Draft version April 8, 2016
Preprint typeset using LaTeX style AASTeX6 v. 1.0

GALAXY ZOO EXPRESS: INTEGRATING HUMAN AND MACHINE INTELLIGENCE IN MORPHOLOGY CLASSIFICATION TASKS

Melanie Beck, Claudia Scarlata, Lucy Fortson
Minnesota Institute for Astrophysics, University of Minnesota, Minneapolis, MN 55454

Chris Lintott
Department of Physics, University of Oxford, Oxford OX1 3RH

ABSTRACT

Abstract goes here.

Keywords: editorials, notices — miscellaneous — catalogs — surveys

1. INTRODUCTION

In the context of Computer Science and/or Astrophysics?

Galaxy evolution. We want to study all the things and sometimes you need some morphologies to really probe how dat galaxy be living its life. So you gotta get dem morphologies. Bolster this with a couple of examples of major insights which have stemmed directly from morphological studies. Make morphology fucking important!
Once the audience believes it's important AF, inform them on the various ways we acquire said morphologies along with pros and cons.

1. Old Skool style: Visual classification (Edwin Hubble, graduate students, Galaxy Zoo)

2. Automated methods: Simplistic methods like Gini-M20, C-A, MID, etc. Cite all the papers (Lotz, Concelise, Freeman, etc.).

3. Automated methods (part 2): Sophisticated methods like machine learning shits (Huertas-Company, Deilman, etc.)

Now that the audience has been briefed on collection methods, explain to them why NONE of them will work when Euclid, WFIRST, and LSST go online:

1. visual classifications: too many to visually classify
2. auto1: accuracy isn't great and you only know morphologies in a statistical sense; can' t do fine detail
3. auto2: need really large training sets; can't apply neural net trained on SDSS to LSST (resolution / pixel size / other image issues)

NOW the audience understands the full problem. Outline our solution for a HYBRID method utilizing both human eyes for the detail work and machines for the broad/bulk classification. Put in the context of Galaxy Zoo (but explain overall method should be modular enough to be used with other formats for visual classification)

With all that said, start the paper! Section blah will be the components of the method. Section blah will be detail about post-processing visual classifications. Section blah will be about the machine algorithm. Section blah will be testing the method in various circumstances. Section blah will be results. Section blah will be Discussion/Conclusions. What sections do we want?

Outline 1:
overview of method the "collaborative filter"
post-processing of human classification (SWAP)
machine classification algorithm
simulation
describe GZ2 classification data describe how the simulation(s) were run – case studies (i.e. smooth/not)
simulation results?

2. OVERVIEW OF THE METHOD?

Figure 1 shows a simple schematic of a single collaborative 'filter' which demostrates the flow of classification data in our hybrid system. A human classifier is shown an image pulled randomly from a pool of images to be classified. She provides her input in the form of a simple yes/noanswer to a *single* question. Simultaneously, several other human classifiers are responding to their randomly assigned galaxy image. These binary responses are then used to both assess the quality of the human classifier's vote as well as the probability of their assigned galaxy to contain the characterstic of choice. If any galaxy obtains a probability that crosses either the upper threshold (thus having a very high probability of exhibiting the feature of interest) or the lower threshold (thus having a very high probability of NOT exhibiting the feature of interest) it is then passed to the machine classifier. Those galaxy images which do not cross either threshold are sent back to the pool to be further examined by human eyes. Those that move on to the machine classifer become a well defined training sample on which the machine learns. The fully trained machine is then applied to the remaining galaxy images in the pool. Those that the machine classifies with high confidence are set aside; the rest remain in the pool for additional scrutiny by human classifiers. In practice we have only run a single collaborative filter at a time, however these could be run in series or parallel; each with its own machine classifier. We now examine each stage of the process in detail.

3. GOTTA TALK ABOUT THE DATA

Our simulations utilize original user classifications from the Galaxy Zoo 2 database (KyleWillett). This dataset is characterized as follows. Each user-provided classification follows a decision tree (show a picture of this probs). Each classification consists of answers to several tasks whereby the tasks are determined by answers to the previous task. All users answer task 1 and task XX. Answers to each task vary in complexity from a binary choice in the case of Task XX: is the galaxy edge on? to several possible choices in the case of Task XX: dominance of galaxy bulge. In total, the data consist of XX milltion full galaxy classifications on 285K unique galaxy subjects. These classifications are composed of 16 million unique answers to X number of tasks.

4. POST-PROCESSING OF HUMAN CLASSIFICATIONS

4.1. *i.e., commandeering SWAP for our nefarious purposes*

In order to reach classification concensus, Galaxy Zoo requires a large number of independent votes for each galaxy. Typically this number is around 40 classificactions per galaxy. Once a particular project is complete, GZ team scientists perform a weighting of user votes once the project is complete (see KyleWillet/Brooke Simmons for description of this process) [Describe in two sentences.] While This process reduces input from malicious users and 'bots' from the overall classification vote; it doesn't up-weight consistent / correct users. Furthermore, waiting until project completion isn't an efficient use of user input (cite the F out of this? and demonstrate with my own shits, obvs!).

GZ:EXPRESS instead tracks user statistics while the project is live by adapting software originally created for the Space Warps Zooniverse project (Phillip Marshall all the way). The Space Warps project demonstrated an efficient technique for finding lenses in galaxy images from CHTFTSLS. The core of the software, Space Warps Analysis Pipeline (SWAP), used Bayesian statistics to compute the likelihood of an image to contain a lens. Additionally, the software assigned an agent to each user. The agent continually updated a confusion matrix for each user tracking the probability that a particular user would correctly identify a lens or not.

For our simulations we have adapted SWAP to run on a regular timestep (in our simulations we've chosen t=1day). At each timestep, all new user classifications since the previous timestep are processed. For each vote, SWAP updates that user's agent's confusion matrix as well as the probability of that image to contain the characteristic of choice (i.e., Featured). Because of the nature of the agent confusion matrix, users that are correct more often than not contribute significantly more than users who are not consistent. Additonally, due to the nature of Bayesian shits, malicious users can contribute just as much as "good" users because SWAP "takes the opposite" of their response as the truth for that galaxy. Users which are neither good nor malicious contribute to a galaxy's probability relatively less.

Each user's vote contributes some fraction to the probability that a galaxy image contains a characteristic of choice. The prior probability for all galaxies is determined by an educated guess for the relative frequency of that characteristic (elliptical galaxies comprise approx. 30% of the local universe). As users vote yes/no, this probability is updated; decreasing if the user votes nay and increasing if yay. An important aspect of this software is that care must be taken when choosing the classification thresholds. The original SWAP set the rejection and acceptanace thresholds equidistant in log-space because their prior was significantly small to begin with (lenses are expected to be very rare; the probability for finding one in a random sky image was estimated at p = 2e-4). Spiral arms or elliptical galaxies are not rare galaxy characteristics; thus determining proper thresholds for retiring an image from the pool must considered lots and I actually haven't done that whatsoever. I just picked some numbers that seemed pretty sweet.

As we show in Fig XXX, the value of the thresholds significantly changes the number of subject images

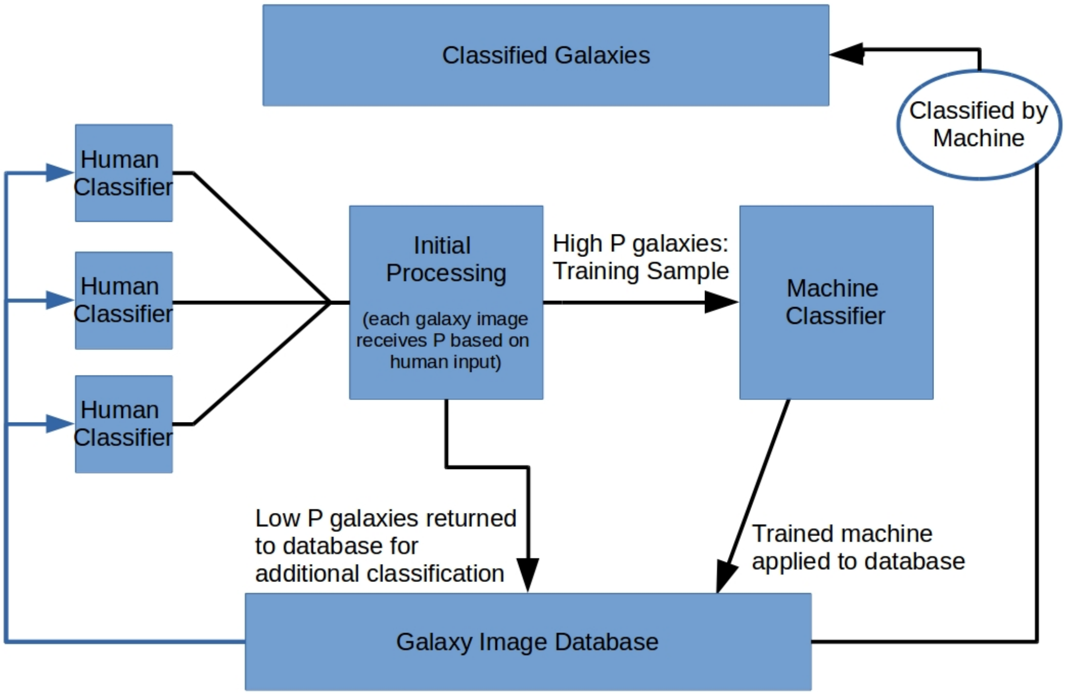
Classified Galaxies
Classified by Machine
Human Classifier
Human Classifier
Human Classifier
Initial Processing
(each galaxy image receives P based on human input)
High P galaxies: Training Sample
Machine Classifier
Low P galaxies returned to database for additional classification
Trained machine applied to database
Galaxy Image Database

Figure 1. Schematic of our hybrid system. Human classifiers are shown images of galaxies via the Galaxy Zoo web interface. These classifications are recorded and processed according to section XXX. As a result of the processing, those subjects whose probabilties cross the classification thresholds are passed to the machine classifier as a training sample. The trained machine is then applied to the remaining subjects in the database (test sample). Those subjects which the machine classifies with high confidence are removed from the sample and considered fully classified. The rest remain in the database to be seen by human classifiers.

which then move on to the next stage.

Anyway. In SWAP they were called ”rejection” and ”acceptance” thresholds. Here we interpret these differently – we aren’t rejecting anything, simply acknowledging that it’s probability strongly suggests that it does not have the characteristic of interest. Thus, we instead consider these ”classification” thresholds. Any subject image which crosses one of these thresholds is then passed on to the next stage.

5. MACHINE CLASSIFIER

5.1. *Supervised Learning*

Supervised learning is the machine learning task of inference from labeled traning data. The training data consist of a set of training examples, consisting of an input vector and a desired output (or label). In general, a supervised learning algorithm analyzes the training data and produces an inferred function that can then mapped to new examples. An optimized algorithm will correctly determine class labels for unseen data. In the case of a collaborative filter, the task requires designation of an image to either have a feature of interest or not and is thus a discrete, binary task. *can I dig up any references which show support for binary tasks being easier for machines to learn?* As such, there are a wide variety of algorithms which are suitable and we discuss our initial choices later on.

5.2. *Training and Validation Samples*

Any supervised learning algorithm requires some form of input and associated label upon which the classifier will attempt to learn the appropriate characteristics which identify a subject as, in our case, either having the feature of interest or not. In order to learn a model which will generalize well to unseen examples, one should provide the largest, most representative training sample possible. In our case, however, this is not immediately possible as the size of the training sample is highly dependent on the rate of human classifications. During the initial stages of the project, human classifiers have not yet provided adequate information to generate sufficiently large training examples. The size of the training sample contains exactly zero subjects at the first timestep but continually grows as human classifications are processed. With a timestep of T and the

SWAP parameters XXX discussed above, we achieve a training sample of about 10K subjects within a 7 days of user classifications. Figure XXX shows how the sample size grows as a function of time for various timesteps and parameters of such and things.

Discuss how this affects the accuracy of the machine classifier.

Discuss the validation set – Nair / GZ2 / Expert user classification set

5.3. *Feature Representation and Pre-Processing*

Machine learning algorithms require a feature vector for each training example. This vector is composed of D individual numeric quantities associated with the subject which the machine will use to discern that subject from others in the training sample. These features can be composed of any set of parameters which inform or corrolate with individual galaxy morphology. Good examples include various color metrics, Sersic index, and B/T ratio, among others. Though we plan to incorporate some of these characteristics in the future, our current feature list is composed of well-known morphologicial parameters including Concentration, Asymmetry, Gini, M20 and ellipticity in a similar fashion as ZEST. In essence, we construct the feature vector based on measurements of the individual image pixels.

This method is quite distinct from *deep learning* methods which learn on every pixel of an image through a series of non-linear transformations of the image as opposed to features derived from the galactic light profiles as discussed in Huertas-Company and Company. However, one drawback to these methods is the difficulty of interpretting the results of such classification. While features like CAS, G-M20 are well demonstrated to correlate with specific galaxy stuffs like SFH and ... whatnots.. and described below.

Concetration measures the

Asymmetry is ...

The Gini coefficient is...

M20 is ...

We measure these parameters for 272K galaxy cutouts in the SDSS shits. Quality characteristics are described in Appendix A.

Combined, these features describe a 5-D feature space in which the machine attempts to do its thang.

Before we feed the algorithm with these feature vectors we first perform two pre-processing steps. First, we must clean the data as there are some very few number of cases where our algorithm failed to recover appropriate values for the Petrosian radius, C, A, G, or M20. Our code represents these failures as infs and we thus remove these subjects from all samples. The second transformation puts each of the features on equal footing. Taken at face value, each of the five morphology parameters resides in a different range of values: M20 is nearly always negative as it is logarithmic while Asymmetry and Gini are always between 0 and 1. In order for the machine classifier to treat all features equally we scale each feature along columns. If a row represents an individual subject and its corresonding unique features, then a column represents the same feature for all subjects. We normalize each subject's feature in the standard way:

$$z_{feature} = \frac{f_i - \mu}{\sigma} \tag{1}$$

Where f_i is the ith subject's feature value, μ is the mean of the entire feature sample, and σ is the standard deviation of the entire feature sample. This scales each feature to values between 0 and 1.

5.4. *Algorithms*

5.4.1. *K-nearest neighbors*

One of the simplest algorithms to conceptualize and use is the K-nearest neighbors (KNN) classifier. This classifier works simply by considering the K neighbors nearest to the test point in Euclidean space. As we have only a binary classification challenge, the test point is classified according to the majority of its K neighbors. In the event of a tie, the label if the closest neighbor to the test point determines its class. Though simplistic, this algorithm is surprisingly powerful in that it performs well in higher-dimensional space and is relatively fast to train. Obviously, the most important parameter to consider is the number of nearest neighbors to assign to each test point. But this is pretty much the only knob to turn in this method which is another benefit – ease of interpretation.

We use scikit-learn's implementation of K-Nearest Neighbor Classifier (KNC) and optimize the K parameter.

5.4.2. *Random forests*

Random forests (RFs) are an ensemble classifier in that they take an average of several individual classifiers, in this case, decision trees. The training data is bootstrapped (sampled with replacement) and a decision tree is performed on each sub-sample. The resulting classifications of each decision tree are combined. A decision tree

This model requires optimization of several parameters including the number of trees, the depth of each tree and BLANK.

Again we use scikit-learn's Random Forest implementation.

5.5. *Cross-validation*

Cross-validation goes here?

Another fundamental characteristic of any machine learning algorithm is the various parameters used to guide the training of the machine. For example, the number of k neighbors in a k-nearest neighbors algorithm; or the depth, d, of a tree in a Random Forest algorithm. These parameters cannot be chosen a priori as they must be tuned to reflect the nature of the classification and to optimize the accuracy of the output. Ideally, one would train a machine classifier with every possible combination of parameters and test the resulting accuracy of each combination against another sample withheld from the training set so as not to bias the results. For this, we choose a subset of the GZ subject sample designated as a validation sample which we describe in detail in the next section.

At each time step, the machine classifier will have some some number, N, of training galaxies. The machine is then trained using several parameter sets (enough to sample the parameter space) and each trained machine is applied to the validation set. The resulting accuracy, completeness, and contamination are recorded. This process repeats at each time step until the machine has achieved a large enough training sample thus allowing the machine to reach its maximum accuracy. Only once the machine has reached its peak performance is it finally applied to the test set (remaining galaxy images in the pool).

5.6. *Machine Output?*

Along with a prediction, the machine also returns an associated probability (discussed below). If this probability, or confidence, is above some threshold (the machine threshold, not to be confused with the post-processing thresholds), then those galaxy images are considered classified and

5.7. *modularity?*

we used a super simple algorithm for this but guess what? If you structure your code a certain way, we can modularize and insert your favorite machine. Additionally, I'd really like it to be able to run multiple machines simultaneously.

APPENDIX

A. MEASURING MORPHOLOGICAL PARAMETERS ON SDSS CUTOUTS

So we did a LOT of work to measure all that shit.

$$I = \frac{1}{1 + d_1^{P(1+d_2)}} \tag{A1}$$

Appendix tables and figures should not be numbered like equations. Instead they should continue the sequence from the main article body.

B. AUTHOR PUBLICATION CHARGES

Finally some information about the AAS Journal's publication charges. In April 2011 the traditional way of calculating author charges based on the number of printed pages was changed. The reason for the change was due to a recognition of the growing number of article items that could not be represented in print. Now author charges are determined by a number of digital "quanta". A single quantum is 350 words, one figure, one table, and one enhanced digital item. For the latter this includes machine readable tables, figure sets, animations, and interactive figures. The current cost is $27 per word quantum and $30 for all other quantum type.

REFERENCES

Corrales, L. 2015, ApJ, 805, 23

Hanisch, R. J., & Biemesderfer, C. D. 1989, BAAS, 21, 780

Lamport, L. 1994, LaTeX: A Document Preparation System, 2nd Edition (Boston, Addison-Wesley Professional)

Schwarz, G. J., Ness, J.-U., Osborne, J. P., et al. 2011, ApJS, 197, 31

Vogt, F. P. A., Dopita, M. A., Kewley, L. J., et al. 2014, ApJ, 793, 127