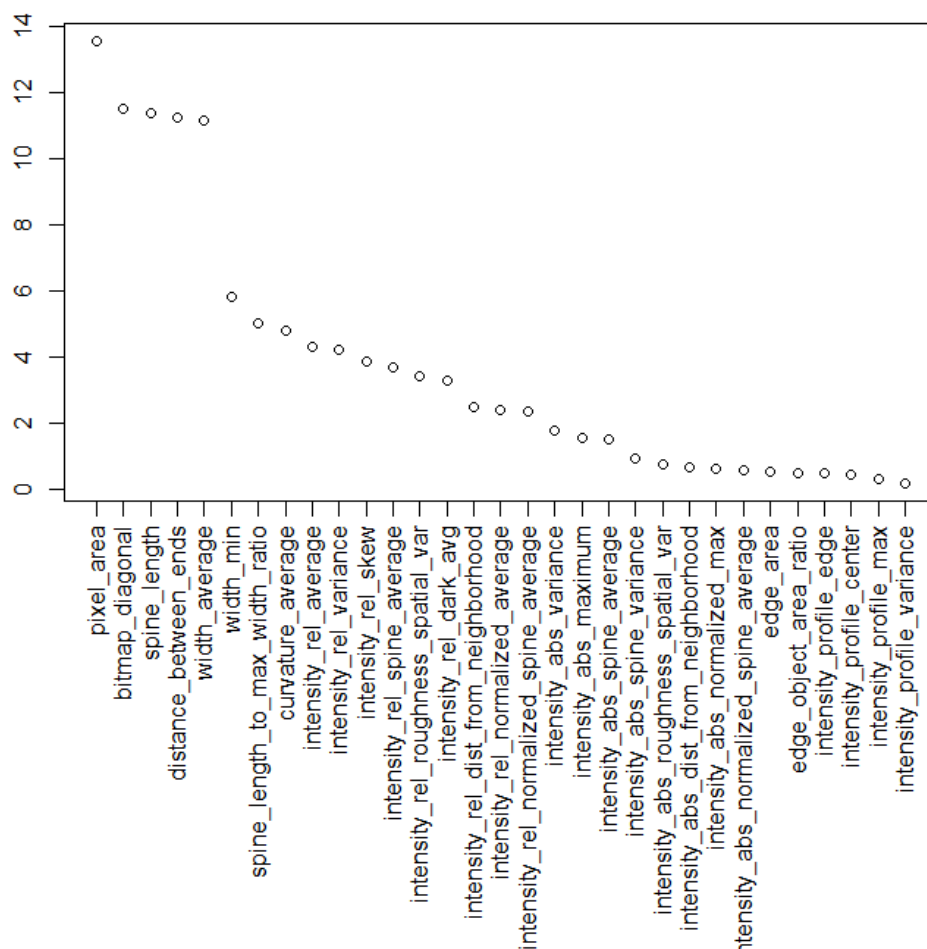
The second line to specify is:

model\_file\_to\_analyze = "training\_sets\\high\_24\_plus\_fscore\_features"

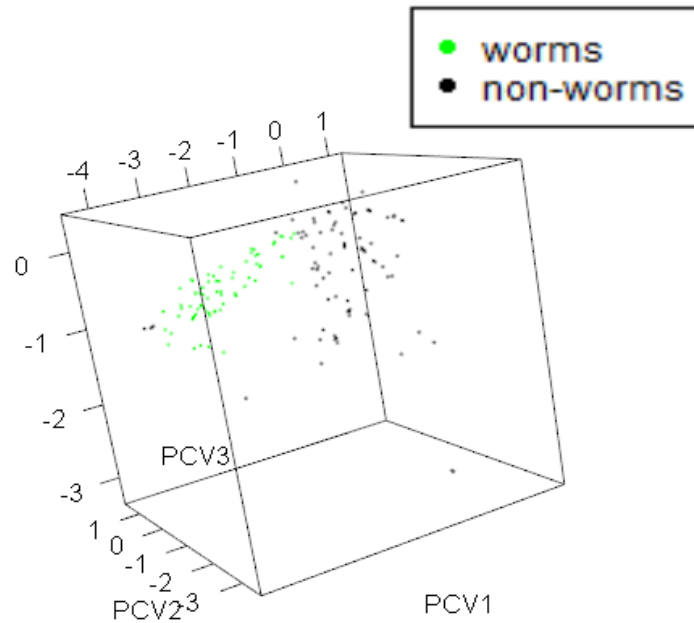
which should be set to the specific feature set you want to use. This should probably be left as the default, “high\_24\_plus\_fscore\_features”, unless you want to re-do feature selection.

Once you set these two settings, you can simply run the script in an R interpreter.

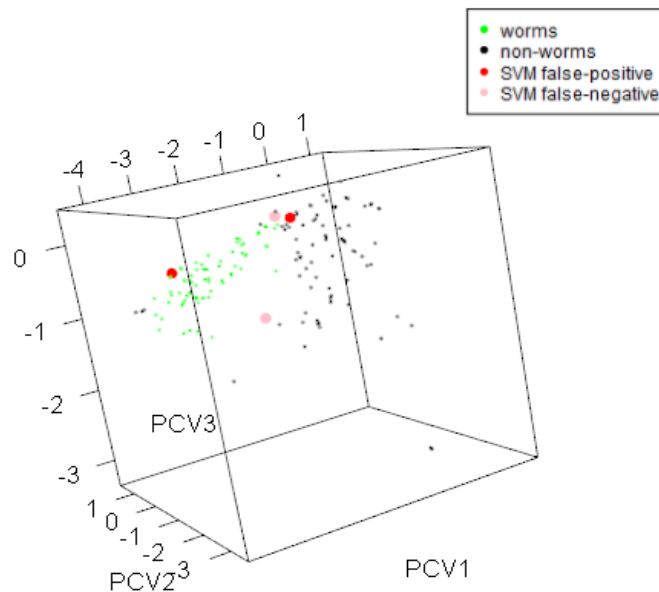
5. When you run “ns\_worm\_detection\_svm\_model\_generator.r”, you will get some diagnostic info about which features are most useful for discriminating worms and non-worm objects (Figures 4,5, and 6). The script will also output the SVM model file, which will be located in the “analysis/training\_sets” subfolder. That’s it! All you need to do is copy the files  
high\_24\_plus\_fscore\_features\_model.txt  
high\_24\_plus\_fscore\_features\_range.txt  
high\_24\_plus\_fscore\_features\_stats.txt  
into your model file directory, your\_storage\_directory/worm\_detection\_models/  
and you can immediately start using your new model file.



**Figure 4**—In this particular training set, “pixel\_area” is by far the most useful in discriminating worms from non worms.



**Figure 5—The `r` script included in the `lifespan_machine` github repository allows you to visually inspect the separation between worms and non-worms in the PCA-transformed feature axes. Because worms cluster away from non-worms, the discrimination task is likely to work well.**



**Figure 6—The `r` script included in the `lifespan_machine` github repository allows you to visually inspect the separation between worms and non-worms in the PCA-transformed feature axes. Here, the results of SVM classification have been overlaid, with false-positive and false-negatives highlighted.**

## Understanding your SVM model results

Often, it is useful to understand a bit more about the performance of the SVM model. The worm browser can help you do this. In the worm browser, choose the option “Calibration/Worm Detection/Analyze SVM Training Results”, and select the “high\_24\_plus\_fscore\_features\_results.txt” file produced by the libsvm R script. Running this command will create several useful diagnostic files. The first “test\_results\_high\_24\_plus\_fscore\_features.txt” contains information about the false positive and false negative rates for each genotype present in the data set. The best model files should provide good results for all genotypes. The next file is

“error\_summary\_high\_24\_plus\_fscore\_features=all\_genotypes.html”. This file contains pictures specifically of the objects that showed up as either false positive or false negative in the SVM model test. Often, many of these objects reflect errors in the by hand annotations—e.g worms annotated as non-worm objects, or vice versa. You can go back and correct the annotations, re-run the SVM fitting procedure, and generate an improved SVM model.