AvatolCV

Introduction

AvatolCV is a system that supports running computer vision and machine learning algorithms on images of biological specimens. The goal is to automate scoring biological characters, leveraging training data provided by the biologist.

AvatolCV is integrated via web services with two web-based biological image repositories:
Morphobank (`morphobank.org`) – a taxon vs character matrix based system
BisQue (`bisque.iplantcollaborative.org/client_service`) – an image centric repository for plant images

Dependencies

64 bit Windows / 64 bit OSX (Mac)
python 2.7.x
Java 8 64 bit
MATLAB R2015b
8G RAM

Configuring Python

Your system may come with python 2.7.x pre-installed.

To check on Mac, open up a terminal (Finder->Applications->Utilities->Terminal) and type "python --version". If present you will see something like : ($ is the Mac prompt)

```
      $python --version
  Python 2.7.10
```

If not present, install it from https://www.python.org/downloads/release/python-2711/. If "python --version" does not yield an answer, you will need to set the PATH directory to contain the directory with the python executable using this command:

```
	$export PATH=$PATH:<location where python was installed>
```

for example

```
	$export PATH=$PATH:/usr/bin/python
```

To check on a Windows machine, type "python --version"

```
	C:\>python --version
	Python 2.7
```

If not present, install it from https://www.python.org/downloads/release/python-2711/. If "python --version" does not yield an answer, you will need to set the PATH directory to contain the directory with the python executable. (Control Panel -> System and Security -> System -> Advanced System Settings -> Environment Variables. In the System Variables window, select Path and click edit. Click once in the Variable Value field to ensure it is no longer highlighted. Right arrow to the end and type a semicolon followed by the path to where python was installed, for example if python was installed at C:\, you would append this to the path: ";C:\Python27". Then click OK, click OK, click OK. Now open up a new command shell and "python --version" should yield "Python 2.7)

Getting the Installer

1. Got to https://github.com/AVATOL/AvatolCVInstaller
2. Click the button that says "Download ZIP"
3. unzip the file and then proceed with the instructions in the next section

Installing AvatolCV

To run it, open up a command shell on Mac and type:

```
	$cd whereTheDownloadedInstallerWasUnzippedTo
	$python updateAvatolCV.py directoryWhereYouWantAvatolCVToLive
```

EXAMPLE

	(create a directory to install into)
	$cd /Users/jirvine

```
$mkdir av
$cd /Users/jirvine/Downloads/AvatolCVInstaller-master
$python updateAvatolCV.py /Users/jirvine/av
(will create directory /Users/jirvine/av/avatol_cv with the system inside it)
```

or on Windows

```
      C:>cd whereTheDownloadedInstallerWasUnzippedTo
      C:\whereTheDownloadedInstallerWasUnzippedTo>python updateAvatolCV.py
directoryWhereYouWantAvatolCVToLive
```

EXAMPLE

(first, using Windows explorer, create a directory called c:\mystuff\av. Then...)

```
>cd C:\Users\irvinej\Downloads\AvatolCVInstaller-master
>python updateAvatolCV.py c:\mystuff\av
(will create directory c:\mystuff\av\avatol_cv with the system inside it)
```

Updating AvatolCV

If there is a bug fix and you need to pull in the updated system, merely run the same command that you used to install it initially.

```
(Mac)        $python updateAvatolCV.py /Users/jirvine/av
(Windows)    python updateAvatolCV.py c:\mystuff\av
```

How To Run AvatolCV

Once the files are in place, you would do (substitute your own directories) :

```
cd c:\mystuff\av\avatol_cv\java\lib
java -jar avatol_cv.jar
```

...to start the system

or on Mac

```
cd /Users/jirvine/av/avatol_cv/java/lib
java -jar avatol_cv.jar
```

NOTE - before starting an AvatolCV Session, it is recommended that you read the sections titled:

An AvatolCV Session
AvatolCV Scoring Session
AvatolCV Results Review Screen
Caveats, Limitations, and Assumptions

...so that you will know what to expect and are aware of assumptions made by AvatolCV and certain limitations of the system.

An AvatolCV Session

High Level Overview

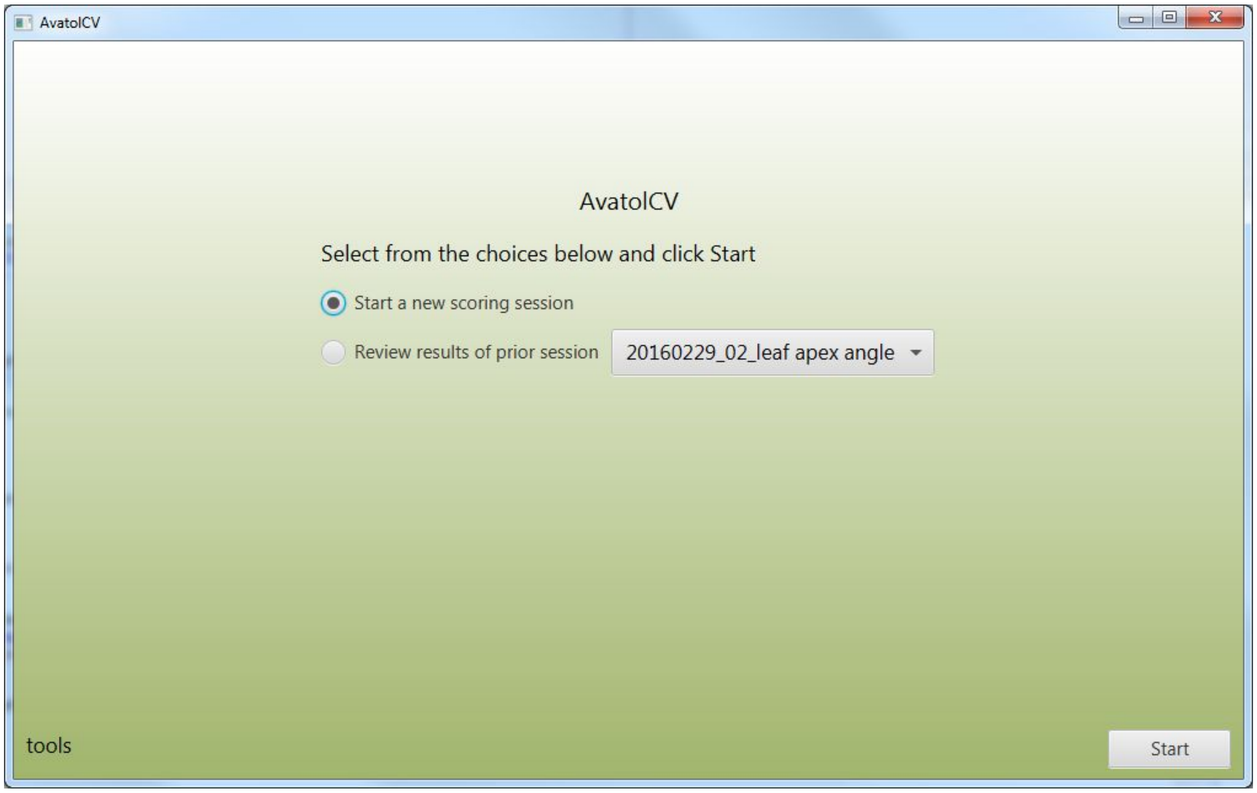
Before going into the details on each screen, here is an overview of the algorithm pipelines (sequences of algorithms) that work together as of this writing (May 17 2016). Since AvatolCV is a research project, the pipelines reflect the proof-of-concept nature of the work.

Pipeline	Platform	Segmentation	Orientation	Scoring
leaf	Mac	basicSegmenter	basicOrienter	shapeTextureScoring
batskull	Mac, Windows	skip	skip	partsScoring
nematocyst	Mac, Windows	higClutterSegmenter	NA	NA

AvatolCV has been designed to allow other algorithms to be plugged into it without modifying the system. (See “Adding New Algorithms to AvatolCV”) This means depending on which platform you are running on, certain algorithms will be available.

Start Screen

The start screen of AvatolCV offers two choices: starting a new scoring session or reviewing results from a prior session. Prior sessions are named byt the foloowing pattern:
<date>_<one-up number>_<characater name>
Date is of the form year-month-day. Date is assigned when the run is started, not when finished (in case it runs past the midnight threshold). One-up number is needed in case there are more than one scoring sessions for a particular character on a particular day.
For example, “20160415_02_leaf apex angle” refers to the 2nd scoring run of the character “leaf apex angle” on April 15, 2016.


AvatolCV

AvatolCV

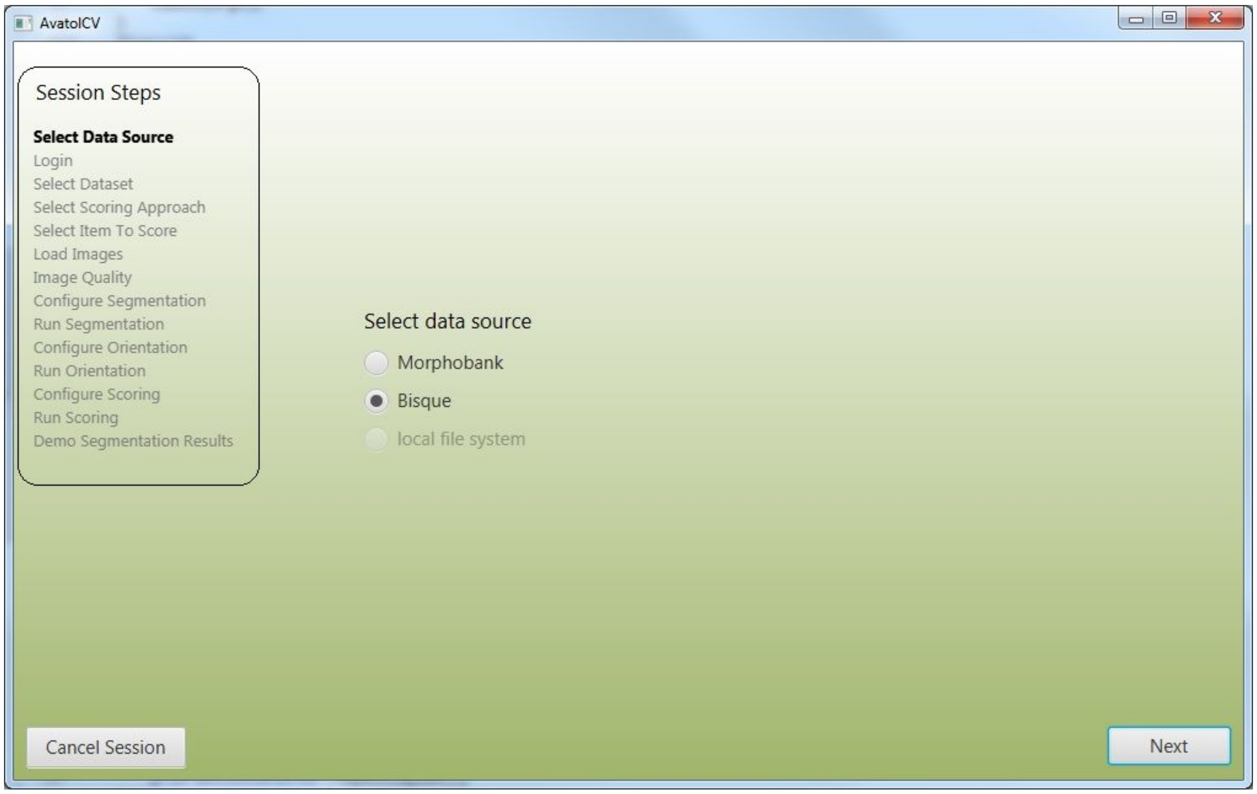
Select from the choices below and click Start

[✓ Start a new scoring session]

Review results of prior session   20160229_02_leaf apex angle

AvatolCV Scoring Session

Once "new scoring session" is chosen, the remaining screens show a list of steps (circled below) for the session on the left, and where in that sequence of steps the current screen lies. The greyed out items have not been reached, the black items have been completed, and the bolded item is the current screen.

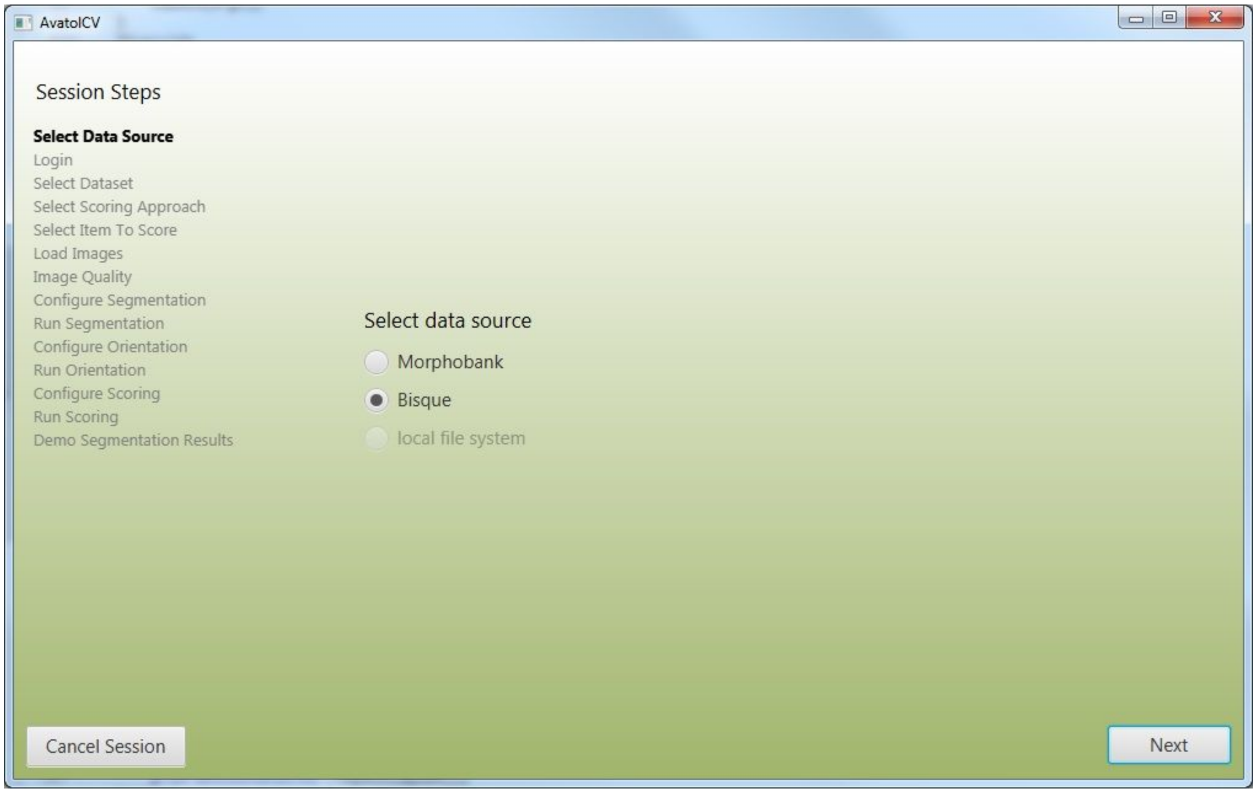


Select data source

Morphobank [✓ Bisque] local file system

Data Sources

AvatolCV is integrated using web services apis with Morphobank and BisQue. If either of these is selected, the DataSource component of AvatolCV will be interacting with the appoprriate system. There is also an option for choosing “local file system”. “local file system” will work only if the dataset (images and metadata) are formatted and arranged in AvatolCV’s internal data format. (See AvatolCV Internal Data Format section for details). *As of this writing (5/17/2016) a “local file system” run should work except that the functionality to save results has not yet been implemented.*

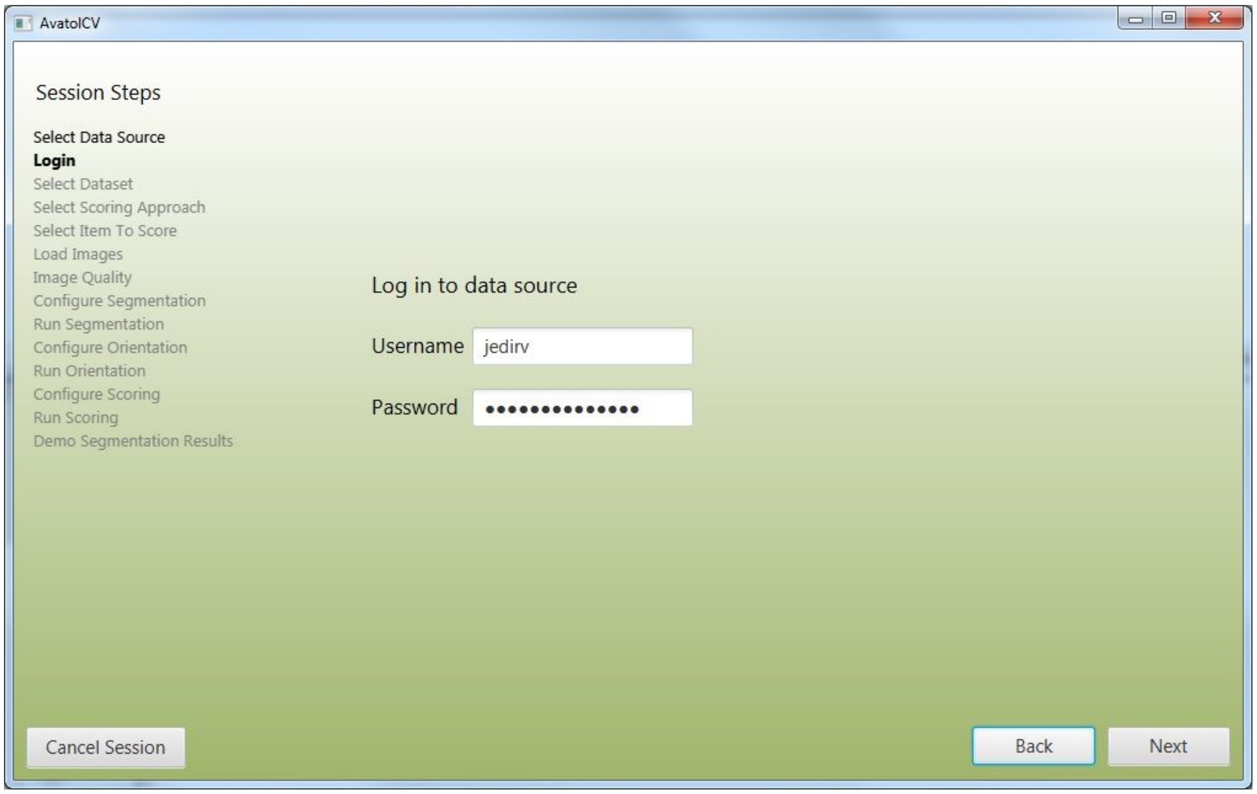


Select data source

Morphobank [✓ Bisque] local file system

Log in to Data Source

At this screen, the user should specify their username and password for either Morphobank or BisQue, depending on which they chose.



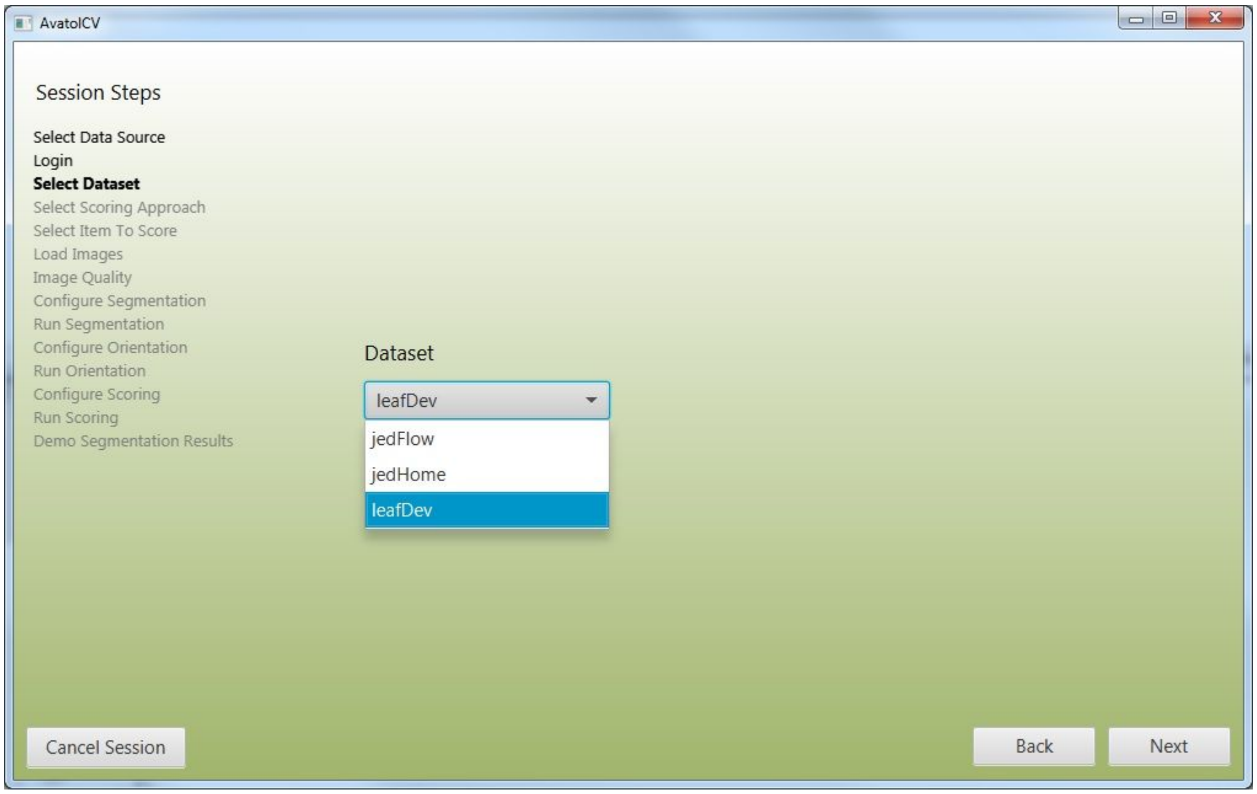
Log in to data source

Username jedirv

Password ●●●●●●●●●●●●●●●

Choose Matrix or Dataset

For Morphobank sessions, the user is presented with the list of the matrix names they have available at their Morphobank account. For BisQue users, they are presented with the names of the Datasets they have access to in BisQue.



Once the user makes a selection, enough metadata is pulled in from the data source to enable the system to pose the next question

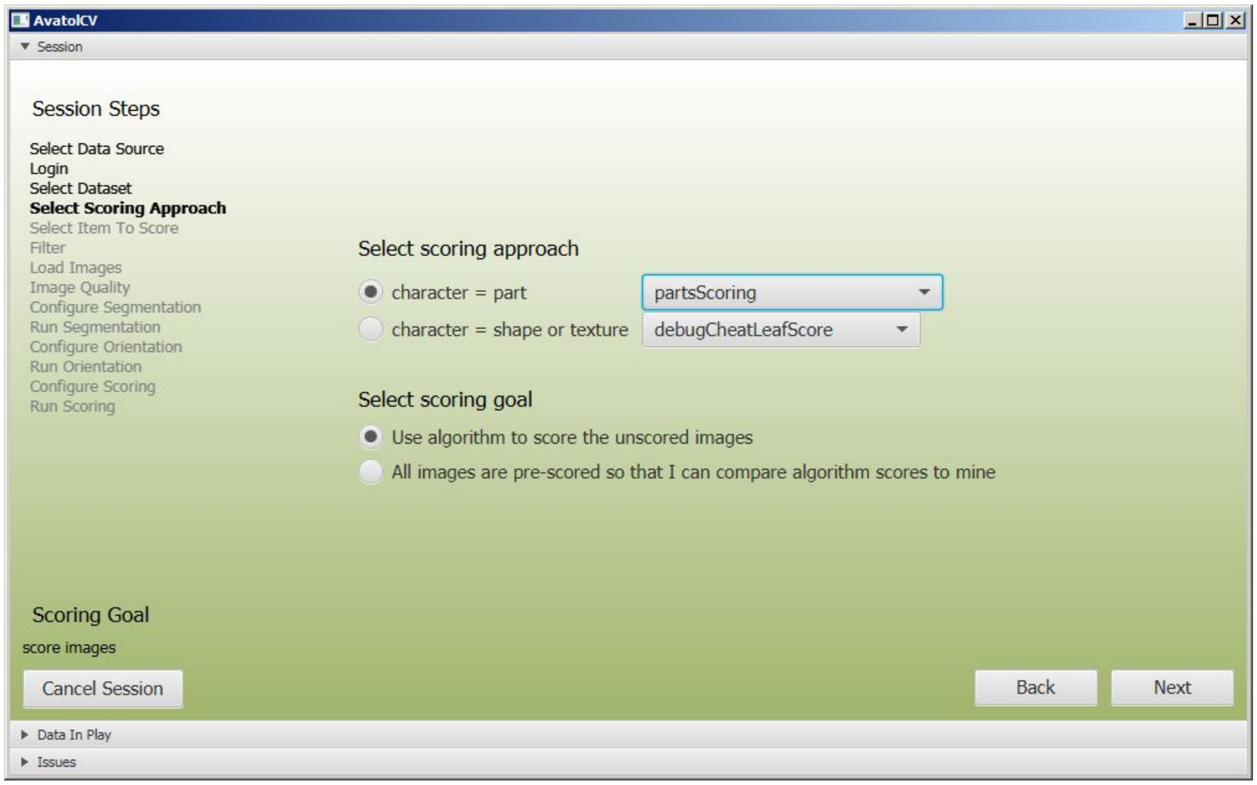
Select Scoring Approach

The choice of scoring algorithm will determine whether the character choice question will allow single or multiple answers (for scoring single characters vs multiple characters at once). Thus, the scoring approach must be selected as this next step. Two general choices are provided:

character = part
and
character = shape or texture

An example of character as part would be a tooth in a batskull. It is a visual region of the image which represents a distinct entity in the image. An example of "character as shape or texture" would be the angle of the apex of a leaf.

Depending on this choice, and depending on the platoform, particular scoring algorithms are available. The partsScoring algorithm is available on both Mac and Windows. The shapeAndTextureScoring algorithm is available on the Mac only. The system has been designed to accommodate additional algorithms being added later.



Select scoring approach

● character = part — partsScoring
○ character = shape or texture — debugCheatLeafScore

Select scoring goal

● Use algorithm to score the unscored images
○ All images are pre-scored so that I can compare algorithm scores to mine

Scoring Goal

score images

Scoring Goal

The second question on the scoring approach screen is the scoring goal. There are two goals that are supported: true scoring and evaluation. In true scoring, the scoring algorithm is trained using the images you've already scored, and then the algorithm will score the rest. In evaluation mode, all the images in play are already scored. The system will choose some to train on and then pretend the others aren't scored, and will score them automatically. Then the results are rendered in a way where you can compare your true scores to the algorithm's scores. Choose the first option for "true scoring" mode and the second for "evaluation.

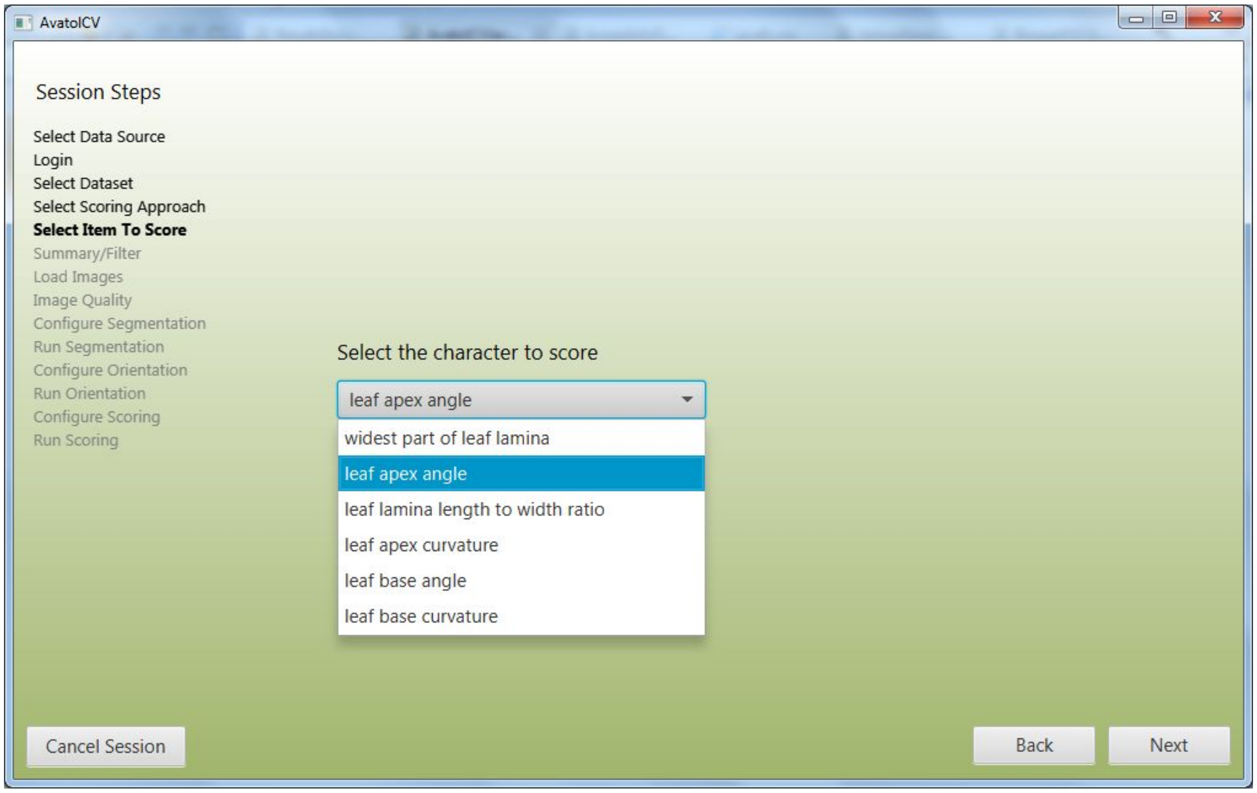
There is a nuance to mention. For the Morphobank session, AvatolCV will pull in all cells in the matrix (i.e. all taxa) for the chosen characters (see Select Item To Score). If you want to do an evaluation run but just on a portion of the taxa, select the second choice here(for evaluation). Then, later on at the scoring configuration screen, you can select "ignore" for the taxa you do not want to be involved.   Once you pass the filter screen, the system will point out issues with the data, but the ones involving the later-to-be-excluded taxa can be ignored.

Select Item To Score

This screen allows the user to select one or more characters to be scored.

NOTE - The partsScoring algorithm performs best when the all the presence/absence parts are scored at the same time – they help disambiguate each other. So, when multiple characters are selected, the scoring of those, in a single pass, generates multiple outputs, one for each character. For example, if the characters called X, Y, and Z were selected, then there would be three results generated for that single run: 20160516_1_X, 20160516_1_Y, and 20160516_1_Z. So reviewing this single run would require visiting three result pages. Results reviewing is organized by focusing on one character at a time. (See AvatolCV Results Review for more details on reviewing results)

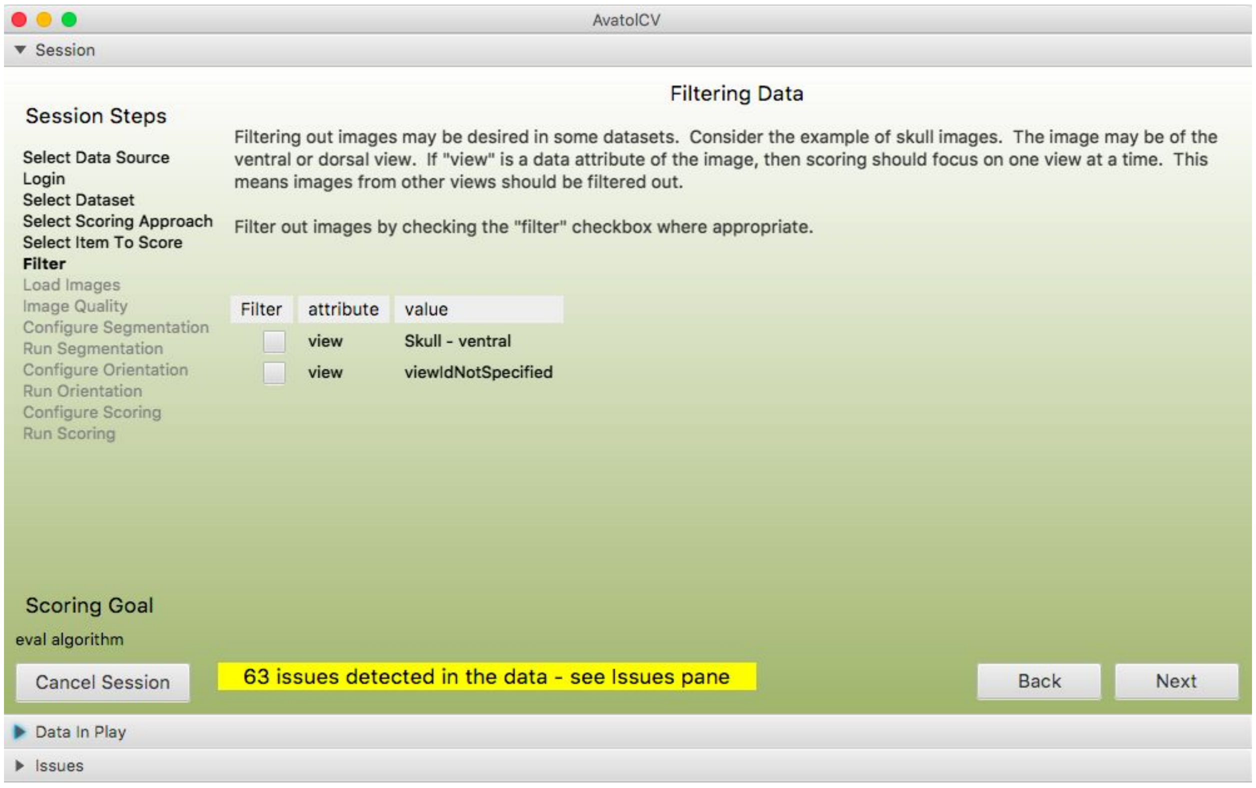
If the algorithm doesn't support multiple character scoring, then the character choices appear in a dropdown list.



Once the character(s) is selected, the remaining metadata (i.e. for those characters) is pulled in.

Filtering Data

For Morphobank matrices, each cell may contain multiple images for the taxon+character combination. These images might be taken from different viewpoints (ventral vs dorsal). Since (for some algorithms) AvatolCV requires images to be from a consistent aspect, the filtering screen is provided as a way to let the user exclude images if they are not of the desired view. However, this screen is generic to any dataset's situation. If there was a BisQue project, for example, where the images that had a certain attribute/value pair that should not to be included in the later processing, that attribute/value pair could be selected here to screen those images out. THe screenshot below is from a Morphobank session, where there are two views in play: ventral, and one where the view was not specified.



Filtering Data

Filtering out images may be desired in some datasets. Consider the example of skull images. The image may be of the ventral or dorsal view. If "view" is a data attribute of the image, then scoring should focus on one view at a time. This means images from other views should be filtered out.

Filter out images by checking the "filter" checkbox where appropriate.

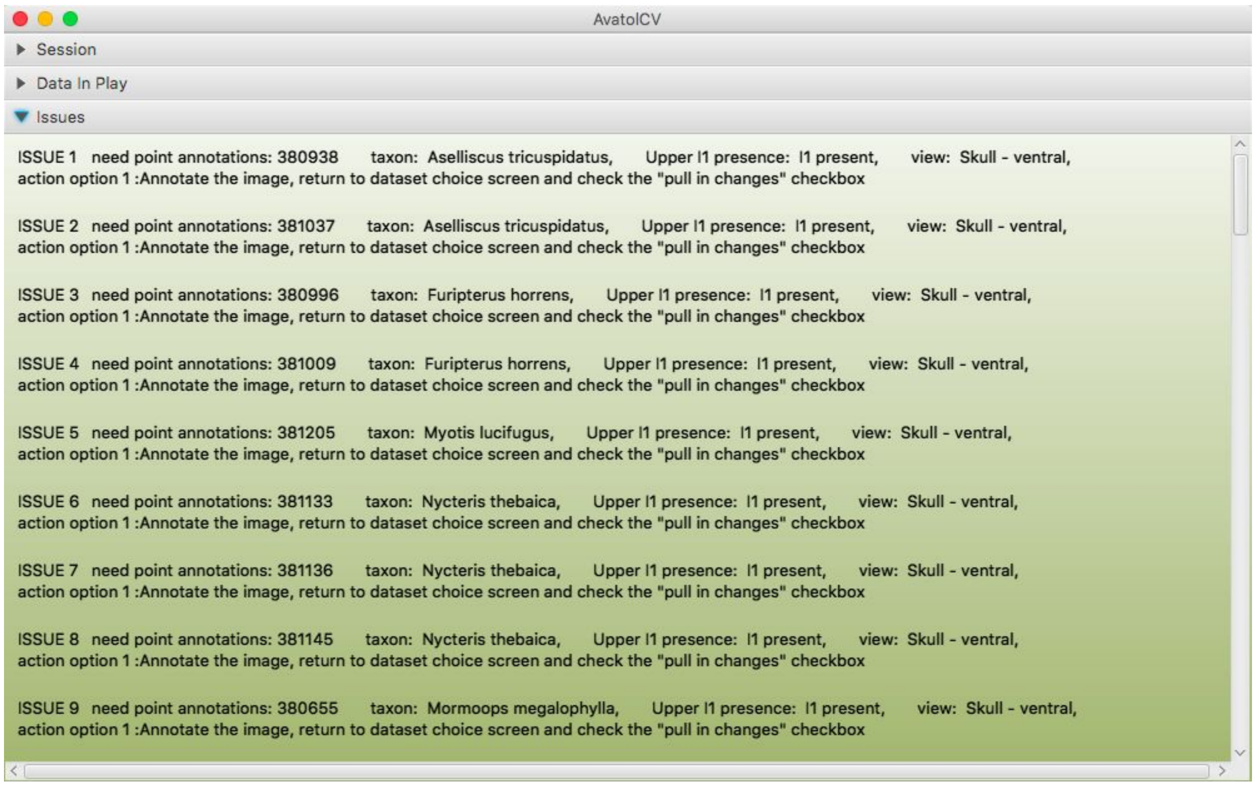
Filter	attribute	value
☐	view	Skull - ventral
☐	view	viewIdNotSpecified

eval algorithm

63 issues detected in the data - see Issues pane

Once filtering is complete, the system can now pull in images. Only the images that are relevant to the selected character(s) and which have not been filtered out, are pulled in. Also, if the images had been pulled in by a prior session, they are not pulled in again.

Note the yellow highlight on the bottom mentioning 63 issues. FOr the screenshot above, a Morphobank session was in progress, and the partsScoring algorithm had been chosen. The partsScoring algorithm makes a number of assumptions about the state of the data that will be fed into it. To reduce the likelihood that the user will launch partsScoring only to have it fail due to a violated assumption in the data, an issue detection system was added to the system. Once the filter screen is reached, the issue detection system runs at each screen to report on remaining problems, up until the scoring configuration screen is reached. If issues are reported,click on the issues bar at the bottom to switch to the window where each issue is described. Here is a screenshot of the issues screen:



ISSUE 1 need point annotations: 380938 taxon: Aselliscus tricuspidatus, Upper I1 presence: I1 present, view: Skull - ventral,
action option 1 :Annotate the image, return to dataset choice screen and check the "pull in changes" checkbox

ISSUE 2 need point annotations: 381037 taxon: Aselliscus tricuspidatus, Upper I1 presence: I1 present, view: Skull - ventral,
action option 1 :Annotate the image, return to dataset choice screen and check the "pull in changes" checkbox

ISSUE 3 need point annotations: 380996 taxon: Furipterus horrens, Upper I1 presence: I1 present, view: Skull - ventral,
action option 1 :Annotate the image, return to dataset choice screen and check the "pull in changes" checkbox

ISSUE 4 need point annotations: 381009 taxon: Furipterus horrens, Upper I1 presence: I1 present, view: Skull - ventral,
action option 1 :Annotate the image, return to dataset choice screen and check the "pull in changes" checkbox

ISSUE 5 need point annotations: 381205 taxon: Myotis lucifugus, Upper I1 presence: I1 present, view: Skull - ventral,
action option 1 :Annotate the image, return to dataset choice screen and check the "pull in changes" checkbox

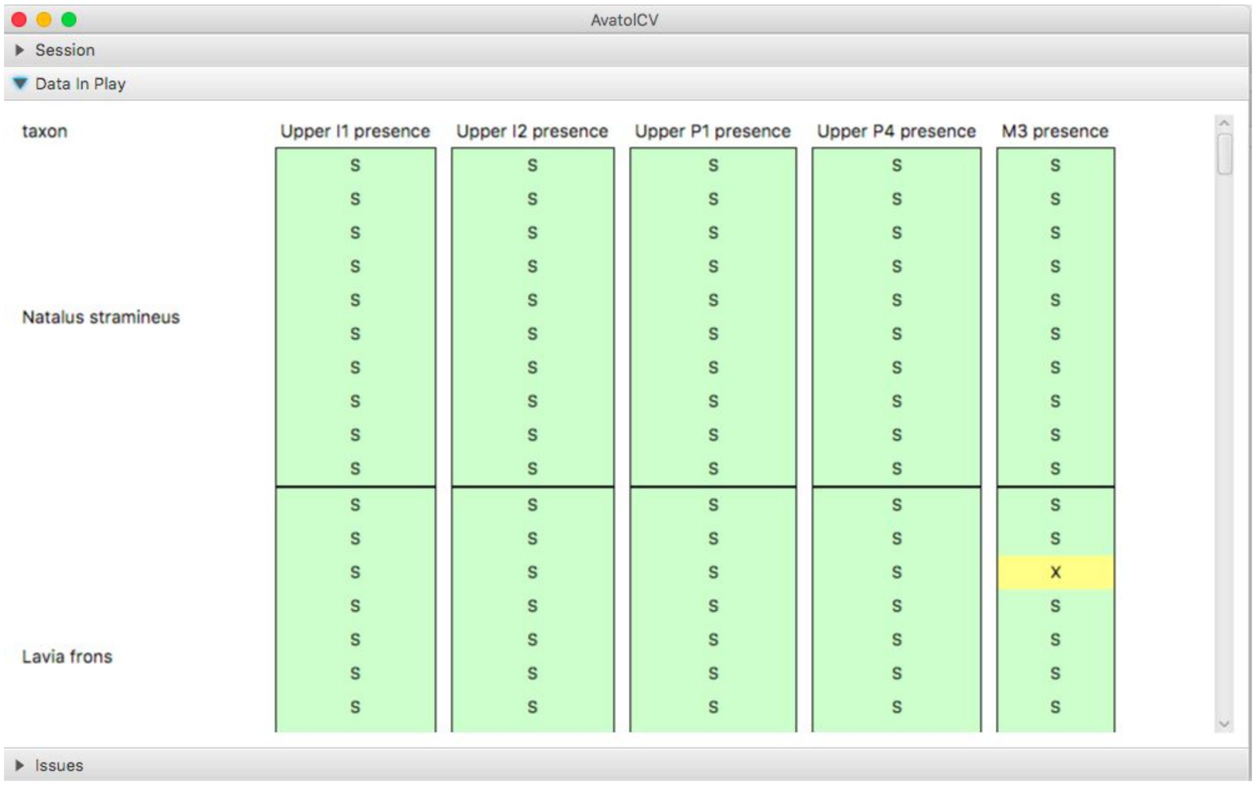
ISSUE 6 need point annotations: 381133 taxon: Nycteris thebaica, Upper I1 presence: I1 present, view: Skull - ventral,
action option 1 :Annotate the image, return to dataset choice screen and check the "pull in changes" checkbox

ISSUE 7 need point annotations: 381136 taxon: Nycteris thebaica, Upper I1 presence: I1 present, view: Skull - ventral,
action option 1 :Annotate the image, return to dataset choice screen and check the "pull in changes" checkbox

ISSUE 8 need point annotations: 381145 taxon: Nycteris thebaica, Upper I1 presence: I1 present, view: Skull - ventral,
action option 1 :Annotate the image, return to dataset choice screen and check the "pull in changes" checkbox

ISSUE 9 need point annotations: 380655 taxon: Mormoops megalophylla, Upper I1 presence: I1 present, view: Skull - ventral,
action option 1 :Annotate the image, return to dataset choice screen and check the "pull in changes" checkbox

For the case in this example, the dataset has 60 or so cells that are already scored, but that do not have the needed point coordinates specifying the locations of the tooth character. Clicking back on the Session bar at the top will return you to the session. To make it easier to address the missing point coordinates issue, a “Data In Play” window is populated (for Morphobank runs) that renders the information in a matrix format. Here is a screenshot of the Data In Play window:



taxon	Upper I1 presence	Upper I2 presence	Upper P1 presence	Upper P4 presence	M3 presence
	S	S	S	S	S
	S	S	S	S	S
	S	S	S	S	S
	S	S	S	S	S
Natalus stramineus	S	S	S	S	S
	S	S	S	S	S
	S	S	S	S	S
	S	S	S	S	S
	S	S	S	S	S
	S	S	S	S	S
	S	S	S	S	S
	S	S	S	S	S
	S	S	S	S	X
	S	S	S	S	S
Lavia frons	S	S	S	S	S
	S	S	S	S	S
	S	S	S	S	S

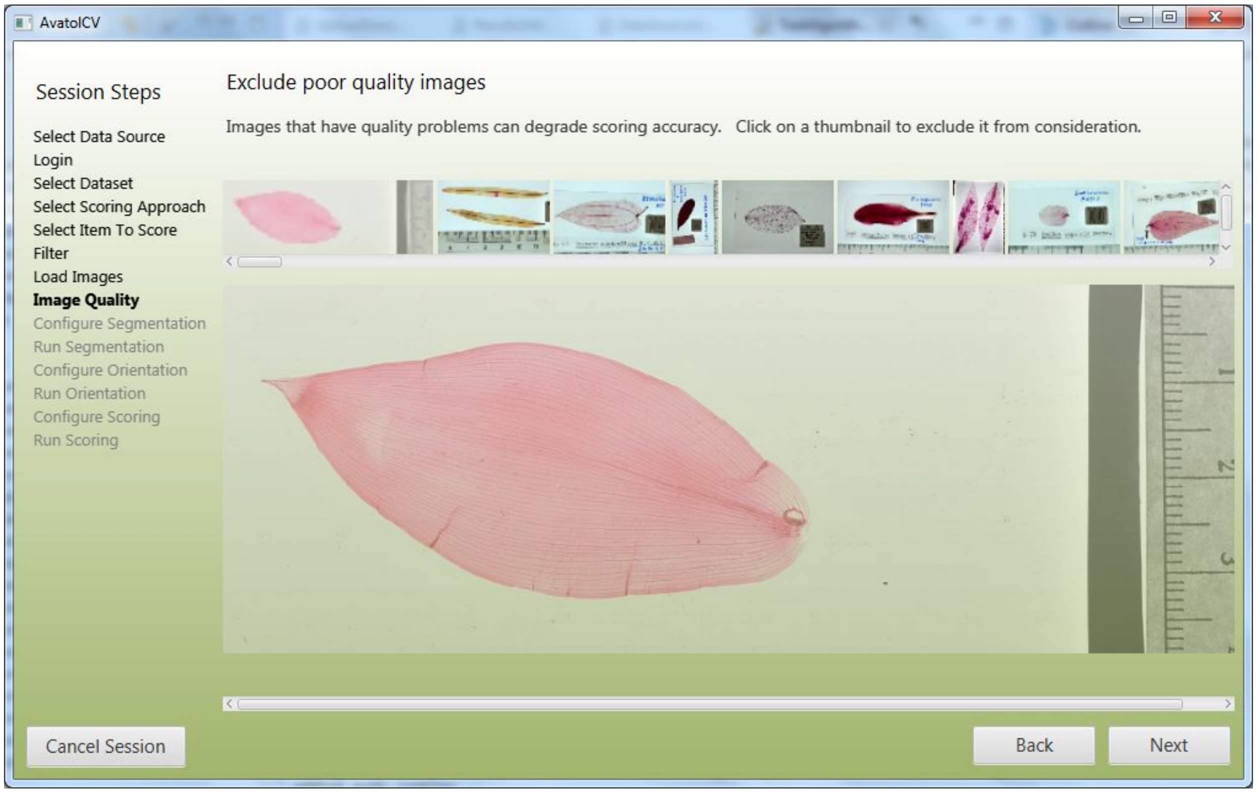
Just like the Morphobank matrix, the taxa names are down the side and the characters are across the top. In this case, there are about ten images per cell. Each image will have a different representation for each of these cases:

-scored and coordinates present:	green with “S”
-not yet scored:	blank
-scored, but missing coordinates:	yellow with an x
-image has been excluded:	exclusion reason

Once you encounter issues in a Morphobank session, the idea is to return to Morphobank’s standard web interface and fix the issues. In this case, you could use the Data In Play view to find the cells that had missing coordinate annotations, add the annotations in Morphobank. Then you would switch back to the Session view, and click on “Previous” until you came to the matrix choice window. Then you would click the “?” checkbox as seen below, and then click “Next”. This will cause all the metadata to be flushed and re-pulled so that your recent updates to the data will be onboard AvatolCV. Iterate until there are no issues or it is determined that the issues can be dealt with later. An example of this is if you are doing an evaluation run and you have scored a certain set of taxa, but not others. AvatolCV will raise the issue that for an evaluation run, all images need to be pre-scored. However, if the only images that are unscored correspond to taxa that you want to exclude from the run, you can leave those issues in play until you get to the scoring configuration screen, at which point you can select “Ignore” for those taxa.

Exclude Poor Quality Images Screen

All the images in play for the run are shown in thumbnail form. Hovering over one of the images shows a larger version of the image. If the image has quality issues as described shortly, it can be excluded so that it does not adversely affect the quality of the later processing. Clicking on a thumbnail will grey it out (and the larger version of it). This signifies that this image will be excluded. Clicking a greyed-out thumbnail will delete the exclusion marker and the image will once again be back in play.



Exclude poor quality images

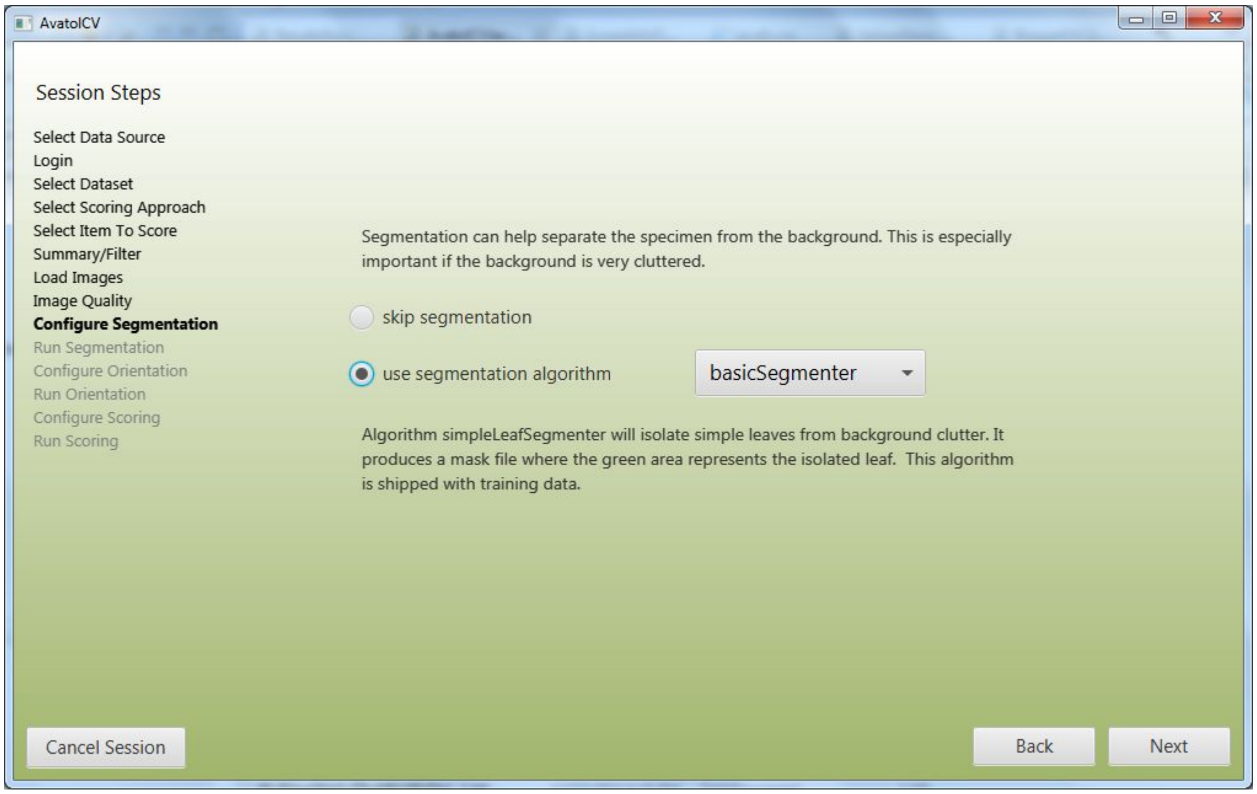
Images that have quality problems can degrade scoring accuracy. Click on a thumbnail to exclude it from consideration.

Segmentation

Segmentation can help separate the specimen from the background. This is important if the background is very cluttered. It may or may not be needed depending on the images.

The screen allows the user to skip segmentation or select a segmentation algorithm.
On Windows, the only segmentation algorithm that has been implemented is highClutterSegmenter. However, this algorithm generates output formats that aren't usable by either of the scoring algorithms in play as of this writing (May 17, 2016), so it's presence is just for demonstration purposes. highClutterSegmenter was deleveloped to try to segment nematocysts in images where multiple specimens were overlapping.

On the Mac, basicSegmenter, part of the leaf pipeline, is supported. It is trained on a leaf dataset. If other types of images are in play, the algorithm would perform better if training was on more relevant images. For instructions on how to retrain basicSegmenter, see the section called "User Supplied Training Examples"



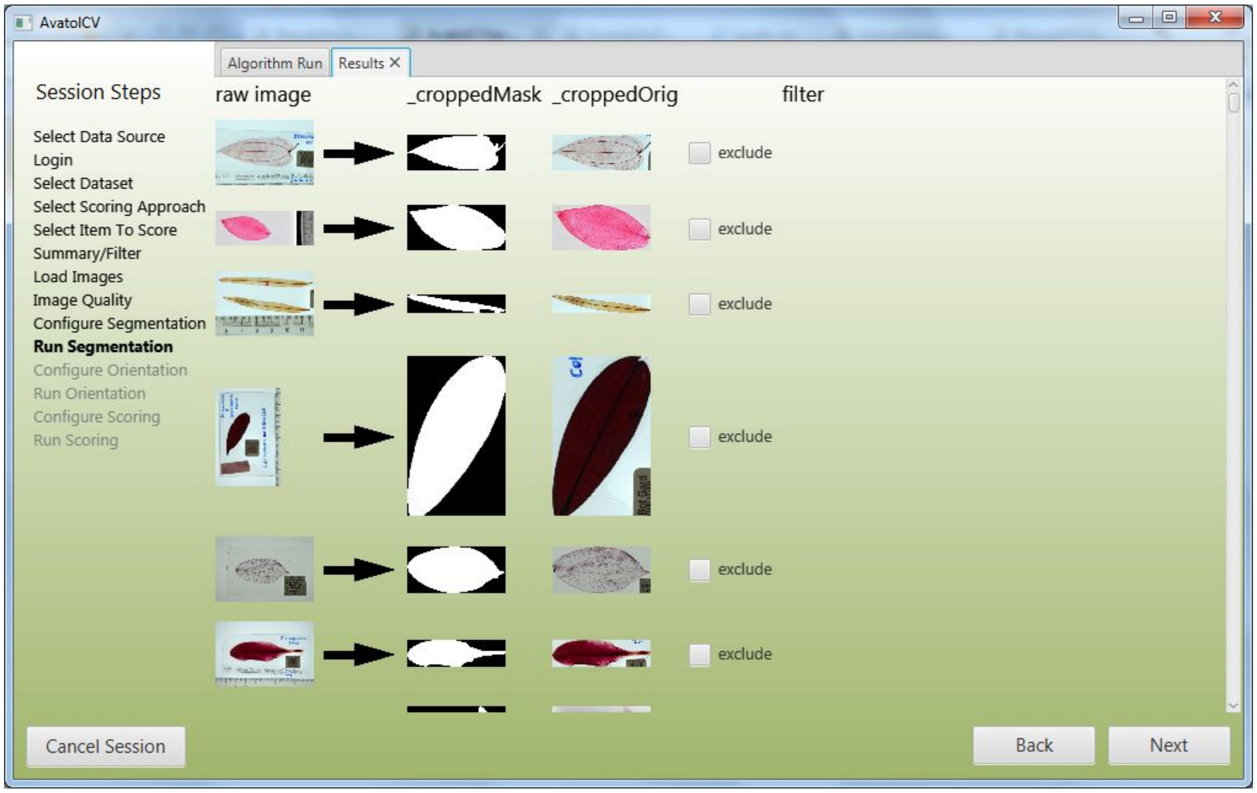
Segmentation can help separate the specimen from the background. This is especially important if the background is very cluttered.

skip segmentation [✓ use segmentation algorithm] basicSegmenter

Algorithm simpleLeafSegmenter will isolate simple leaves from background clutter. It produces a mask file where the green area represents the isolated leaf. This algorithm is shipped with training data.

On the Mac, when segmentation is running, the output is directed to AvatolCV Run Segmentation window. When the algorithm has finished, the AvatolCV automatically switches to the segmentation results tab on the same screen. This view shows the input and output images and provides a checkbox next to each for excluding that image from further processing.

The reason this would be done would be if the segmentation results were really bad, then there's no point in having later processing working with that segmentation output.



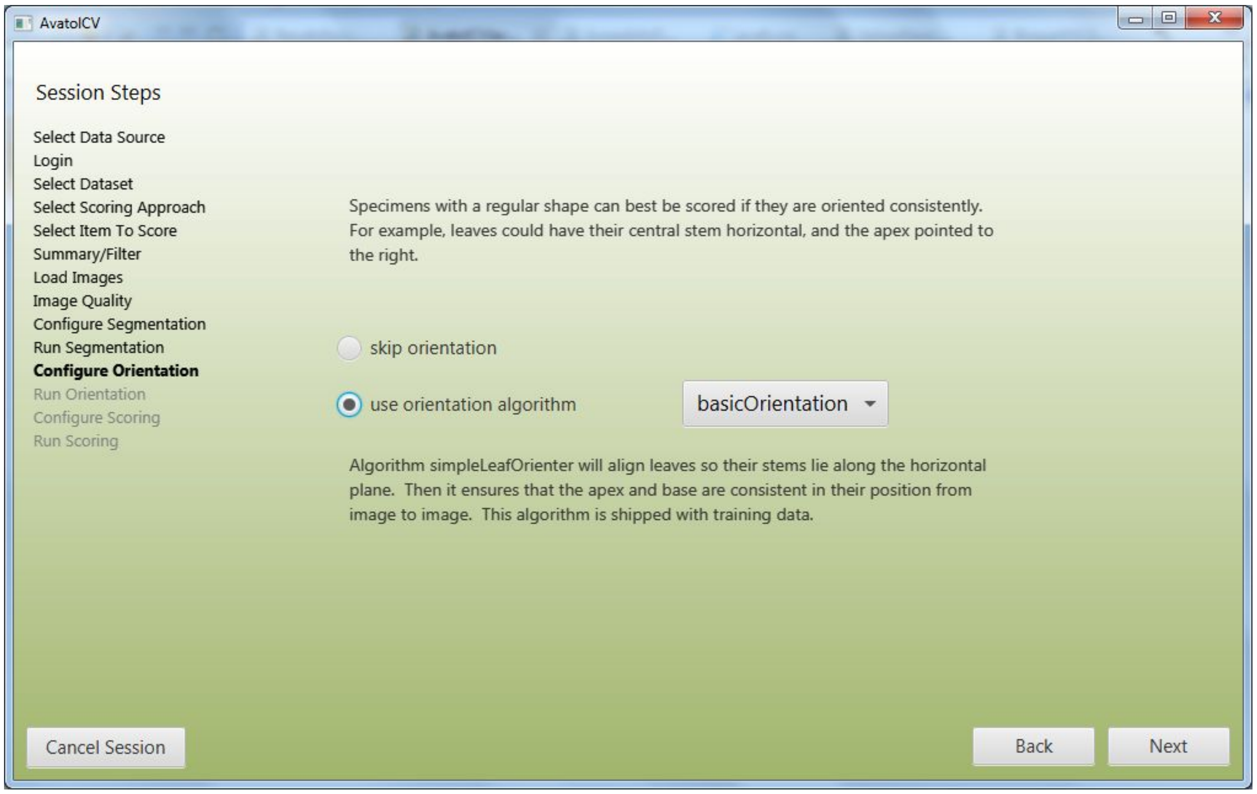
On Windows, then segmentation is running, the output is not directed to the AvatolCV Run Segmentation window, due to how Matlab (which is used by the algorithms) handles input/output on windows. To allow some visibility into how the algorithm is running, the system instead routes stdout (the console output) into a log file located at :

avatol_cv\sessions\<matrix_or_dataset_name>\<run_id>\logs\<algorithmType>

Where run_id is of the form date_number_characterName (20160415_01_leaf apex angle) and algorithmType is either segmentation,orientation or scoring.

Orientation

Orientation is to ensure that specimens are oriented in a consistent manner, for algorithms where that is important. The one orientation algorithm currently implemented is simpleOrienter (Mac only) which is trained using leaf images.



Specimens with a regular shape can best be scored if they are oriented consistently. For example, leaves could have their central stem horizontal, and the apex pointed to the right.

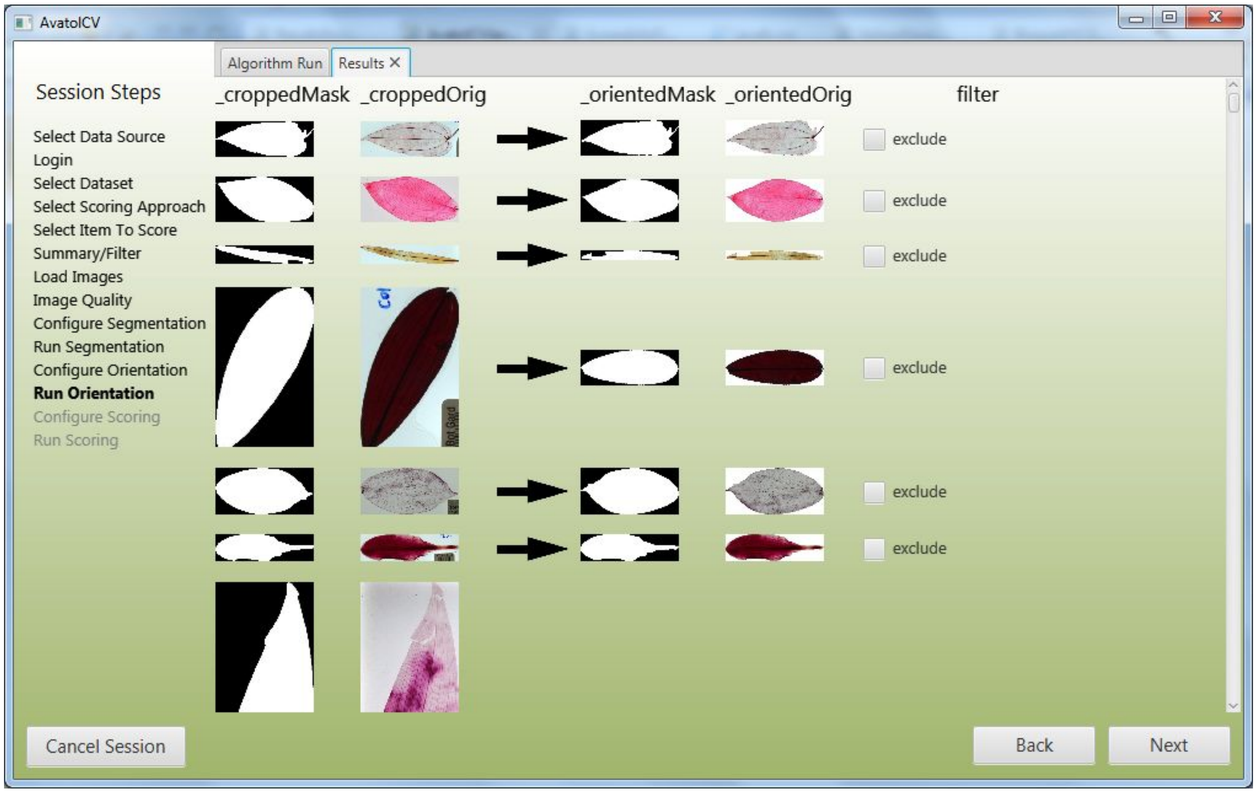
○ skip orientation

● use orientation algorithm   basicOrientation

Algorithm simpleLeafOrienter will align leaves so their stems lie along the horizontal plane. Then it ensures that the apex and base are consistent in their position from image to image. This algorithm is shipped with training data.

See the Segmentation section for a description of why algorithm output is not rendered into the AvatolCV window on Windows.

Orientation Results are shown in the same fashion as segmentation results - input images on the left and output images to the right of the arroa, with a checkbox to exclude those with poor results.


AvatolCV
Algorithm Run
Results
Session Steps
Select Data Source
Login
Select Dataset
Select Scoring Approach
Select Item To Score
Summary/Filter
Load Images
Image Quality
Configure Segmentation
Run Segmentation
Configure Orientation
Run Orientation
Configure Scoring
Run Scoring
_croppedMask
_croppedOrig
_orientedMask
_orientedOrig
filter
exclude
exclude
exclude
exclude
exclude
exclude
Cancel Session
Back
Next

Scoring Configuration

The scoring configuration screen controls how data is divided between training and scoring. There are different ways in which the data can be divided, depending on the nature of the data.

In “algorithm evaluation” mode, AvatolCV will divide the data into training and scoring sets. It will pretend that the images in the scoring set are not yet scored and will score them after training on the training set. The user may adjust which images are selected for training vs scoring, and also may choose to ignore certain images (certain taxa in Morphobank). Then, at the results screen, the score values can be compared to the user’s prior scores.

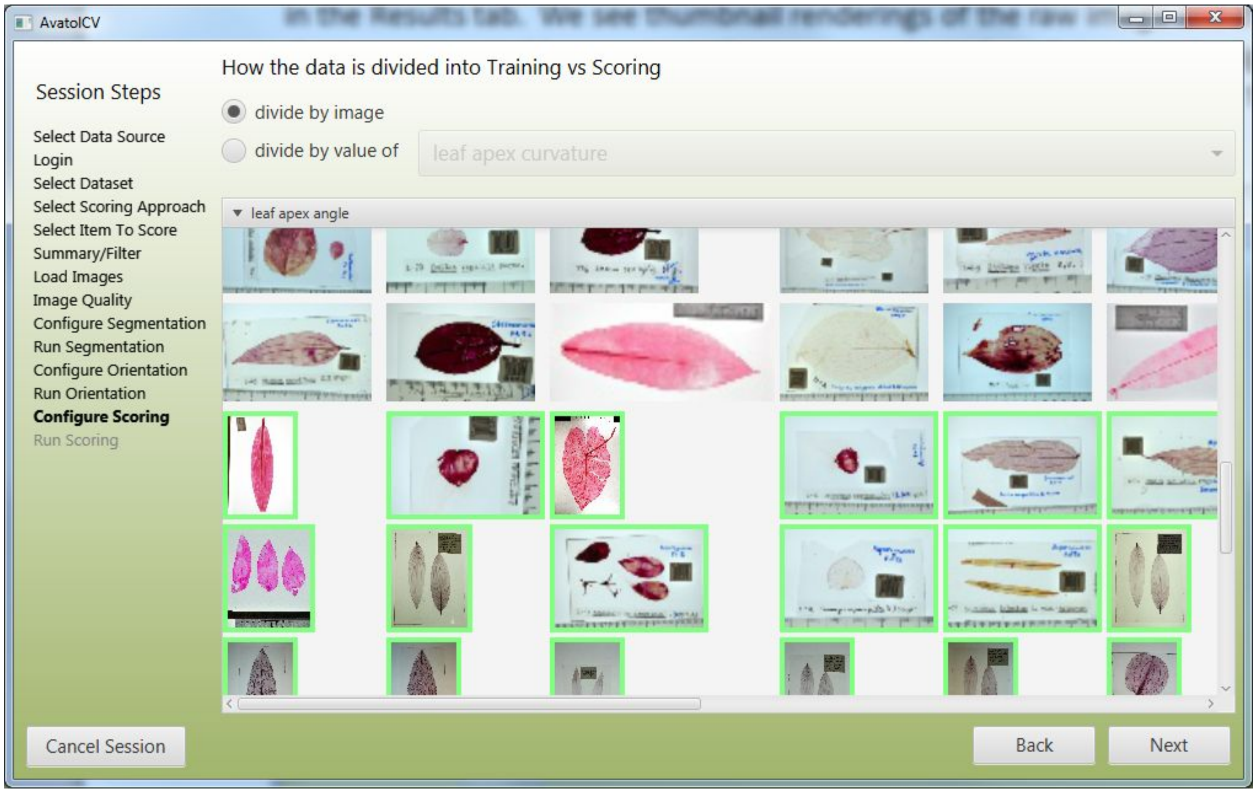
In “true scoring” mode, the training and scoring sets are predetermined by some of them having been scored and some of them not. The user is not given the opportunity to adjust which items are to be scored and trained as it wouldn’t make sense to do so in this situation.

Training Set vs Scoring Set Nuances

Another issue relevant to scoring configuration is whether there is a “trainVsTestKey” in play. In Morphobank, matrices have taxa on one axis, and character on the other. AvatolCV will want to train on some taxa and score on others. This makes the notion of “taxon” the trainVsTestKey for all Morphobank projects, in the the value of the taxon will determine whether an image is in the scoring set or training set.

So, the screen gives the choice of “divide by image” or “divide by value of”.

“Divide by image” is used when there is no trainVsTestKey in play, which will likely be the case for Bisque runs. The screen shot below shows this mode. The thumbnails with the green outlines are the ones that are in the scoring set. The ones lacking outlines are in the training set.



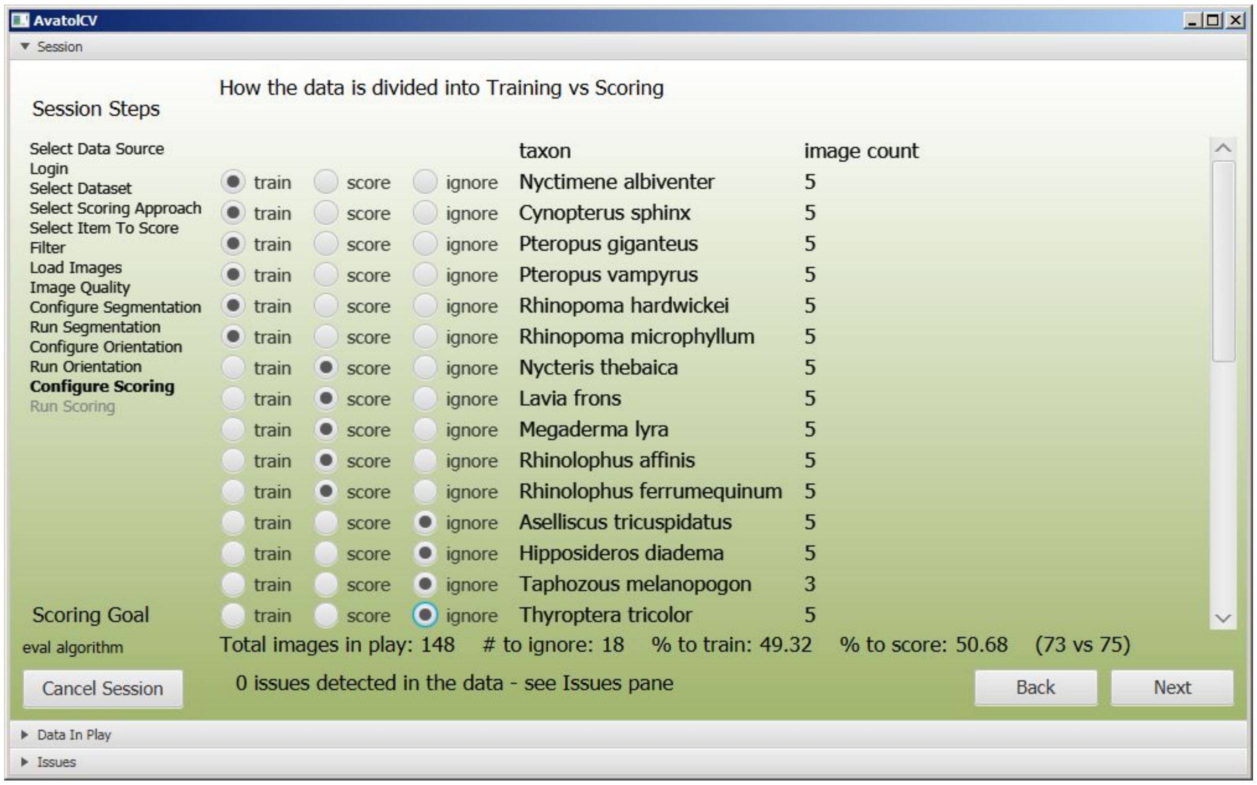
How the data is divided into Training vs Scoring

[✓ divide by image]

divide by value of leaf apex curvature

leaf apex angle

For Morphobank runs, “divide by image” is always disabled because taxon will always be the trainVsTestKey. Below is a screen shot of an evaluation mode run, where “divide by value of” is enforced because it is a Morphobank run, which will train on certain taxa and score on others.



How the data is divided into Training vs Scoring

			taxon	image count
[✓ train]	score	ignore	Nyctimene albiventer	5
[✓ train]	score	ignore	Cynopterus sphinx	5
[✓ train]	score	ignore	Pteropus giganteus	5
[✓ train]	score	ignore	Pteropus vampyrus	5
[✓ train]	score	ignore	Rhinopoma hardwickei	5
[✓ train]	score	ignore	Rhinopoma microphyllum	5
train	[✓ score]	ignore	Nycteris thebaica	5
train	[✓ score]	ignore	Lavia frons	5
train	[✓ score]	ignore	Megaderma lyra	5
train	[✓ score]	ignore	Rhinolophus affinis	5
train	[✓ score]	ignore	Rhinolophus ferrumequinum	5
train	score	[✓ ignore]	Aselliscus tricuspidatus	5
train	score	[✓ ignore]	Hipposideros diadema	5
train	score	[✓ ignore]	Taphozous melanopogon	3
train	score	[✓ ignore]	Thyroptera tricolor	5

Total images in play: 148   # to ignore: 18   % to train: 49.32   % to score: 50.68   (73 vs 75)

0 issues detected in the data - see Issues pane

The notion of trainVsTestKey is also supported for BisQue runs, but might not ever be relevant to the dataset in question. Where it would be relevant (for a BisQue run) would be if there was an attribute associated with images in the dataset where the user wanted to train on all the images where that attribute had a certain range of values, and score the ones where that attribute had a different range of values.

In summary, for a BisQue run, “divide by image” will always be enabled. “Divide by value of” will be enabled if there are any image properties other than the character(s) in play. (The automatically generated BisQue attributes of name and timestamp are ignored)   If the user decides that one of these other attributes should be a trainVsTestKey, then the user can select the “sort by value” option, and AvatolCV will divide the dataset by finding

IMPORTANT NOTES FOR MORPHOBANK USERS: please refer to the section called Training Data Consistency.

Scoring

The scoring screen shows progress of the scoring algorithm, just as Segmentation and Orientation do. When completed, AvatolCV automatically moves the user to the Results Review screen.(see next section) See the Segmentation section for a description of why algorithm output is not rendered into the AvatolCV window on Windows.

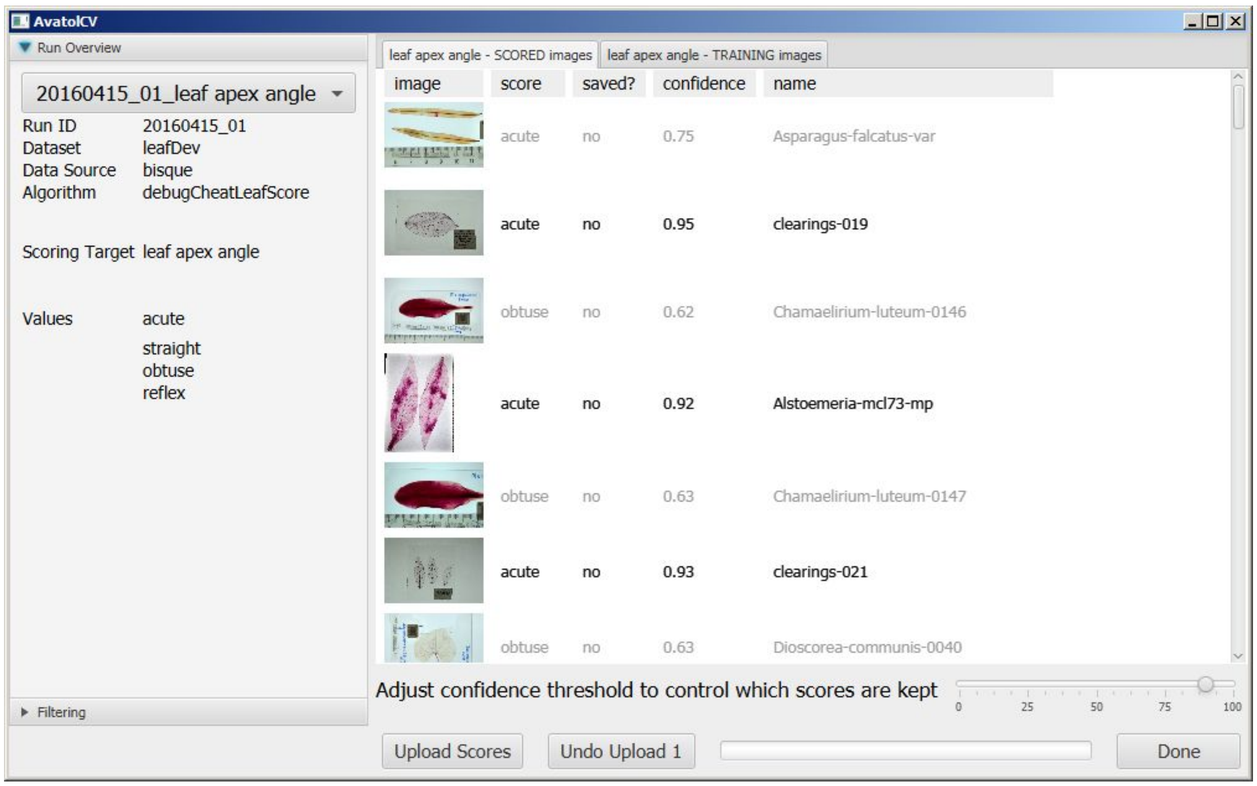
AvatolCV Results Review Screen

This screen appears when scoring is complete or when the user chooses to review results from a prior scoring session at the Start Screen. There are two flavors of Results Review Screen. They are distinguished by what information is presented.

“True Scoring” mode runs are runs done with datasets that have only been partially scored by the user. This means that when AvatolCV has completed its automatic scoring, there are no “true lables” to compare these results to. In “Algorithm Evaluation” mode, the dataset has been completely scored prior and AvatolCV has divided the data up into a training and scoring set. Then, when scoring is complete, we do have “true labels” to compare the scores to.

True Scoring Results

Below is a screenshot of the Results Screen for a True Scoring mode run.


AvatolCV
Run Overview
20160415_01_leaf apex angle
leaf apex angle - SCORED images
leaf apex angle - TRAINING images
Filtering
Upload Scores
Undo Upload 1
Done
0
25
50
75
100

Run ID	20160415_01
Dataset	leafDev
Data Source	bisque
Algorithm	debugCheatLeafScore
Scoring Target	leaf apex angle
Values	acute straight obtuse reflex

image	score	saved?	confidence	name
	acute	no	0.75	Asparagus-falcatus-var
	acute	no	0.95	clearings-019
	obtuse	no	0.62	Chamaelirium-luteum-0146
	acute	no	0.92	Alstoemeria-mcl73-mp
	obtuse	no	0.63	Chamaelirium-luteum-0147
	acute	no	0.93	clearings-021
	obtuse	no	0.63	Dioscorea-communis-0040

Adjust confidence threshold to control which scores are kept

The left side of the screen shows which run is selected, and this can be changed using that control. Under the run selector, we see the Run ID, the matrix or dataset name, the datasource used, and the scoring algorithm that was used.

Below that we see the character chosen for scoring specified as the scoring target. Below that are the range of values possible for that character. (the character states in Morphobank parlance).

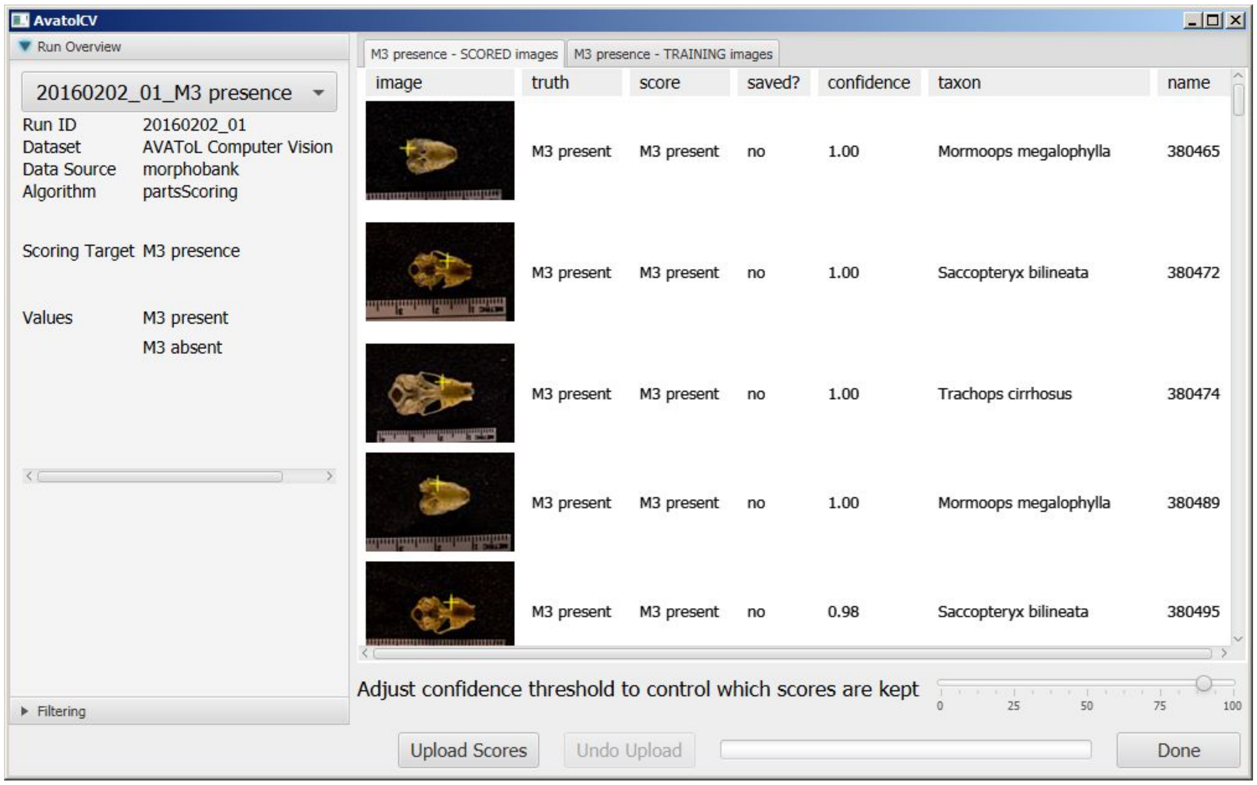
Remember that if the partsScoring run involved multiple characters, each character would have its own results screen that can be chosen in the selector at the top.

On the right side of the screen we see two tabs - one for SCORED images and one for TRAINING images. The training images are there for reference with their user assigned scores. More interesting is the scored image view which shows:

Image - a thumbnail for each image scored
Score - the score generated by the scoring algorithm
Saved? - an indication of whether this score has already been uploaded
Confidence - The algorithm's confidence in that score
Name - the original name of the image, if available

Algorithm Evaluation Results

The screen shot below is from an Algorithm Evaluation run so we can show all the columns that might be present..



Run ID	20160202_01
Dataset	AVAToL Computer Vision
Data Source	morphobank
Algorithm	partsScoring
Scoring Target	M3 presence
Values	M3 present
	M3 absent

image	truth	score	saved?	confidence	taxon	name
	M3 present	M3 present	no	1.00	Mormoops megalophylla	380465
	M3 present	M3 present	no	1.00	Saccopteryx bilineata	380472
	M3 present	M3 present	no	1.00	Trachops cirrhosus	380474
	M3 present	M3 present	no	1.00	Mormoops megalophylla	380489
	M3 present	M3 present	no	0.98	Saccopteryx bilineata	380495

Adjust confidence threshold to control which scores are kept

Notice these additional columns:

Truth - the true label (if it was an evaluation run)
Taxon - this is the name of the trainVsTestKey that is relevant for Morphobank runs

Note also that the Morphobank batskull images have yellow crosshairs. In the training tab, these crosshairs indicate where the user has specified that the part exists. In the scores tab, they indicate where AvatolCV thinks the part is.

In Morphobank, image names are not available in the web service api used to access the information so the internal Morphbank image ID is used to stand in for the image name.

Sorting Columns

Clicking on any column header (image, truth, score, etc) will sort the table on that column. Clicking on the same column header will then reverse the sort. Clicking on the image column has the same effect as clicking on the name column.

Zooming In and Out of an Image

To see an image in more detail, click the thumbnail and a large version of the image will appear below it. Click again and the large image will be removed.

Confidence Threshold Control

Below the scrolling image list, there is a slider for adjusting the confidence threshold. Any images with scores lower than the confidence threshold will be greyed out and ignored when scores are uploaded (saved to Morphobank or BisQue). IMPORTANT note for those running partsScoring - the confidence numbers generated by partsScoring for its scores are not trustworthy. To evaluate how partsScoring does on your data, do an evaluation run. When results are pulled up, a file is generated that summarizes how well the algorithms scores compare to your scores. The file is located at avatol_cv\sessions\<dataset name>\runID\scoredData\outputStats*.txt. FOr example,

avatol_cv\sessions\Bat Skull Matrix\20160726_04\scoredData\outputStats_20160726_04_M3 presence.txt

Also be sure to visually compare I(in the results viewer) the results that have correct scores to see if the point drawn on the image looks like it's in the correct place. This is important because the character state might be correct, but it might have been derived at the wrong place on the image, and thus shouldn't be considered as correct.

Uploading Scores

Once the user has set the threshold in such a way that they are happy with the scores above it, they can click the Upload Scores button at the bottom of the screen. Once pressed, the user will be presented with an authentication screen, unless they have already authenticated with Morphobank or BisQue already during that session. After that, the progress bar along the bottom of the screen will show how far in the upload process the system is. If the user decides they made a mistake and wants to undo the upload, they can click the Undo Upload button and scores at the data source will be restored to their prior value.

As scores are uploaded, the associated rows are highlighted and the value of the "saved?" column is changed from "no" to "yes".

A Morphobank Nuance

If a Morphobank matrix cell (taxon/character combination) has multiple images, then AvatolCV will score each of those images. However, when it comes time to save the scores, AvatolCV should provide a single score for the entire cell. Since it may be the case that AvatolCV scores images in the cell differently, AvatolCV needs to come up with a way of determining a single score value.

The process works as follows. If there are scores that conflict, AvatolCV counts how many are assigned each value. If one value has a majority of votes, then that score is uploaded. If there is a tie, then no score is uploaded.

Also, in the case of an Algorithm Evaluation run, AvatolCV will allow the user to upload scores if desired, even though all the values have pre-existing scores. If a high confidence AvatolCV score is deemed to be a better score than what the user had, then it should be saved. So, AvatolCV does the following check before uploading. If there is already a score, and that matches the AvatolCV score, then we don't bother uploading.

Whatever action is taken or not taken during the upload phase, the "saved?" column is updated with info on what happened for each cell as follows:

"no, score wouldn't change"	
"no, scores were tied"	
"changed to <some score>"	(when there was a prior score)
"set to <some score>"	(when there was no prior score)

So, if there were ten images in the cell, we would upload one score for the cell, and all ten images would have the same message in the “saved?” field.

Clicking Done will take the user back to the Start Screen.

Caveats, Limitations, and Assumptions

Dataset or Matrix Name Uniqueness

AvatolCV assumes that for a given user account in either BisQue or Morphobank, that the matrix or dataset names will be unique. Since AvatolCV places all the information about for a run under a directory named after the project or matrix, two same-named projects pulled into AvatolCV would have their files co-mingled, with corrupting results. Further, if a user has both a BisQue account and a Morphobank account, if they reuse the project name in both places, and pull those both into AvatolCV, those will also co-mingle and corrupt each other.

Training Data Sufficiency

An issue in general for CV algorithms is having sufficient training data. For Morphobank matrices, in order for partsScoring to function well, there must either be a large enough number of taxa involved or multiple images per taxon if there are fewer taxa. Development was done using a Morphobank matrix with around ten images per taxon and 24 taxa.

BisQue has no formal notion of taxon as part of its data model, though in theory it could be a property of the image and if so could be specified as the trainVsTestKey at the “divide by value of” control in the UI. For development we used a Bisque dataset with around 200 images.

Training Data Consistency

The partsScoring algorithm makes certain assumptions about the state of the data that is fed into it. There are specific issue checks that are run once the user makes it to the filter screen, with issues rendered in the Issues and Data In Play windows, as mentioned earlier. Here is an explanation of each issue and what to do if it is encountered:

ISSUE: unscored image - if the user has chosen an Evaluation run, the idea is that all images in the matrix should be pre-scored. Thus AvatolCV will note each unscored image as an issue that should be addressed.

Action to take: look in the Data In Play window. If any image boxes are blank it means that you should either go back to Morphobank to score that image, or that you will exclude that taxon later at the Scoring Configuration Screen.

ISSUE: taxa partially scored - for a True Scoring style run, partsScoring wants all images for a taxon to be either scored or unscored. Also, it is possible in Morphobank for an image for a specimen to be reused for different characters. There is a restriction in partsScoring that the same image not be used for both training and scoring. This is another reason why all characters for a given taxon should either be scored or not scored.

Action to take: either go back to Morphobank and finish scoring that taxon, or exclude that taxon at the Scoring configuration screen.

ISSUE: image lacks point coordinates - for either Evaluation or True Scoring mode, a scored image that lacks point coordinates will be flagged.

Action to take: go to Morphobank and add the annotation.

ISSUE: character has no unscored images - if you have chosen true scoring mode, but all the images for a particular character are already scored, it violates the assumption that each character will have some images to train and some to score.

Action to take: return to the character selection screen and deselect the character. OR, return to Morphobank and remove one score or the character. The latter seems counter productive, but may help the overall effectiveness of the partsScoring algorithm because it does better with more characters in play.

ISSUE: all chosen characters are disqualified - this is unlikely but could happen if each character had a circumstance that disqualified it. For example, let’s say in a true scoring run, a character has a single unscored image, but that image is excluded by the user for quality reasons. Then that character has no images to score, which is a problem for the true scoring case. Perhaps another character only had dorsal view images, and you had filtered those out, etc.

Action to take: depends on each circumstance..

ISSUE: multiple views in play - if there are images of more than one view for the characters chosen in the Morphobank matrix, then all but one must be filtered out - partsScoring needs a consistent view.

Action to take: at the filter screen, filter out unwanted views.

ISSUE: true scoring was chosen but all images are already scored.

Action to take: go back to scoring goal question and select evaluation mode.

ISSUE: no training data for character - one of the selected characters has no scored images

Action to take: go to Morphobank and score some taxa for that character, OR, go back and unselect that character for the run

Image Orientation Consistency

For partsScoring, specimens must be consistently oriented, for example, images of skulls should all be pointed in the same direction. As of this writing (May 17, 2016), this orientation must be done prior to images being pulled into AvatolCV, since the basicOrienter algorithm has been trained on leaf or leaf-like images so will not perform well on other types of images.

How To Run AvatolCV Using Different Versions of Matlab

Each of the following scripts refers to R2015b so you will need to adjust them before you try to run the algorithms:

For running on the MAC:
avatol_cv/modules/segmentation/yaoSeg/segmentationRunner.sh
(the line that says : /Applications/MATLAB_R2015b.app/bin/matlab -nodisplay -r "$matlab_func" quit)

avatol_cv/modules/orientation/yaoOrient/orientationRunner.sh
(the line that says /Applications/MATLAB_R2015b.app/bin/matlab -nodisplay -r "$matlab_func" quit)

avatol_cv/modules/scoring/leafScore/leafScore.sh
(the line that says : /Applications/MATLAB_R2015b.app/bin/matlab -nodisplay -r "$matlab_func" quit)

avatol_cv/modules/scoring/batskullDPM/batSkullScore.py
(the line that says : MAC_MATLAB_PATH = "/Applications/MATLAB_R2015b.app/bin/matlab")

For running on WINDOWS:
avatol_cv/modules/scoring/batskullDPM/batSkullScore.py
(the line that says : WIN_MATLAB_PATH = "C:\\Program Files\\MATLAB\\R2015b\\bin\\matlab.exe")

AvatolCV Internal Data Format

Morphobank Data Model

To be written… contact irvine@eecs.oregonstate.edu

BisQue Data Model

To be written… contact irvine@eecs.oregonstate.edu

Normalized Data Model

To be written… contact irvine@eecs.oregonstate.edu

AvatolCV Directory Structure

To be written… contact irvine@eecs.oregonstate.edu

Adding New Algorithms into AvatolCV

To be written… contact irvine@eecs.oregonstate.edu

User Supplied Training Examples

To be written… contact irvine@eecs.oregonstate.edu

What the Installer Does In Detail

AvatolCV is broken up into the following bundles:

```
docs
modules_3rdParty
modules_osu
java
```

usage: `python updateAvatolCV.py  <installRoot>`

Here is the process updateAvatolCV.py uses:

1. ensures specified installation root dir exists
2. creates an avatol_cv dir under that installation root dir
3. determines a platform code - what system we are running on : win | mac | unsupported
4. generates the name of the downloadManifest file that it needs to download (from http://web.engr.oregonstate.edu/~irvineje/AvatolCV/) using:

```
        downloadManifest_<platform_code>.txt
```

5. generates a temporary name that it will save that file under

```
    downloadManifest_<platform_code>_new.txt
```

6. downloads that file and saves as that name

```
        new_manifest_pathname = downloadFile(avatolcv_root, new_manifest_filename,
manifest_truename)
```

7. if this is not the first installation maneuver, the prior maneuver would have saved the file as...

```
    downloadManifest_<platform_code>_old.txt
```

        ...so we look to see if that file exists

8. If that old file exists we dump it to the console, then dump the new one to the console for sanity comparison by user

9. We determine which bundle names need downloading by comparing old and new.
   - If no old file exists, everything in new will be pulled.
   - If any bundle name represented in new is not represented in old, it is pulled
   - otherwise, we compare the datestamp (i.e. the version) associated with new to see if it is different than old. If so, we download.
   Note that we don't check for a more recent version, just difference with prior file. This will simplify regressing back to prior version if need be.

   If no bundles are needed, we declare we are up to date and exit, otherwise we continue

10. foreach bundle we need to download, we first uninstall the prior-installed version of that bundle using the

allFiles_<bundle_name>.txt

file that came with each bundle as a guide for which files to delete.

11. we delete the old manifest file and rename the new one to downloadManifest_<platform_code>_old.txt

...so it can be checked at the later download

12. now we install each bundle by looking up the bundle filename from the downloadManifest and pulling from the distro site.

This yields a .tgz bundle that we then unwrap underneath the avatolcv dir

Special instructions for trying large datasets in Morphobank runs

For those of you trying out partsScoring on large matrices, I wanted to give some step by step instructions in more compact form. Also, I'll list all the known issues at this point so that you are up on all the risks.

First the issues:

- I'm focusing on other projects currently so can't promise that I can follow through with significant technical support if issues arise. No harm in asking, as some issues might come up that are easy to address, but I won't have time to fix difficult bugs should they arise, or add features. That said, I've invested a couple years in the system and want it to be of use, so I'm motivated to want to help.
- The initial implementation of partsScoring (v1) was designed to train and score within taxa. This was a stepping stone toward the the current implementation (v2), which was designed to train on some taxa and score others. This matches the Morphobank user's use case but is a tougher problem. The results for v2 were not as good as for v1 on the data we had to work with, which matched expectations. We don't yet have a feel for how

much training data is needed before results become acceptable and this is very data dependant as well.

- As a reminder, partsScoring only works for simple presence/absence characters (ex. tooth is present or absent)
- partsScoring needs at least two characters in play for the run. The more characters it has in play, the better it can do but at time cost.
- Confidence scores are broken for partsScoring. After learning this I added a feature that makes it easier to get a sense of how many scores match true labels. This info is shown in the results viewer on the left side (for evaluation runs). It is also pesisted to a file at the time that particular result is loaded into the results viewer. However, Andrea found that a run on one of her matrices showed that many of correct scores were not fully correct in that the predicted location of the tooth was off in the weeds. This means that to REALLY assess whether the algorithm is performing well, you'll need to also review the locations of points in the results viewer for a number of the correct scores and see if they make sense. I'm confident that the number of training samples in play in Andrea's matrix is too small for the algorithm to get traction, but I don't have a sense of what the improvement curve might look like (accuracy vs # training samples) since our experience with data is so limited.
- partsScoring needs for only one Morphobank "view" to be in play (i.e. ventral, dorsal, etc). Extraneous views need to be filtered out at the filter screen.
- Recent work adding support for rectangle annotations revealed that point annotations will always yield 40 pixel squares around the center of the character, for use by the algorithm. Rectangle annotations can yield 35 - 80 pixels per side squares. This means that characters that are much larger than that, or elongated so that only a portion of the character is captured by the square, will not be scored with good accuracy.
- We've only run our system on a few datasets, none of them gigantic. The largest Morphobank project we've run partsScoring against has 24 taxa, ten specimen images per cell with five characters in play (1173 images) . Runs on this matrix would take some number of hours - I usually ran them overnight. It's possible that very large datasets could either run into memory issues or simply take too long.

Given these issues, I recommend doing an iterative approach.

- James, I think your dataset had 100 living taxa (i.e. skulls vs fossils). You could pre-score 50 taxa, 5 - 10 characters, then do an evaluation run on those where you train on 45 taxa and score 5. Assess the results. Assuming those aren't perfect (I doubt they will be), pre-score another chunk and do another evaluation run where you rescore the same 5. See if/how much the results improve and then decide from there.

- Jonathan and Phil, since you have more taxa, I'd recommend pre-scoring a hundred taxa to start with and then doing an evaluation run where you have training with 95, testing on 5.

If working through Morphobank, here are step-by-step instructions.

- First, a note about preparing your data. Orientation of images that are trained and scored together must be consistent. Since the orientation algorithm that was developed for AvatolCV is specific to leaves, this means your non-leaf images must already be oriented consistently in Morphobank. For example, for ants, you would want all the specimens to be facing the same way for a given view. Also, image backgrounds that are consistent in appearance will serve scoring purposes better.

- Startup AvatolCV
- New scoring session
- Choose Morphobank
- Give Morphobank credentials
- Select the desired matrix from the list (no need to check the "pull in changes" box unless you pulled this data in a prior AvatolCV session and have since updated the matrix somehow
- Select "partsScoring" and "all images are pre-scored…"
- Select the presence/absence characters you want
- Filter out any unwanted views and click next before you bother evaluating any issues that the system found.
- Once the images are pulled in, you will likely have a high number of issues mentioned, probably mostly complaining about cells not having scores, but many of these are expected since you've only prescored a portion of the matrix.
- Let's call the taxa that you have prescored (and that you will be doing an evaluation run with) "taxa_in_play". Let's call the rest of the taxa "taxa_to_ignore".
- Click on the "Data In Play" bar at the bottom to open up a graphical representation of the matrix (taxa vs characters) and the issues that the issue checker found. Find the taxa_in_play and see if any issues are shown for those that require fixing. (Issues for taxa_to_ignore you can leave unfixed). I didn't have time to add a visual key into that screen as it was a last minute addition but I put an explanation for that in the Filtering Data section of this document.
- Go back to Morphobank and address any missing point coordinates or other issues for taxa_in_play. Then, back up the UI to the matrix choice screen and the check the box marked "pull in changes". Then move forward again and after moving past the filter screen, check the Data In Play screen again to verify that AvatolCV is no longer complaining about any issues for taxa_in_play.
- Now you are ready to move to the quality exclusion screen, where you can click on any fuzzy or broken-specimen images, if present.
- At the segmentation screen, choose "skip"
- At the orientation screen, choose "skip"
- At the scoring configuration screen, click "ignore" on all the taxa_to_ignore. THe remaining taxa should be taxa_in_play. Select train on the ones you want to train on and score on the rest. Click Next.

- If you are on a Mac, you will see text scrolling by in the window to give you a sense of the processing going on. If you are on windows, you will instead see a disturbing lack of activity. This is because of the way Matlab is implemented in Windows - there was no direct way to capture the stdout and route it to the screen. It was possible, though, to route the output to a log file. Once the process is finished, the log files can be loaded into the screen by clicking on the "load log files" button. If you want to verify that the windows run is not hung, you can load the log files into the text editor. They should be located in the directory avatol_cv\sessions\<mb_project_name>\<runID>\logs\scoring\matlab_run_<someNumber>_invoke_batskull_system.txt
- Once the "Next" button is enabled, you can click it to move to the results viewer, which will show the results for one of the characters involved. You can switch to other characters' results using the dropdown list at the upper left.
- Each time you view a character, the stats file for that character's results is emitted to avatol_cv\sessions\<mb_project_name>\<runID>\scoredData\output_stats*.txt
- Remember to also look at the location of the crosshairs in the result images to assess how well the algorithm is doing.