3082
doi:10.1017/S1431927622011485
Microsc. Microanal. 28 (Suppl 1), 2022

Software Package for Efficient Creation of Training Data for Machine Learning Classifiers from Micrographs

Mathieu Therezien[1], Tomas J. McIntee[1] and Zachary E. Russell[1*]

[1] Ion Innovations, Boone, NC, United States.
* Corresponding author: zach.russell@ion-innovations.com

The extraction and classification of labeled objects from acquired images are tedious and time-consuming processes. We designed an integrated software that utilizes computer vision (CV) methods of image processing (IP) to automate the isolation of individual objects from an image and to output results in a format chosen by the user (allowing the use of any machine learning tools to classify those objects). The present proposal focuses on the tool in our software package that allows the user to visualize individual extracted objects in a series of images, sequentially tag each object, and save the results to file.

This suite is composed of tools that build upon each other to follow the workflow illustrated in Figure 1. A recipe builder provides an intuitive interface to let the user combine and parameterize IP instructions into a "recipe" tailored for the extraction of objects of interest in the types of image analyzed. These instructions (e.g., denoising, color thresholding, or morphological opening) each aim at removing certain pixels of an image that do not contain relevant information in the context of the analysis performed, while retaining those that do, i.e. the isolated individual objects of interest. Further IP methods then turn each object into a vector of base characteristic features (e.g., bounding box, area, or average components in RGB color space) that represents the object in a limited multidimensional feature space. That feature vector can optionally be expanded to include a much broader set of advanced features (e.g., convexity, geometric moments, or average and standard deviation components in a variety of color spaces). The objects and their feature vectors are then passed along to the tagger tool presented here.

The tagger (Figure 2) is a simple interface that lets the user visualize the objects found in an image and, for each one of them, decide which "type" it belongs to from a selection of user-defined choices. It sequentially goes through all the images selected, and through all the objects detected in each image. The bounding boxes of detected objects are drawn in gray, the one being tagged in bright green, and the already tagged ones in red. This lets the user focus on identification rather than also having to tally up the different types and keep track of what they have and have not tagged yet. If a detected object happens to be an aggregate, the user can enter the number of elements of each type contained in the aggregate, in which case the composition of the object gets saved in a dedicated field, and its type set to aggregate.

As optional functionalities, the tagger can also be connected to a classifier of the user's choice, opening up three additional working modes. In training mode, the manually tagged data is directly used to feed and train a new classifier. In review mode, objects tagged by an automatic classifier are displayed with that tag, which then gets verified by an expert, either to evaluate the classifier performance, or to correct its results. In that mode, the user can customize which features get displayed alongside the object and, if available, the confidence threshold over which a classified object does not require expert review. Finally, learning mode combines both review and training modes, such that the tagged output from a

classifier that passes review (or has a high enough confidence, if available) is added to that classifier, incrementally increasing its accuracy and reducing the need for expert review.

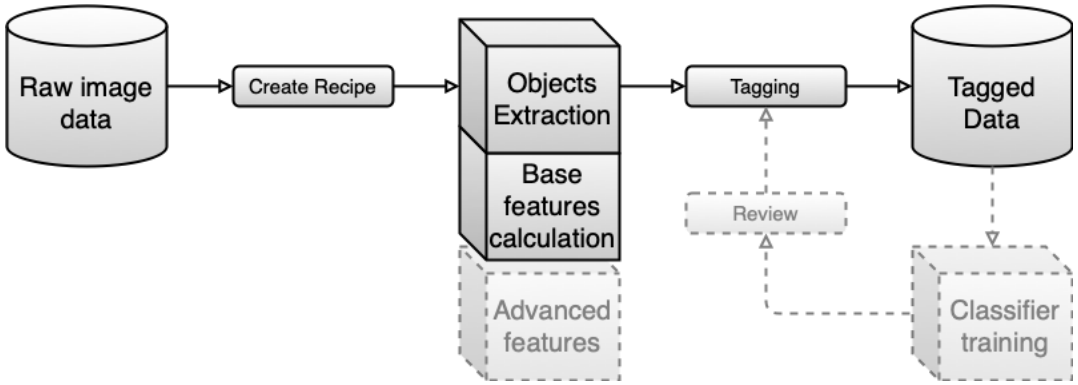

Figure 1: Tagger workflow from an acquired image. The tagger is flexible on its input and output data types, which can be fed into the classification system of the user's preference. Output from a classifier can also be fed back into the tagger for review of results, and the object extraction can be used to extract advanced features for use in classifications if desired.

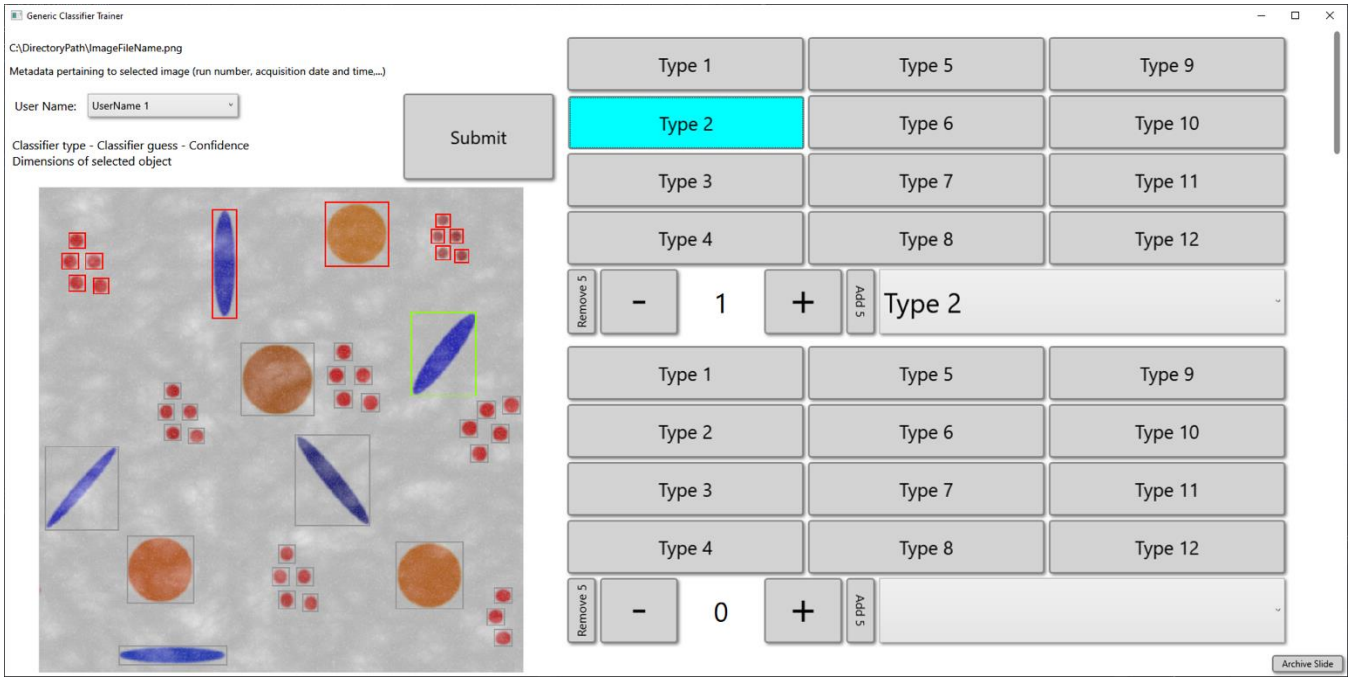

Figure 2: Tagger used on a mock-up image consisting of simple objects of different colors, sizes, and shapes, with added noise and blur. Detected objects are highlighted in gray, the currently tagged object in bright green, and the already tagged ones in red.

https://doi.org/10.1017/S1431927622011485 Published online by Cambridge University Press