Draft version June 11, 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

Phil Marshall

ABSTRACT

We implemented one of the first human-machine combos by running a kick ass simulation on previous citizen science data in conjunction with machine algorithms. And guess what? We can obtain at least an ORDER OF MAGNITUDE improvement in the efficiency of classification. So we got that going for us. Which is nice.

Keywords: editorials, notices — miscellaneous — catalogs — surveys

1. INTRODUCTION

The age of Big Data is upon us. Has been upon us. The astrophysics comnunity is already shifting focus, preparing for the way in which our science will change and the way in which we perform our science will chagne. Look at the new CasJobs – This is the type of shit we need: where analytical tools are integrated at the source of the data repository. Downloading datasets is a thing of the past. you can't do Big Data science if you have to constantly move dem data around.

Another area we need to get ready for is how we label all that shit in the sky. We absolutely love labelling things and it's damn necessary too! And the more sky we see both in terms of area and depth is going to grow huge AF. We need to find efficient, clever ways of picking out transients, radio shits, gravitational lenses, galaxy morphology, ... make a really big list with things that are rare or common or time-domain-y. LSST, Euclid, WFIRST are going to swamp us.

In this paper we consider the particular problem of galaxy morphology. This challenge is actually several combined because it necessitates the need to identify the mundane from the unique or rare and, ideally, requires an incredible amount of detail in order to withdraw useful science. Additionally, morphology is a great place to start because we can already begin to plan for the future by considering the Data of Today. The imaging techniques of future surveys will change mostly in resolution and depth; things we can account for.

Another great reason to use morphology as an example is that we can draw on vast, well-established citizen science projects which have contributed to several past publications and have lead to serenditious discovery on multiple occasions. There is no doubt that to spurn this resource would be a disservice to science!!!!

So then. Morphology it is. And don't think that morphology is just a waste of time either. While there is certainly always room for improvement in our classification system including the fact that our categories were made up 100 years ago and only work for the local universe... putting galaxies into categories helps us learn about the way dem galaxies be living their lives.

The idea of combing human and machine classifications IS NOT NEW. That shit's old AF and a big topic of study in computer science circles; circles we astronomers have never been invited to but of which we should still be aware. **Citations from Chris go here!** So this idea is not novel. What IS novel is one of the first practical applications and the ability to explore the repercussions of such a system by simulating various outcomes on previously collected data.

In this paper we consider visual classifications from both citizen scientists through the use of Galaxy Zoo data as well as expert visual classifications from various published catalogs as well as visual classifications from within our own team. We will combine these with various parameters which originally sought to automatically classifiy galaxy morphology. parameters like the

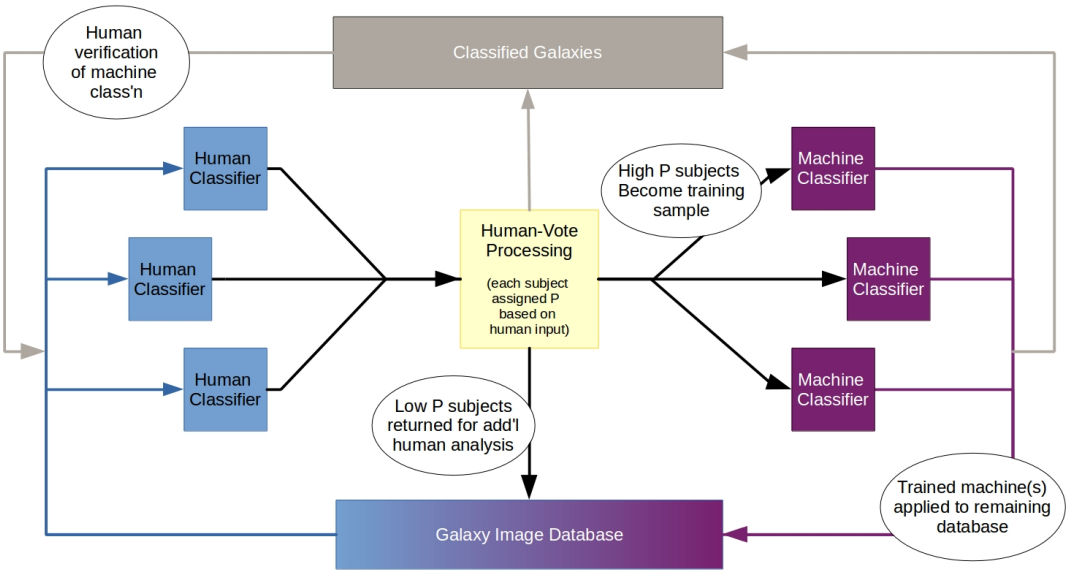

Figure 1. Schematic of our hybrid system. Human classifiers are shown images of galaxies via the Galaxy Zoo web interface. These classifications are recorded and processed according to section XXX. As a result of the processing, those subjects whose probabilties cross the classification thresholds are 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.

Gini coefficient, M20, CAS, etc. We'll wrap this all up in a neat little package by throwing it all in the supervised machine learning algorithm black box which I'll actually explain. And out will pop some sweet classifications!

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?

2. OVERVIEW OF THE METHOD?

Any system combining human and machine classifications will have a set of generic features which we must replicate.

First, a set of humans willing to classify data on request. We will simulate this using a database of classifications from the Galaxy Zoo project which we can draw on at will. These classifications are processed by a Bayesian code first developed for the Space Warps project (SWAP).

Secondly, we need a machine classifier; for this project, we have developed a random forest classifier using easily measured physical parameters such as CAS and Gini as input. See Section X for details.

Thirdly, we will need to make decisions about how the two sets of classifications are combined. After a batch of (human) classifications is processed, then the machine will be trained and its performance assessed against a validation sample. This process is repeated and the machine will grow in accuracy as the size of the training sample increases. Once the machine reaches some acceptable level of performance it is run against the remaining galaxy sample. Images reliably classified by machine are not further classified by humans.

Even with this simple description, one can see that classification will proceed in three phases. At first, the machine will not reach the acceptable level of performance and the only galaxies retired from classified are those for which human classifiers have reached consensus. Secondly, the machine will rapidly improve and both human and machine classifiers will be responsible for image retirement. Finally, improvement in the machine performance will slow, and the remaining images will need to be classified by humans. Working in this allows even moderately successful machine learning routines to be used alongside human classifiers and removes the need for ever-increasing performance in machine classification.

3. GALAXY ZOO 2 CLASSIFICATION DATA

Our simulations utilize original classifications made by volunteers during the GZ2 project. These data are described in detail in (Willett et al. 2013) though we provide a brief overview here. The GZ2 subject sample was designed to consist of the brightest 25% (r band magnitude < 17) of resolved galaxies residing in the SDSS

North Galactic Cap region from Data Release 7 and included both subjects with spectroscopic and photometric redshifts out to $z < 0.25$. In total, 285,962 subjects were classified in the GZ2 Main Sample catalogs (reference website?). Of these, 243,500 have spectroscopic redshifts while 42,462 have only photometric redshifts.

Subjects were shown as color composite images via a web-based interface wherein volunteers answered a series of questions pertaining to the morphology of the subject. In terms of GZ2, a *classification* is defined as the total amount of information about a subject obtained by completing all tasks in the decision tree. A *task* represents a segment of the tree consisting of a *question* and possible *responses*. With the exception of the first task, subsequent tasks were dependent on volunteer responses from the previous task creating the decision tree as shown in Fig **??**. In total, the data consist of over 14 million classifications from 83,943 individual volunteers.

Our first simulated run considers only the first task in the decision tree: 'Is the galaxy simply smooth and rounded, with no sign of a disk?', to which possible responses include 'smooth', 'feature or disk', and 'star or artifact'. Because all volunteers see the first task, our simulations are run with as many as 14,144,941 classifications. The SWAP software requires that each classification consist of at least volunteer ID, subject ID, timestamp of the classification, and the volunteer's vote.

4. POST-PROCESSING OF HUMAN CLASSIFICATIONS

Galaxy Zoo decision trees require a large number of independent classifications for each subject where this value is typically set at forty individual volunteer classifications. Once a project reaches completion, GZ team scientists down-weight inconsistent and unreliable volunteers while the vast majority of volunteers are treated equally with no up-weighted volunteers. While this process reduces input from malicious users and 'bots' from contributing to the consensus, it doesn't reward consistent and correct volunteers. Furthermore, waiting until project completion doesn't allow for efficient utilization of super-users, those volunteers who are exceptional at classification tasks. [Do I need to cite something here?]

Instead, GZ:EXPRESS employs software adapted from the Space Warps Zoonivere project (Marshall et al. 2016) which searched for and successfully found several gravitational lens candidates in the CFHT Lensing Survey (cite XXX). Dubbed SWAP (Space Warps Analysis Pipeline), the software predicted the probability that an image contained a gravitational lens given volunteers' classifications as well as their past experience. While full details can be found in Marshall et al. (2016), we briefly outline the method here.

The software assigns each volunteer an *agent* which interprets that volunteer's classifications. Each agent assigns a 2 by 2 confusion matrix to their volunteer which encodes that volunteer's probability of correctly identifying feature 'A' given that the subject actually exhibits feature A. The confusion matrix also encodes that volunteer's probabiliy of correctly identifying the absense of feature A (denoted as N) given that the subject does not exhibit feature A. The agent updates these probabilities by estimating them as

$$P(\text{“}X\text{”}|X, d) \approx \frac{N_{\text{“}X\text{”}}}{N_X} \tag{1}$$

where $N_{\text{“}X\text{”}}$ is the number of classifications the volunteer labeled as type X, N_X is the number of subjects the volunteer has see that were actually of type X, and d represents the history of the volunteer (all subjects they have seen). The software employs two prescriptions for when the agent updates the volunteer's confusion matrix. In *Supervised* mode the probabilities are only updated after the volunteer identifes a training subject, i.e., one which the scientest knows the correct label *a priori* while the volunteer does not. In *Supervised and Unsupervised* mode, the agent updates the probabilities after every subject the volunteer identifies.

In addition to agent probabilities, each subject begins with a prior probability that it exhibits feature A: $P(A) = p_0$. When a volunteer makes a classification C, Bayes' Theorem is used to derive how the agent should update the subject's prior probability into a posterior:

$$P(A|C) = \frac{P(C|A)P(A)}{P(C|A)P(A) + P(C|N)P(N)} \tag{2}$$

where this value can then be calculated using the elements of the agent's confusion matrix. Marshall et al. (2016) show that perfect volunteers (i.e., those with $P(\text{“}A\text{”}|A) = 1.0$ and $P(\text{“}N\text{”}|N) = 1.0$ would calculate the posterior probability of the subject to be 1.0 which is not surprising (perfect classifiers are perfect!). However, they also show that *obtuse* classifiers (those with $P(\text{“}A\text{”}|A) = 0.0$ and $P(\text{“}N\text{”}|N) = 0.0$ also produce a posterior probability of 1.0; demonstrating that obtuse volunteers are just as helpful as perfect volunteers.

As the project progresses, each subject's prior probability is continually updated and is nudged to higher or lower probability depending on volunteer classifications. Eventually most subjects cross a classification threshold which define wthether that subject has been confirmed or rejected for exhibiting feature A and the subject is considered to be retired. The software no longer records volunteer information for these subjects.

4.1. *SWAP Requirements*

To simulate a live project we run SWAP on a regular timestep which we set as $\Delta t = 1$ day. At each timestep,

the software pulls from the database all volunteer classifications within that range. Before the simulation can be run, a number of parameters which control the behavior of SWAP must first be chosen. These include the initial confusion matrix assigned to each volunteer, the classification thresholds and the prior probability of the subject. Specifically, we must choose

- $P_{S,0}$, the initial probability that a volunteer identifies a subject as being 'Smooth', $P_0(\text{“}S\text{”}|S)$
- $P_{N,0}$, the initial probability that a volunteer identifies a subject as being 'Not Smooth', $P_0(\text{“}N\text{”}|N)$
- p_0, the prior probability of a subject to be 'Smooth'.
- t_s, the threshold defining the minimum probability for a subject to be classified 'Smooth'
- t_n, the threshold defining the maximum probability for a subject to be classified 'Not Smooth'

Additionally, one must choose whether to run in Supervised mode or Supervised and Unsupervised mode. We explore the outcomes of these choices in the next section.

Finally, the SWAP software requires a subset of images to be used as a training sample for the volunteers. Space Warps accomplished this with a collection of simulated lens images where a portion were designed to contain lenses while the rest did not. The original GZ2 project did not employ a training mechanism though we attempt to simulate this process by selecting a subsample of the SDSS imaging that overlaps the Nair & Abraham (2010) catalog. This catalog contains $\sim$ 14K galaxies expertly classified by eye into TTypes along with detailed annotations of a slew of features. We select subset of the overlapping population most representative of the 'Smooth'/'Not' question. We choose only those galaxies with TTypes < -2 (for 'Smooth') and TType $>= 1$ as being 'Not'. We thus exclude S0/a galaxies to remove ambiguity from our training sample.

Where should the above conversation go? We need to talk about training samples but we have two different ones and I use them both ...

We perform several simulations to explore SWAP performance compared to the original GZ2 project in terms of overall accuracy achieved and, perhaps more importantly, in terms of time efficiency. Thus, to evaluate SWAP performance we consider two basic metrics: the cumulative sum of classified subjects at a given point in GZ2 project time, c_{tot}, and the accuracy of those classifications as compared to the original GZ2 published labels, c_{acc}.

Figure 1 shows how SWAP retirement works as a function of GZ2 project time and compares this output to the original retirement scheme. The blue shaded region represents the cumulative number of retired subjects as a function of GZ2 project time. GZ2 retirement was defined by the number of invididual volunteers who had provided a classification for that subject. This value was set at forty, and more than 99.9% of GZ2 subjects have at least 25 unique classifications. For this figure we use the more lenient definition, that is, we consider a subject retired by GZ2 if it had at least 25 individual volunteer votes on a given day of the project. In yellow and orange are the cumulative number of subjects retired via the SWAP software. It is immediately obvious that by clever and adaptive processing of volunteer classifications we can dramatically increase the speed with which subjects are classified.

In figure **??** we show the accuracy of the SWAP software by comparing the label ultimately assigned via SWAP compared to the published label in the GZ2 Main Sample catalog [Link??]. The red line tracks the accuracy of SWAP as a function of project time. At the start of the project SWAP's accuracy is quite high reaching nearly 100% accuracy. As the project progresses, the accuracy decreases slightly, dropping to about 85%. This behavior may seem counterintuitive when compared to an actual machine learning algorithm which general increases accuracy but is actually easily understood. There are several subjects that volunteers find very obvious to classify, which can be seen as several unanimous votes. Several unanimous votes have the effect of quickly increasing the subject's probability which in turn quickly pushes that subject over the retirement threshold. Subjects which are this easy to classify are more likely to have the same label as they did in the original GZ2 project. On the other hand, there are several subjects which volunteers find difficult to classify as evidenced by a split vote count; nearly half of the volunteers vote one way the other half the opposite. In this case the subject probability is kicked back and forth between the retirement thresholds. Eventually, it could find itself crossing one of them but when we compare this subject's SWAP label with its GZ2 label, they are less likely to match as the volunteers had such a difficult time classifying it in the first place. Thus we see that, over time, the easiest to classify subjects pop out first with the more difficult subjects trickling out later. The easiest are more likely to match GZ2 while the more difficult are less likely to match. These curves are, in part, a function of the parameters listed above and we now turn to a discussion of how this output changes when one or more of these parameters is adjusted.

LOTS of TO DO:

1. I'm comparing SWAP label to the debiased GZ2 label – try it with RAW label

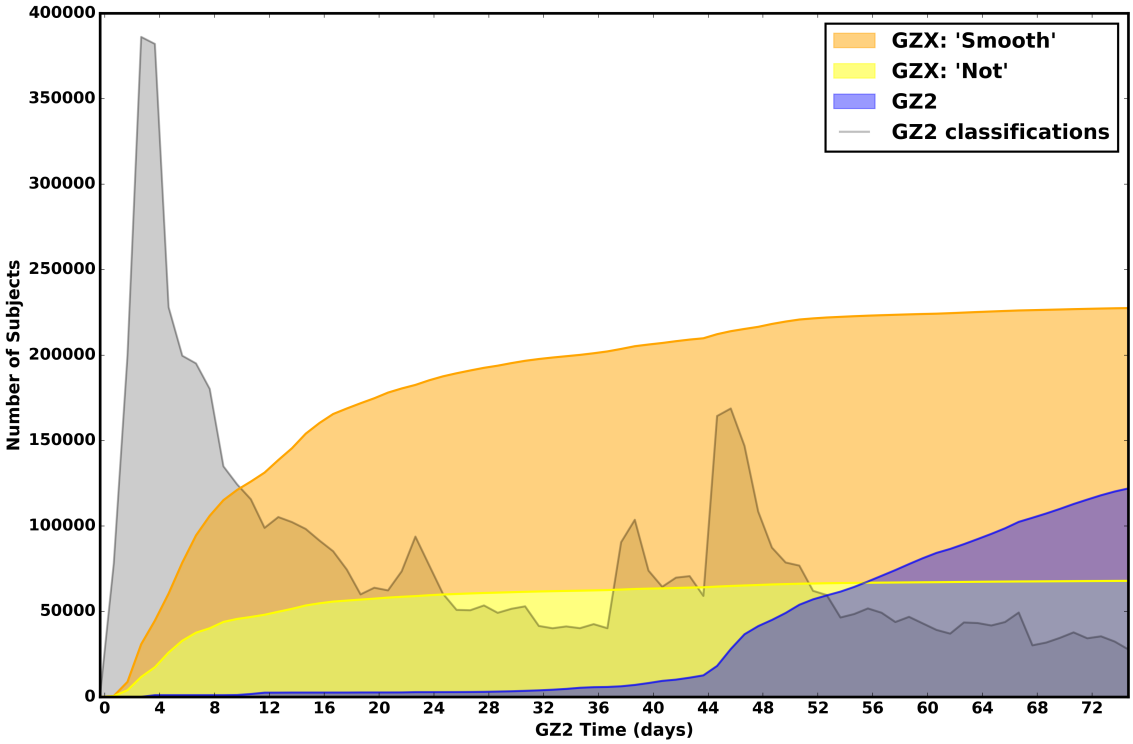

Figure 1. Simulation of top level GZ2 question reprocessed using the SWAP software only. In shaded grey are the actual number of volunteer votes. The blue shows the cumulative number of retired subjects according to the original GZ2 retirement scheme whereby a subject must achieve forty volunteer votes. The orange and yellow shading represents the cumulative number of retired subjects according to the SWAP retirement system, in that subject probabilities must cross an appropriate threshold for being labeled as having a feature or not.

2. I'm comparing SWAP label to the full GZ2 vote but when I show the retired subjects, I'm being generous to GZ2 by allowing GZ2 to only have 25 votes instead of the full 40+ – so how would GZ2's own labels change if they had only 25 votes instead of the full 40?!!

3. Then, obviously, check SWAP against the GZ2[25only] label

4. How do you debias SWAP labels?! It's one thing to downweight shitty users but GZ2 also debiases votes as a function of redshift. How would we do that now???

4.2. *SWAP Simulation Outcomes*

Subject prior probability, p_0. 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) In the case of Space Warps, the prior was simply computed as the number of expected lenses to the number of images in the CFHT Lensing Survey. In the case of galaxy morphologies, this number should represent the probability that a galaxy is elliptical, has a bar, or whatever feature one is considering. This, of course, varies as a function of mass, redshift, star-formation rate, surface brightness, as well as a host of other things depending on the feature in question.

How sensitive are we to significant changes in the prior? We ran several simulations allowing the prior to take on values from 0.01 to 0.9 to demonstrate that the subject prior is not a dominant source of variability. Comparing the output of each run to the original GZ2 results, an order of magnitude change in the prior results in only XXX change in the output accuracy of SWAP. For the remainder of this paper we choose a value of $p_0 = 0.3$.

TO DO: 1. A FEW RUNS WITH DIFF p_0

Initial agent confusion matrix. Space Warps agents assigned each volunteer $P(\text{“}A\text{”}|A), P(\text{“}N\text{”}|N) = (0.5, 0.5)$, assuming that humans started out no better than random classifiers. We explored a range of initial confusion matrix probabilities. In general, we find that regardless of how lowe we initilize $P_{S,0}$ and$P_{N,0}$, GZ2 volunteers all become astute at discerning between 'Smooth' and 'Not Smooth'. When initilizing $P_{S,0}$ and $P_{N,0}$ to lower values it requires more simulation time (measured in GZ2 'days') for them to become good classifiers which affects how quickly subjects can be retired. For example, when $P_{S,0}$, $P_{N,0}$= (0.4, 0.4), XXX subjects are retired by day XXX in GZ2 project time. When $P_{S,0}$, $P_{N,0}$= (0.6, 0.6), that number increases to XXX by the same GZ2 day. In both cases, $c_{acc} \sim 85\%$

TO DO: 2. I did these runs but they were S only – try in S&U? Won't work in that mode unless I'm using "expertsample"

Retirement thresholds. The Space Warps project set their retirement thresholds equidistant in logspace. Their prior was significantly small to begin with as lenses are expected to be very rare. However, 'Smooth' galaxies (which roughly correspond with early-types) are nearly equal to the 'Not Smooth' within a factor of 2 at low redshift. Thus care must be taken when setting t_s and t_n for retirement. The SWAP output is most sensitive to these parameters as they are directly responsible for the label assigned to a given subject. When these thresholds are low, subjects more easiliy attain the appropriate probability to cross that threshold. This can increase speed of classification however it also greatly affects accuracy. If the next volunteer disagreed with the previous volunteer on the nature of the subject, her vote could bring the probability of the subject below the threshold and thus it wouldn't yet be classified. Therefore setting these values is crucial.

TO DO: 3. I haven't don't any real analysis of changing thresholds. So get on it!!!

Summary of this section? Segue into Machine Classifier. Regardless of the parameters with which one begins (within reason) the number of retired subjects grows significantly. GZ2 required at least 48 days before it could retire 50K subjects whereas SWAP can retire that many within 4 days, depending on parameter choice / with some % difference? / something like that. It is instructive to consider the number of volunteer 'clicks' instead of the physical timescale as this is a quanity that spans multiple projects regardless of the number of volunteers that project may have. In this light, GZ2 requires nearly 2.5×10^6 votes to retire 60K subjects while, to match it, SWAP requires only 4×10^5, an ORDER OF MAGNITUDE REDUCTION BITCHES!

5. MACHINE CLASSIFIER

Supervised learning is the machine learning task of inference from labeled training data. The training data consist of a set of training examples, including an input (feature) vector and a desired output (or label). Generally speaking, a supervised learning algorithm analyzes the training data and produces an inferred function that can then be mapped to new examples. An optimized algorithm will correctly determine class labels for unseen data. In general, most classification algorithms can handle prediction of several labels simultaneously. Work has been done to predict the entirety of GZ2 classification labels using deep learning (Dieleman et al. 2015) with great success. However, it is still simpler for a machine to predict fewer labels than it is to predict several dozen. [**citation?**] Fortunately, by handling individual features and processing human classifications through SWAP, we arrive with a discrete, binary task for a machine to tackle. However, in the future we plan to explore more sophisticated algorithms which are optimized to handle a continuum since the actual output of the SWAP software is a probability for any given subject to exhibit a particular feature.

5.1. *Random Forests*

Because our task is simple, we choose a simple machine. In particular, we use a Random Forest (RF) algorithm, an ensemble classifier that operates by bootstrapping the training data and constructing a multitude of individual decision tree algorithms, one for each subsample. An individual decision tree works by deciding which of the input features best separates the classes. It does this by only performing splits on the values of the input feature that most decrease the classification error. These feature splits proceed recursively, always with the goal of decreasing the classification error. As such, decision trees alone are prone to overfitting the training data which precludes them from generalizing well to new data. Random Forests mitigate this effect by combining the output label from the multitude of decision trees. In particular we use the `RandomForestClassifier` from the Python module `scikit-learn` (Pedregosa et al. 2011).

5.2. *Cross-validation*

Of fundamental importance is the task of choosing an algorithm's hyperparameters, values which determine how the machine learns. In the case of a RF, one must choose the maximum depth of the tree, the minimum leaf size, the maximum number of leaf nodes, etc. The goal is to determine which values will optimize the machine's performance and thus cannot be choosen *a priori*. Idealy, one would train the machine with every combination of parameters and consider the resulting performance by testing the trained machine on a sample withheld from the training sample so as not to contaminate the results. Formally, we perform k-fold cross-validation whereby the training sample is split into k subsamples. One such subsample is withheld while the remaining data is used to train the machine. This is performed k times and the average performance value is recorded. The entire process is repeated for every combination of the specified hyperparameter space and the optimal values can be recovered.

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 sub-

ject from others in the training sample. To segregate 'Smooth' from 'Not Smooth' our feature set draws on ZEST (Scarlata et al. 2007) and is composed of Concentration, Asymmetry, Gini, M20 and ellipticity (See Appendix A for details concerning the measurement process). Altogether, these features describe a five dimensional parameter space in which the machine attempts to distinguish the two classes. The RF algorithm is capabale of handling a considerable number of features and in a future paper we will explore increasing this feature space to include parameters like Sersic index, B/T ratio, color, etc.

THIS STEP NO LONGER NEEDS TO BE DONE – RF SHOULDN'T CARE:

TO DO: 4. Check that I'm right! Before we feed the algorithm with these feature vectors we first perform two pre-processing steps. First, we 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 or nans 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, then a column represents the same feature for all subjects. We normalize each subject's features in the standard way:

$$z_{feature} = \frac{f_i - \mu}{\sigma} \tag{3}$$

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. *Training and Validation Samples*

We are now ready to discuss the training sample. As we showed in the previous section, the SWAP software retires subjects much more quickly than the GZ2 project by adaptively tracking volunteer skill and subject probabilities. This provides us with a way of quickly generating considerably large subject samples with accurate labels provided by human classifications. These retired subjets are the basis of the machine's training sample. That training sample is dynamically generated as a function of project time. Within the first week of a project there are perhaps only a few subjects which reach retirement, but as the project progresses, that number soon becomes thousands.

As discussed above, in addition to a training sample we also need a validation sample to estimate the generalization (true) error of our trained machine. For this purpose we maximize the utility of our expertly classified sample. This sample thus provides training to our volunteers and verification for our machine.

5.5. *MachineShop Simulation*

As before, we simulate a live project by running our machine directly after running SWAP and in timesteps of $\Delta t = 1$ day. As the simulation progresses each night, a flag associated with each subject is triggered in the SWAP software if it reaches retirement. This then signals the machine software that it indeed has some amount of training data to consider. For our first run, any subject retired by SWAP is immediately considered part of the machine's training sample, however we do not allow the machine to begin learning until it has reached a minimum number of training subjects. For our first run this number is set to 10K. Once SWAP has reached that number of retired subjects the machine begins learning.

The machine is allowed to learn at each timestep. Thus, the machine classifier will have some some number, N, of training galaxies. K-fold cross-validation is performed at each time step to determine optimal hyperparameters. The machine is then trained with the optimal parameters and is applied to the validation sample. A slew of performance metrics are recorded including the validation accuracy, completeness, and contamination. This process repeats at each time step until the machine has learned all it can. This requires some discussion. While it is true that increasing the sample size will, in general, increase the optimal performance of the machine, this behavior eventually reaches a plateau. In order to judge our machine as being 'fully learned', we implement a learning criterion, which considers the growing history of the machine's learning and judges the machine learned after its accuracy varies by less than 1% over the course of 3 consecutive timesteps. We will discuss how varying this criterion changes the outcome of the classification results.

Once the machine has been fully trained, it is then applied to the test sample. In this case, the test sample is any subject which has either not reached retirement through the SWAP processing, or is not part of the validation sample. Since the total number of subjects in GZ2 is XXX, the validation sample comprises XXX, the initial training sample is 10K, thus the first test sample contains XXX subjects. The test sample decreases as a function of project time in tandem with the increasing training sample. Thus the further into the project one goes, the more trained the machine becomes and the easier time it has predicting the test sample since that sample shrinks.

5.6. *Machine Output*

Once the machine is happily applying itself to the test sample at each timestep, we first simply record its predictions along with the probability associated to that prediction. Nearly every machine classification algorithm can output a confidence score or decision function associated with its prediction. In the case of a RF, this is calculated as **[Look this shit up.]**

In figure **??**, we can see how the machine is performing on the test sample as a function of project time. Understandably, the performance is poor the first time the machine crosses the 'learnned' criterion. Flukes can happen and it was most likely not quite as learned as expected or desired. However, as the project progresses the performance on the test sample increases. In particular, the black line denotes the performance as obtained on the entire test sample and we can see that even when the machine is at its peak, the accuracy doesn't increase much beyond 70%. The red line, however, is when we select only those subjects which the machine is most confident about (more than 90% confidence, in particular). Now the accuracy of the machine on this subsample reaches nearly 85%.

This allows us to set a criterion for which subjects the machine should be allowed to retire from the system. We don't necessarily trust it on all subjects but can we trust it when it's most confident? Explore this more?

5.7. *Feedback Loop*

Our system has now incorporated both clever use of human classifications and integrated machine classification. We now simulate the final element which is the feedback mechanism. Those subjects which the machine is most confident about are flagged as retired and votes on these subjects are no longer recorded in SWAP or elseware. Those which the machine is not confident, however, remain in the pool such that, during the next timestep, volunteers can contribute to the classification of that subject. This will increase the diversity of the training sample thus providing the machine with a larger sample space in which to learn.

We now perform full simulations, culminating all steps in the method and examine the overall performance.

TO DO: Implement and run this. OY! That's a tall order...

6. RESULTS

Results of a FULL RUN – including the feedback loop.

6.1. *Performance #s*

How well does the overall human/machine system perform together and separately.

what are sensible criteria for using the machine? If we change these criteria, how does performance change?

When does the machine kick in? How quickly does it learn?

Efficency of classification increased by order of magnitude.

6.2. *The effect of human training*

this all relies on training humans in addition to training machines.

Fewer users trained

Fewer training images

Less front-loading (how far apart can the training images be staggered and still produce good results?)

This is all qualitative (not fitting functions to anything)

7. SO FUCKING WHAT?

All you did was classify Smooth and Not Smooth. Anyone and their mom can do that!

We've now identified several ways to suss out those subjects which require additional intervention. If SWAP can't classify it, then potentially these subjects should be diverted to experts. If a machine can't classify it, then those subjects can be relegated back to humans. Thus we have a cute little chain of command!

Apply all these performance metrics to the datasets expected from Euclid, LSST, etc. Estimate reduction in classification time.

APPENDIX

A. MEASURING MORPHOLOGICAL PARAMETERS ON SDSS CUTOUTS

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

Concetration measures the ...

$$C = 5\log(r_{80}/r_{20}) \tag{A1}$$

where r_{80} and r_{20} are the radii containing 80% and 20% of the galaxy light respectively. Large values of this ratio tend to indicate disky galaxies, while smaller values correlate with early-type ellipticals.

Asymmetry quantifies the degree of rotational symmetry in the galaxy light distribution (not necessarily the physical

shape of the galaxy as this parameter is not highly sensitive to low surface brightness features).

$$A = \frac{\sum_{x,y} |I - I_{180}|}{2 \sum |I|} - B_{180} \tag{A2}$$

where I is the galaxy flux in each pixel (x, y), I_{180} is the image rotated by 180 degrees about the galaxy's central pixel, and B_{180} is the average asymmetry of the background.

The Gini coefficient, G, describes how uniformly distributed a galaxy's flux is. If G is 0, the flux is distrubuted homogeneously among all galaxy pixels.; while if G is 1, all of the light is contained within a single pixel. This term correlates with C, however, unlike concentration, G does not require that the flux be concentrated within the central region of the galaxy. We calculate G by first ordering the pixels by increasing flux value, and then computing

$$G = \frac{1}{|\bar{X}|n(n-1)} \sum_{i}^{n} (2i - n - 1)|X_i| \tag{A3}$$

where n is the number of pixels assigned to the galaxy, and $\bar{X}$ is the mean pixel value.

M_{20} is the second order moment of the brightest 20% of the galaxy flux.

$$M_{tot} = \sum_{i}^{n} f_i[(x_i - x_c)^2 + (y_i - y_c)^2] \tag{A4}$$

$$M_{20} = \log_{10}(\frac{\sum_i M_i}{M_{tot}}), \quad \text{while} \sum_i fi < 0.2 f_{tot} \tag{A5}$$

REFERENCES

Dieleman, S., Willett, K. W., & Dambre, J. 2015, MNRAS, 450, 1441

Huertas-Company, M., Gravet, R., Cabrera-Vives, G., et al. 2015, ApJS, 221, 8

Marshall, P. J., Verma, A., More, A., et al. 2016, MNRAS, 455, 1171

Nair, P. B., & Abraham, R. G. 2010, ApJS, 186, 427

Pedregosa, F., Varoquaux, G., Gramfort, A., et al. 2011, Journal of Machine Learning Research, 12, 2825

Scarlata, C., Carollo, C. M., Lilly, S., et al. 2007, ApJS, 172, 406

Willett, K. W., Lintott, C. J., Bamford, S. P., et al. 2013, MNRAS, 435, 2835