Chapter 3
Accelerating Image Analysis Research with Active Learning Techniques in Genetic Programming

Nathan Haut, Wolfgang Banzhaf, Bill Punch, and Dirk Colbry

3.1 Introduction

Image analysis is the process of extracting useful information from image data. This extracted information can then be used to study systems captured in the image data. Image analysis is broadly applied from medical imaging to computer vision [11, 14]. In medical imaging, the image data will often come from magnetic resonance imaging (MRI), positron emission tomography (PET), computerized tomography (CT), x-rays, etc. [11]. For example, in [12] the authors are able to use MRI and PET image data with deep learning methods to improve the success rate of identifying Alzheimer's disease. Image analysis involves extracting useful information from image data, which is generally rich with information, but can also contain significant noise. Segmentation is a specific step in image analysis where the features of interest are isolated and background information (noise) is removed.

Active learning is a field in machine learning where data selection is performed to maximally inform model development [2]. Active learning's origins drew inspiration from query learning, which was a method for designing experiments with the goal

N. Haut (✉) · B. Punch · D. Colbry
Department of Computational Mathematics, Science and Engineering,
Michigan State University, East Lansing, MI, USA
e-mail: hautnath@msu.edu

B. Punch
e-mail: punch@msu.edu

D. Colbry
e-mail: colbrydi@msu.edu

W. Banzhaf
Department of Computer Science and Engineering, Michigan State University,
East Lansing, MI, USA
e-mail: banzhafw@msu.edu

S. Winkler et al. (eds.), *Genetic Programming Theory and Practice XX*,
Genetic and Evolutionary Computation,
https://doi.org/10.1007/978-981-99-8413-8_3

of maximizing information gain using statistical measures [8]. In many ways active learning is similar to query learning, the key difference though is the consideration for how the information gained will inform model development of a specific machine learning method. This is why active learning methods vary based on which machine learning method is being used from neural networks to support vector machines [1, 6, 9]. Uncertainty sampling is a method for selecting new training data that maximizes model uncertainty with the idea that the data points the model is maximally uncertain about will provide the most information for model training [7]. Uncertainty sampling has been shown to be an effective method when used with support vector machines, where one intuitive approach is to sample new data points nearest the decision boundary [6].

There are 3 main classes of active learning: pool-based, stream-based, and membership query synthesis [9]. Pool-based methods rely on an existing set of unlabelled data, which the active learning method search to pick one or several data points that contain maximal information to label and add to the training set. Stream-based methods are similar to pool-based in the sense that the unlabelled data exists. The key difference is that the data is not searched but rather fed to the AL method one data point at a time and the AL method either indicates the data be labelled or not based on some predicted information score. Membership query synthesis does not use existing data and instead searches a space to find a training point to both be generated and labelled. In this work, we focus on pool-based methods where we have existing unlabelled data and select training samples one at a time by selecting the sample with the highest predicted information score.

While genetic programming models individually generally lack statistical properties to compute uncertainty, model populations present in GP can be utilized to quantify uncertainty across the diverse models within a population. We have previously applied active learning to genetic programming in symbolic regression tasks using active learning in genetic programming (AL-GP) with a stack-based genetic programming system (StackGP) [4]. In that work we took inspiration from [5] and explored how populations in GP can be exploited to select new training data that the current model population is maximally uncertain about. This is done by selecting a model ensemble of diverse individuals from the population and using the ensemble to find points that maximize the uncertainty to add to the training set. This process proved successful and additional work is being performed to study how additional data diversity metrics can be included to improve the method.

This general strategy of utilizing the model populations in symbolic regression GP tasks to select training data to maximally inform evolution seemed generalizable to GP and evolutionary computation, so in this work, we demonstrate how AL-GP can be implemented and applied to other evolutionary computation methods and various image analysis problems to accelerate the development of segmentation and object classification models. These models can then be used to aid research in various fields reliant on image data.

The SEE-Insight project is an open-source framework to accelerate the biggest bottleneck in Scientific Image Understanding, which is manual image annotation. SEE-Segment [3] is the first tool developed for SEE-Insight and consists of an evolu-

tionary machine learning approach that utilizes a genetic algorithm to select a computer vision algorithm and optimize parameters for an image annotation task (image segmentation). SEE-Segment will work with a Graphical User Interface (GUI) allowing researchers to upload their image datasets and then incrementally annotate their images. The image annotations are used to test scientific hypotheses or as a first step to feeding into a data-driven model such as a neural network. Because annotating images can be slow and tedious, while researchers are interfacing with the GUI the SEE-Segment system is simultaneously searching this grammar (aka "algorithm space") to find automated methods that can reproduce their manual workflows. This search is happening "in the background" on large-scale systems. Given the complexity and size of the search space, there is no guarantee that it will converge to a reasonable solution. However, if a good algorithm is found then suggestions are passed to the researcher to help speed up their annotation process. In the best case, a fully automated algorithm is identified that can reproduce their manual annotation. In the worst case, SEE-Segment will not take any longer than manually annotating images without the discovery tools.

This application is an instance of a Combined Algorithm Search and Hyperparameter (CASH) problem and uses a genetic algorithm to search for an algorithm (and hyperparameters). Although the space is nondifferentiable and highly heterogeneous, preliminary results are promising. By using a well-defined image grammar and genetic algorithms as the core search tool, results of the machine learning are highly human interpretable. This allows the system to "generate code" that can be used for teaching as well as copy-and-pasted to a researcher's own program.

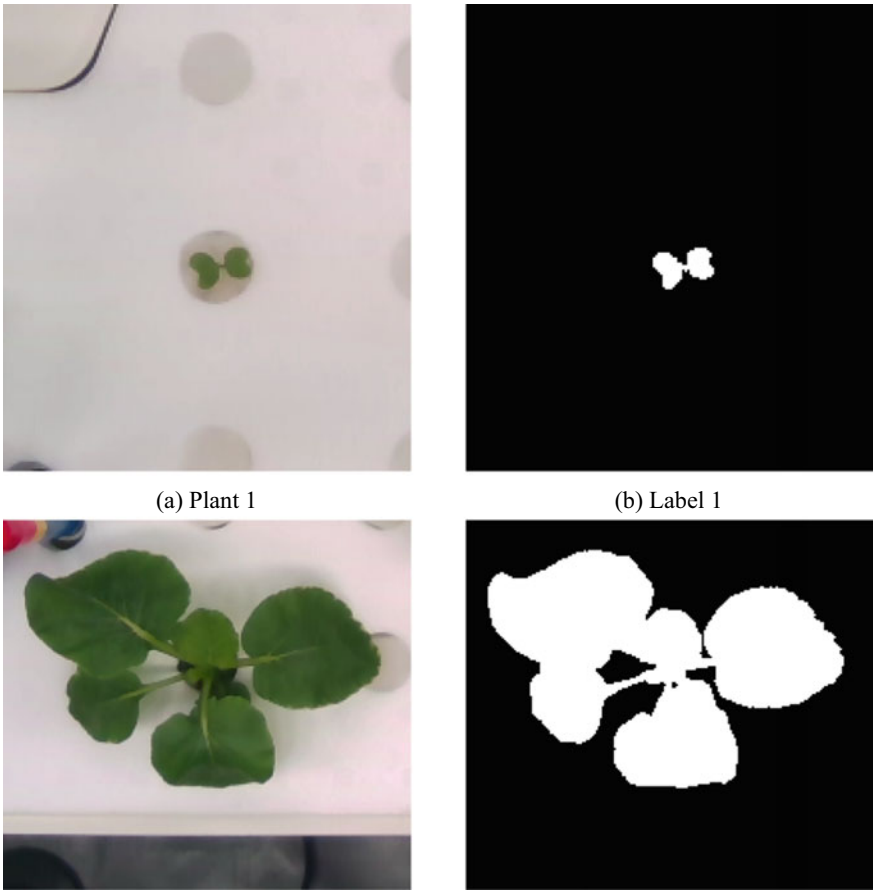
Decision tree GP (DT-GP) is a GP system that evolves decision trees and was developed as part of this work specifically to solve the problem of cell classification but is generalizable to any classification task.

In this work, we explore the efficacy of AL-GP in two different population-based ML systems and then demonstrate how AL-GP can be applied in a research setting to accelerate progress in scientific studies.

3.2 Data Sets

3.2.1 KOMATSUNA

To benchmark the active learning methods in both systems, the KOMATSUNA [13] data set was used since it is a fairly simple segmentation problem and has ground truth labels available. The KOMATSUNA data set contains 300 images of plants where the provided labels are the segmentation patterns that identify the plant from background data. The KOMATSUNA data set is ordered and tracks plants over time as they grow. Each set of 5 consecutive images is taken within the same day so the images are substantially similar within those sets. Example images from the KOMATSUNA data set are shown in Fig. 3.1 to demonstrate how the sizes of plants vary in the set and also how the camera angle and location can vary.



(a) Plant 1 (b) Label 1

(c) Plant 2 (d) Label 2

Fig. 3.1 Example image **a** from KOMATSUNA dataset with its corresponding label (**b**). The label is the true segmentation mask. A second example image (**c**) and it's label **d** are shown to demonstrate the diversity of images in the dataset

3.2.2 Cell Classification

AL-GP was also applied to cell image data to show how active learning could be applied to the problem of cell segmentation and classification. For this data set, the goal of classification is to determine which of the cells are co-transfected (expressing two proteins of interest). The image data [10] consists of two streams, one that tracks the expression of green fluorescence which indicates the presence of one protein, and another that tracks purple fluorescence, which indicates the expression of the other protein. An example of this data is shown in Fig. 3.2.

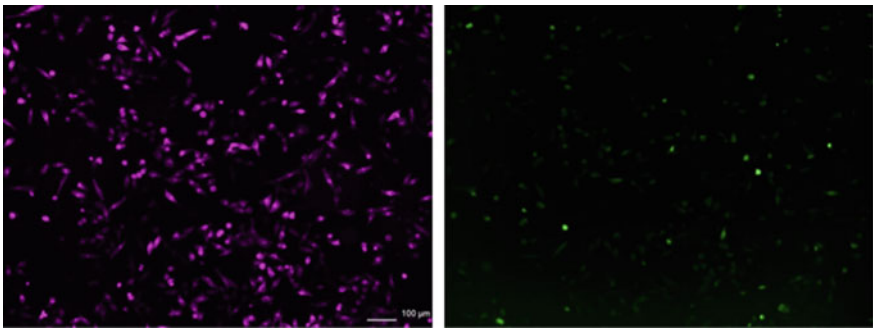


Fig. 3.2 The image on the left shows cells that are expressing a protein with a purple fluorescent piece. The image on the right shows cells that are expressing a protein with a green fluorescent piece

3.3 Active Learning

Active learning was used to iteratively select training samples to maximally inform model populations. Each run begins with a single randomly selected training sample and one additional training sample is selected and added by active learning after a set number of generations. Once the new sample is added, evolution continues on the expanded dataset. Uncertainty was quantified by measuring disagreement between the models in an ensemble. To do this, an ensemble of 10 diverse models is selected from the population. The ensemble of models is then evaluated on every unselected training sample and the uncertainty on each sample is recorded by measuring the average difference between the predictions of each model in the ensemble. The sample with the highest uncertainty value is then selected and added to the training set. This method varies slightly from the original AL-GP approach in that the ensemble sizes are constant and the models are not selected by selecting best-fitting models on different data partitions. This change was made since we begin with a single training sample, which would result in one model selected for the first ensemble. A single model lacks any measure for uncertainty.

3.4 AL-GP Applied to Decision Tree GP

3.4.1 Decision Tree GP (DT-GP)

Image segmentation and object classification were performed using an implementation of decision tree GP (DT-GP), where decision trees are evolved to consider which pixels are foreground or background for segmentation and to identify which class an object belongs to for object classification tasks. The decision trees can utilize 3 types of operators: boolean operators, (in)equality operators, and numerical

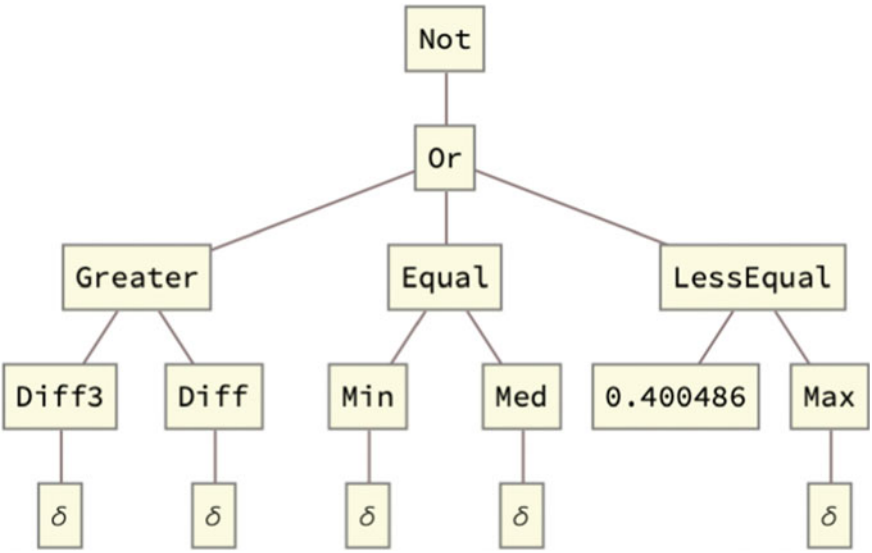


Fig. 3.3 Shown here is an example tree generated to segment the KOMATSUNA data set. Here delta is a placeholder for the data

operators. Boolean operators can take in boolean values and return boolean values. Inequality and equality operators take in numeric values and return boolean values. Numeric operators can take in numeric vectors and return numeric scalars.

The boolean operators available are *And, Or, Not, Nand, Nor,* and *Xor*. The inequality operators available are $\geq$, $>$, $\leq$, $<$, $=$, and $\neq$. The numeric operators are *average*, *median*, *max*, *min*, *difference*, *range*, *standardDeviation*, *getRed*, *getGreen*, and *getBlue*. The last three operators listed simply grab the red, green, or blue values from pixels. The models can also contain randomly generated constants as floating point values from 0 to 1 since the RGB pixel values are stored as values from 0 to 1.

To make the use of these operators meaningful, a hierarchical structure is enforced in the trees such that the boolean operators can only operate on inequality operators, and inequality operators can only operate on numeric operators and constants. Numeric operators are then restricted to the lowest levels of the decision trees where they can operate on numeric data. Trees can only be initialized in this form and then this form is enforced throughout evolution by restricting how mutation and crossover can be performed.

An example of a tree generated to segment the KOMATSUNA plant data is shown in Fig. 3.3. Note that although this tree happens to be balanced, it is not enforced, so trees generated by DT-GP can potentially be heavily skewed.

Subtree crossover, subtree mutation, and point mutation were used as the variation operators. The parameters used to run DT-GP with AL-GP are shown in Table 3.1.

Table 3.1 DT-GP & Active learning parameter settings

Parameter	Setting
Mutation rate	30
Crossover rate	40
Spawn rate	10
Elitism rate	20
Crossover method	Subtree
Mutation method	Point & Subtree
Tournament size	5
Population size	20
Generations	20
Max tree depth	6

3.4.2 Active Learning Implementation

Active learning was used with DT-GP to select training images iteratively to maximally inform model populations. Uncertainty was quantified by measuring disagreement between the models in an ensemble. To do this, each model in the ensemble was used to generate a predicted segmentation pattern on each potential image to be added to the training set. For each image, every pairwise pixel difference was computed for each model in the ensemble's predicted segmentation pattern. This difference can be summarized as the total number of pixels where the two models disagree on its classification. The pairwise differences of all models in the ensemble are then averaged. The image that returns the largest uncertainty value is then selected and added to the training set. Evolution then resumes using the now expanded training set.

The model ensemble is selected in a way that attempts to capture the diversity of the population while also containing primarily high-quality individuals. This is done by selecting the top 10 models from the population that have unique fitness values on the training set.

3.4.3 KOMATSUNA Multi-image Results

DT-GP was tested using the KOMATSUNA dataset to see how active learning impacts its ability to perform on a fairly simple dataset. Active learning was compared to random data selection by recording the test fitnesses after each iteration to see which method improves fitness on the test set quicker and which method arrives at better fitness values. In this case, a fitness of 1 is a perfect model and a fitness of 0 is the worst possible fitness. Each method was tested a total of 40 times with different

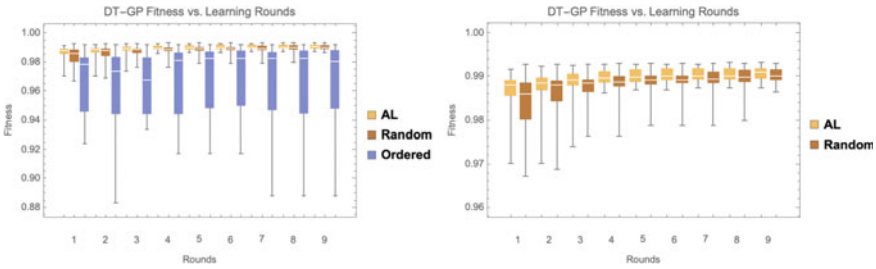


Fig. 3.4 Active learning, random data selection, and ordered data selection using DT-GP are compared with the distribution of test fitnesses shown after each round of learning. Each experiment was repeated 40 times, so each bar represents the distributions from 40 repeated trials. The data shows that active learning outperforms random data selection. The active learning method most quickly increases fitness. The right figure focuses on just AL and random data selection for easier visualization of the same results from the left figure

randomly selected training and test sets of 250 images available to be selected for the training set and 50 images in the test set.

The results of the 40 runs comparing active learning and random sampling are shown in Fig. 3.4. We can see that active learning more quickly improves its test fitness towards 1. We can see though that this is a fairly simple problem for DT-GP since even after just one round of learning the models from both active learning runs and random sampling runs are around 0.98.

3.4.4 KOMATSUNA Single-Image Results

DT-GP was also tested using individual images from the KOMATSUNA dataset where instead of using active learning to select full images to add to the training set we begin with a single pixel and add one pixel each round of learning. We compared active learning and random data selection. As before, we record the fitness on a test set after each round of learning. In this case, the test set is the full image and the training set is the subset of pixels selected by the learning strategy. Both active learning and random selection were tested 40 times. The results of the 40 independent trials for one image are shown in Fig. 3.5. The results show that active learning outperforms random selection by reducing error more quickly and more consistently, as well, we see that the active learning approach arrives at a lower error. The random selection approach seems to have gotten stuck in a local optima since the approach plateaus after around 20 rounds of learning and converges after 30 rounds of learning. The results of a sample run using active learning and random sampling are shown in Figs. 3.6 and 3.7. The figures show the original image, the sampled pixels, and the

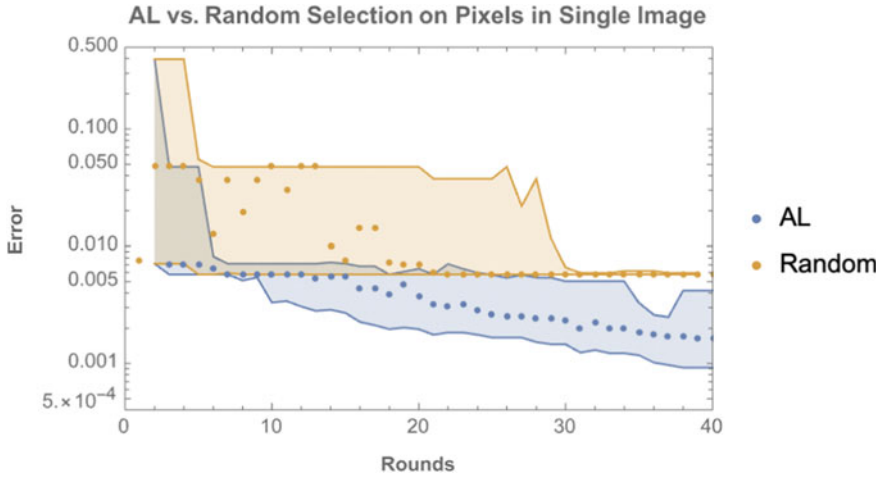


Fig. 3.5 Active learning and random data selection using DT-GP are compared on their ability to learn the segmentation of a single image by selecting one pixel each round of learning. Each experiment was repeated 40 times, so the plot shows the median fitness for each method surrounded by bands representing the upper and lower quartiles. The data shows that active learning outperforms random data selection. The active learning method most quickly reduces error and also does not get stuck on a local optima as the random selection approach does

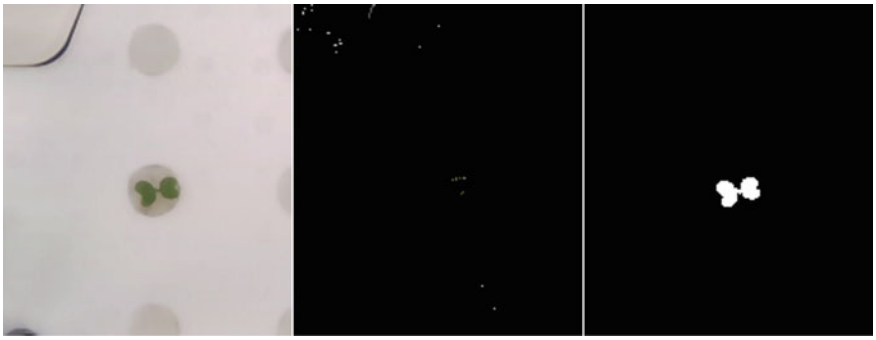


Fig. 3.6 The results of using pixel-based active learning on one of the KOMATSUNA images. The image is shown on the left. In the middle, the 40 sampled pixels are shown, as selected by active learning. On the right, the segmentation pattern is shown, which is nearly perfect

segmentation pattern after 40 points are sampled. The active learning method arrives at a near perfect segmentation pattern while the random sampling approach results in a segmentation pattern that picks up a lot of background and is missing the centre of the plant.

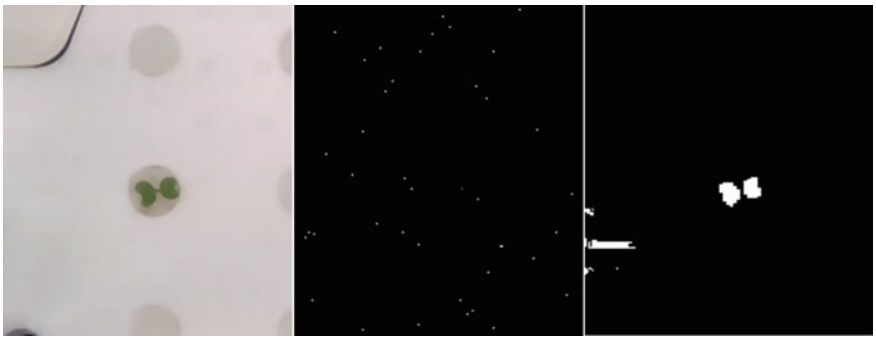


Fig. 3.7 The results of using pixel-based random selection on one of the KOMATSUNA images. The image is shown on the left. In the middle, the 40 sampled pixels are shown, as selected using random selection. On the right, the segmentation pattern is shown

3.4.5 Cell Classification

We applied DT-GP to automate the process of identifying cells in videos and determining which cells are co-transfected to help accelerate a research project where progress is slowed by time spent manually labelling every cell in each video frame. Co-transfected cells are those that express two different proteins of interest that the researcher instructed the cell to produce by inserting the genetic information into the cells. The task of labelling cells in images is currently done manually and occupies a significant amount of time since it requires cells to be outlined in each video frame as well as requires analysis of a second set of images to determine which of the cells in the video stream are co-transfected. This makes it an ideal task to be automated since automation could lead to significant time savings and enhance research productivity.

The goal of this project is to utilize a GP system with an active learning strategy to develop models that can automate the process of identifying cells and labelling them as co-transfected or not. Two types of models are developed. The first being a model that can correctly identify all the viable cells in the image. The second type of model identifies which of the cells are co-transfected. The goal of applying active learning is to only require a few cells be labelled and then the developed models would be able to correctly select and classify the rest of the cells in that frame as well as additional frames if needed. Once all of the cells are labelled the researcher can focus on the analysis of the data sooner.

The first stage of modelling classifies pixels as either being part of a cell or being background. The second stage then identifies if a cell is co-transfected or not by analyzing the two different images from different filters. Using the results from identifying which points are part of a cell or not, a clustering algorithm is used to associate connected points and identify them as individual cells. An active learning strategy was implemented to guide labelling of the cells. This is done by presenting the user with points or cells that have maximal disagreement between

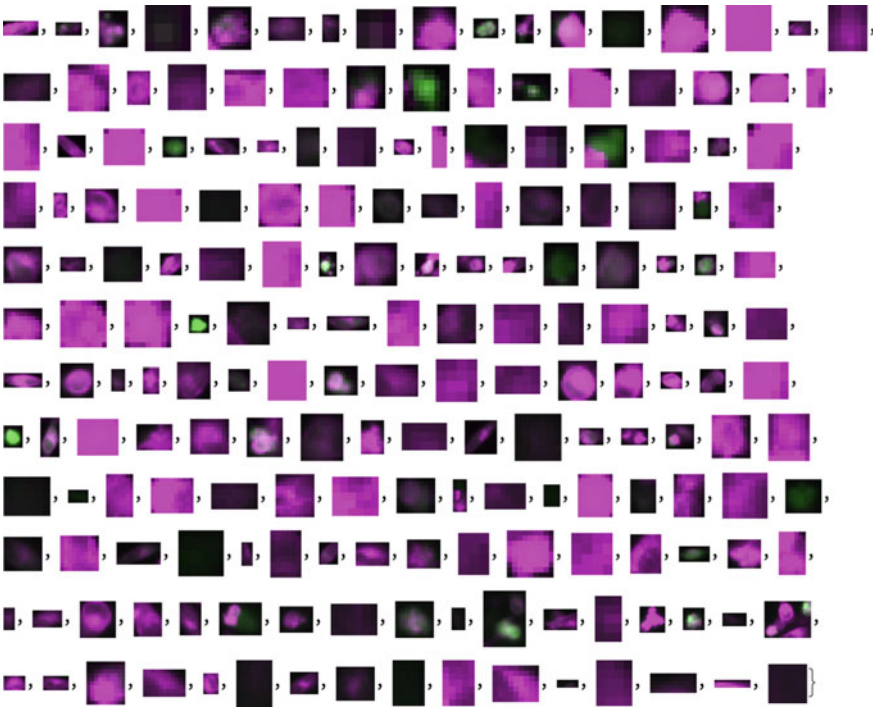


Fig. 3.8 These are the cells identified as co-transfected by the model developed with the GP system. Each box is supposed to fully encapsulate a co-transfected cell. Both the green and purple images have been combined here to show how both colors are present in the images

their classifications amongst the models in an ensemble. The user can then provide a true label for the cell or point. That cell or point with the label is then added to the training set.

Figure 3.8 shows some results where a developed model was tasked with identifying all the co-transfected cells in an image. The two images were overlayed so we can observe both the green and purple colors. It seems that generally, the purple color is more intense so it dominates most of the images. The initial segmentation is done on the green images, so despite the images appearing purple, green is in fact present in all of the cells displayed.

Looking closely at how the points are grouped together as single cells in Fig. 3.9 indicates that improvements could be made in how the clustering is done. There are regions where it is clear that two separate cells should be identified, yet they are grouped together as a single cell. Rather than just connecting all adjacent pixels that are identified as cell material, it may be necessary to fine-tune the method to look for indicators of a cell membrane to separate nearby cells.

One of the cell classification models found is shown in Fig. 3.10. This model was able to correctly identify all the cells in a test set. The model can be interpreted as

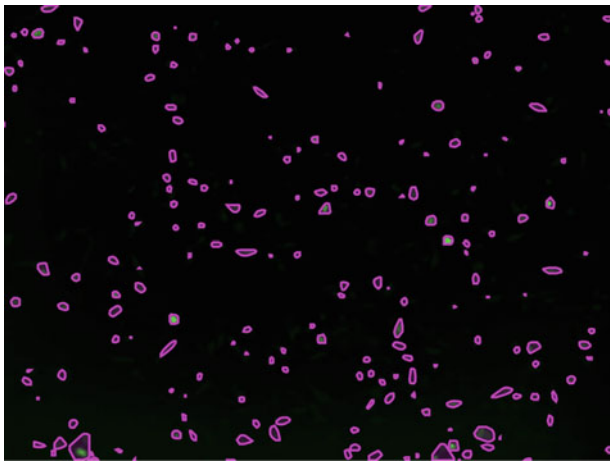


Fig. 3.9 Shown here are the outlines of each cell that result from the model selecting points that are either cell or non-cell material and connecting the adjacent ones. Looking near the bottom left corner we can see an example of where two cells are incorrectly identified as a single cell

follows: If the max blue value minus the min blue value is greater than the average green value, then the cell is co-transfected, otherwise, it is not co-transfected. This interpretation ignores the left half of the tree since the left half of the tree reduces to a constant value of false.

To make the use of AL-GP with DT-GP easy to use for this project, a simple GUI was developed that allows the user to fine-tune a pre-trained segmentation model and then supply labels for cells identified as uncertain by the classification models. A snapshot of the GUI in use is shown in Fig. 3.11. The use of this GUI allowed for automation of cell classification after labelling just a few cells, where previously it was necessary to manually label every cell in every video frame.

3.5 AL-GP Applied to SEE-Segment

3.5.1 SEE-Segment

SEE-Segment was used to evolve a population of image segmenters. Each individual model consists of a segmentation strategy and 8 parameters. The parameters and strategy are optimized through generations. While each individual contains 8 parameters not all parameters are used by all strategies. Mutation and crossover are used as the variation operators. Tournament selection is utilized for selection. All models selected from tournament selection then have a 90% chance of crossover and then from those offspring, there is a 90% chance of a mutation. Elitism is also utilized by guaranteeing the hall of fame models (top 10 genotypically unique models) are

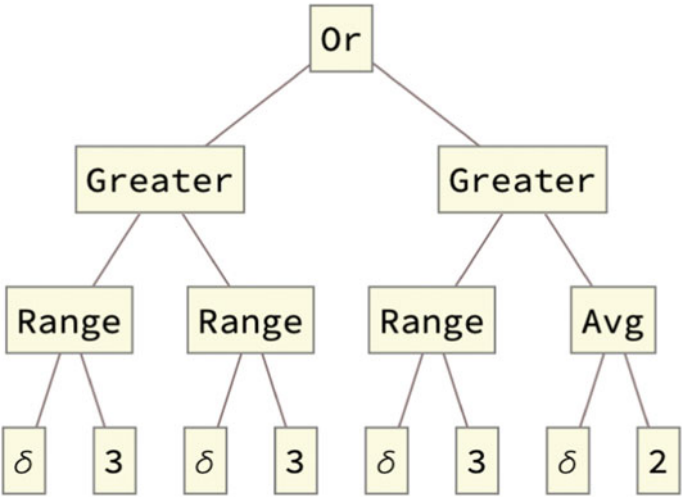


Fig. 3.10 Here is a graphical representation of a model's genotype that was evolved and able to correctly classify a test set. In the tree, "Range" is a function of the max value minus the min value in a specific color stream of the data. The data is stored with three values for each pixel, the red, green, and blue values in that order. So "Range" with the value 3, means the max minus the min value in the cell in the blue values. "Avg" is a function of the average value in a specific color. Looking at this tree we can see that the left side of the tree would always evaluate to false, so only the right side of the tree contains the effective code

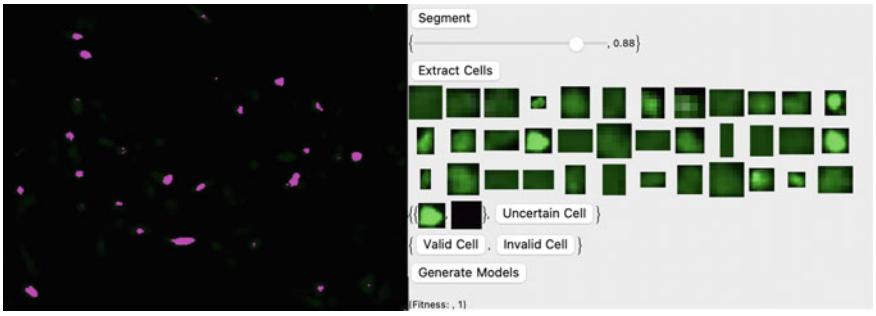


Fig. 3.11 The image shows the GUI developed to make easy use of AL-GP to develop models to classify cells. The slider allows the user to fine-tune a pre-trained segmentation model. The segmented cells are then shown to the user and when "Uncertain Cell" is clicked it presents the two image streams for the cell identified as most uncertain given the current classification models. The user can then select if the cell is co-transfected or not by selecting "Valid" or "Invalid". Once a selection is made, the user can then click "Generate Models" to continue model development on the expanded training set. The fitness on the training set is displayed at the bottom to give the user a sense of the model quality on the training set. Once a few cells have been labelled by the user and a model of sufficient quality is achieved the model can then be deployed and applied to the remaining cells in each video frame

Table 3.2 SEE-Segment parameter settings

Parameter	Setting
Mutation rate	90
Crossover rate	90
Spawn rate	100-HoF-UniqueMods
Hall of fame size	10
Tournament size	4
Population size	100
Generations per AL iteration	100

preserved across generations. The parameters used to run SEE-Segment are shown in Table 3.2. Spawn rate refers to the number of new models introduced in a generation. The number spawned each generation is the number required to bring the population size back up to 100 after including the hall of fame models and the unique individuals produced from mutation and crossover. Tournament size, population size, and generations are the only parameters modified from the default settings. Those settings were chosen since they allowed for sufficient model development while keeping computation time reasonable in each iteration of learning.

3.5.2 AL Implementation for SEE-Segment

Two active learning methods were implemented that vary in how they select the ensemble. The first method simply uses SEE-Segment's hall of fame, which consists of the top 10 genotypically unique models. The second method goes further to ensure models are unique by selecting the top 10 phenotypically unique models. These ensemble selection methods differ from the original AL-GP approach since here we start with only a single image. The original approach relies on generating diverse data clusters and selecting models that best fit each data cluster. If that approach was utilized here, we would initially get an ensemble of one model, which would lack any sort of uncertainty metric.

Using the selected ensemble we compute the average pairwise disagreement of all the models in the ensemble on each potential image. The image with the maximum average pairwise disagreement is selected for labelling and added to the training set. In the event of a tie, the first image found with the max value is selected. An overview of the active learning algorithm is shown in Algorithm 4 and the method for computing uncertainty is shown in Algorithm 5. The pairwise uncertainty on an image between two models is computed by utilizing SEE-Segment's fitness function where instead of supplying the fitness function with a model's predicted segmentation mask and a true mask, the predicted masks from both models are supplied. This leads to a reasonable measure of how different two model's predicted masks are.

Algorithm 4 Active Learning Process for SEE-Segment

```
TrainingData ← RandomImage                                  ▷ Select 1 Random Image
Evolver ← GeneticSearch.Evolver(TrainingData)               ▷ Initialize evolver
evolver.run()                                               ▷ Evolve models with initial data
while i ≤ iterations do                                     ▷ While max iterations not reached
    HoF ← Evolver.hof                                       ▷ Extract hall of fame
    SelectedImage ← MaximizeUncertainty(HoF, Data)          ▷ Find image that maximizes uncertainty
    TrainingData ← Append(TrainingData, SelectedImage)      ▷ Add new image to training data
    Evolver.UpdateData(TrainingData)                        ▷ Update training data in evolver
    evolver.run()                                           ▷ Evolve models with new data
end while
```

Algorithm 5 Uncertainty Computation SEE-Segment

```
procedure UNCERTAINTY(HoF,Image)
    Uncertainties ← []                                      ▷ Initialize uncertainties list
    Pairs ← GeneratePairs(HoF)                              ▷ Generates all pairs of models in HoF
    for i 1 to len(Pairs) do
        Prediction1 ← EvaluateModel(Pairs[i][0], Image)     ▷ Generate prediction of first model in pair
        Prediction2 ← EvaluateModel(Pairs[i][1], Image)     ▷ Generate prediction of second model in pair
        UncertaintyVal ← FitnessFunction(Prediction1, Prediction2)   ▷ Compute uncertainty
        Uncertainties ← Uncertainties.append(UncertaintyVal)         ▷ Append to list
    end for
    MeanUncertainty ← mean(Uncertainties)                   ▷ Compute mean of uncertainties
    Return MeanUncertainty
end procedure
```

3.5.3 KOMATSUNA Results

To determine the success of active learning on the KOMATSUNA data set, active learning was compared to two naïve image selection methods. The first method is ordered selection of training data, where the images are added to the training set in their natural order in the data set. Specifically, images in the KOMATSUNA data set are from a time series and labelled in order, so the order would be from first to last in the time series. The second method is random order selection, where a new image is added to the training set randomly.

The KOMATSUNA data set contains 300 images. Of the 300 images, 50 of them were reserved as the test set and the remaining 250 were available to be selected for training. Each approach began with a single image in the training set and could select one new image to be added to the training set each iteration.

The results of comparing active learning, random selection, and ordered selection are shown in Fig. 3.12. The results show that ordered selection performs very poorly, having the worst fitness values on average and also have the widest distribution of fitness values. This is not surprising since the data set is ordered as a set of several time series. Images nearby in the sequence will be very similar since they are of the same plant within a short period of time. This makes it challenging for models trained only using the beginning of the time series to predict the segmentation patterns correctly of plants that are much larger later in the time series. Random sampling and active learning both eventually achieve similar test errors as seen in Fig. 3.12, but active learning achieves low error in fewer iterations and converges more quickly. The ability of random sampling to achieve good fitness in relatively few iterations is

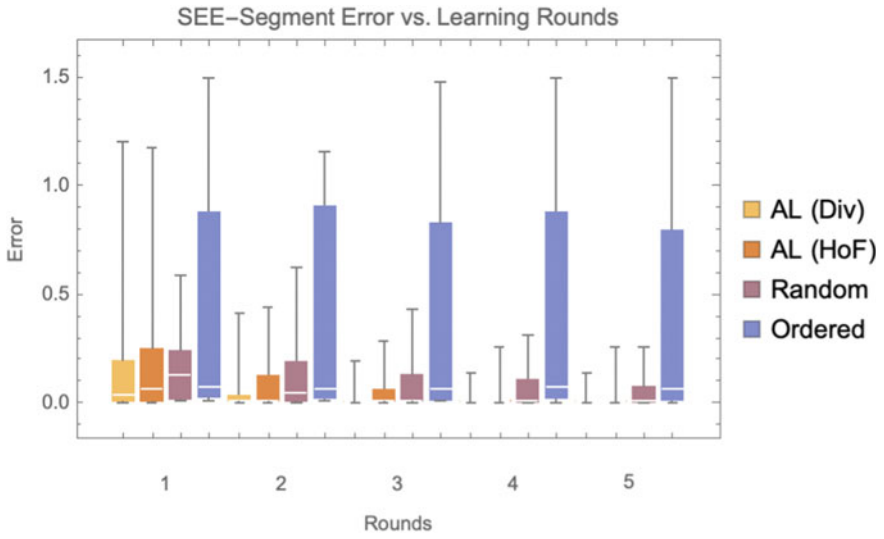


Fig. 3.12 Two active learning methods are compared with ordered data selection and random data selection with the distribution of errors shown after each round of learning. Each experiment was repeated 40 times, so each bar represents the distributions from 40 repeated trials. The data shows that active learning outperforms both random data selection and ordered data selection. The active learning methods most quickly decrease error and then most quickly converge. On and after round 5 we see that from the minimum to the 3rd quartile the active learning error distribution appears essentially as a flat line. AL (Div) represents AL when using a diverse ensemble and AL (HoF) represents AL when using the HoF as the ensemble. We see that using a diverse ensemble improves performance

likely a result of the data set being very balanced, so a random sample that is large enough will contain data representative of the whole set. The key difference between active learning and random sampling is the rate at which low test error is achieved and the consistency of low error solutions in few rounds of learning.

To determine how well active learning can overcome unbalanced data, the KOMATSUNA data set was modified to heavily oversample 3 images by duplicating 3 random images 50 times and adding them back into the training set. The test set is still balanced since it was separated prior to the duplication of the images. The results of comparing the two active learning methods and random data selection are shown in Fig. 3.13. Ordered sampling was not included here since it already performed poorly with the balanced data. The results show that already after the second iteration both active learning approaches are beginning to converge to a good solution. By the third iteration the AL (Div) approach appears to have converged, and the AL (HoF) approach appears to converge by the 5th iteration. After the 6th round, random still has a fairly wide distribution compared to the active learning approaches.

Figures 3.14 and 3.15 compare a sample run of active learning and ordered selection by tracking the segmentation pattern on a sample from their test sets.

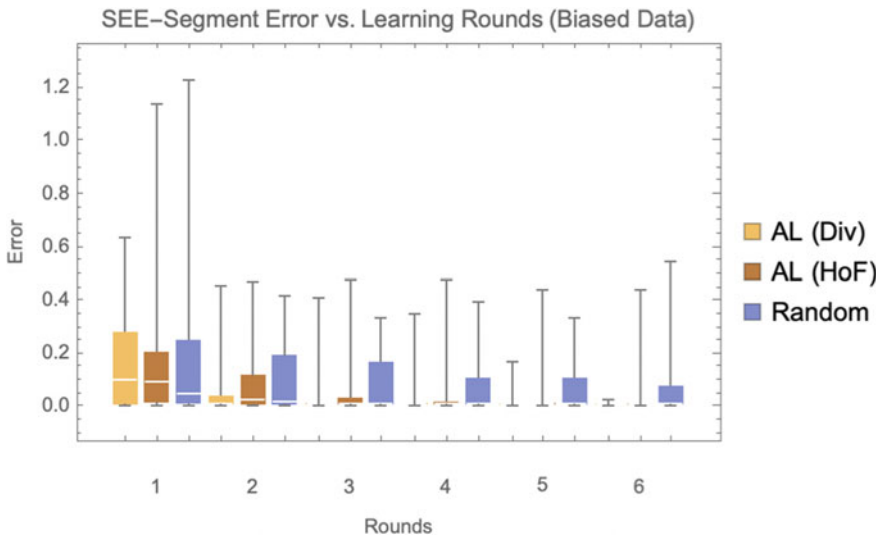


Fig. 3.13 Two active learning methods are compared with random data selection on the biased KOMATSUNA data. The distributions of errors are shown after each round of learning. Each experiment was repeated 40 times, so each bar represents the distributions from 40 repeated trials. The data shows that active learning outperforms random data selection. We also see that the active learning method using diverse ensembles performs better than the method using the hall of fame

Figure 3.14 demonstrates how active learning develops a good model quickly and improves over iterations. Comparing this to Fig. 3.15, we see how there is risk when using suboptimal sampling strategies to overfit training data and perform poorly on test data. We see that the ordered selection method actually gets worse by the fifth iteration, indicating that it is overfitting a non-representative training set. Even by the 20th iteration ordered selection arrives at a rather poor model.

3.6 Conclusions

AL-GP was extended to two new machine learning systems, DT-GP and SEE-Segment, for the purpose of accelerating the development of image processing models. It was shown that AL-GP can successfully be extended to tasks outside of symbolic regression and also to other population-based machine learning systems. When applying AL-GP to DT-GP we verified that active learning accelerates model development on the KOMATSUNA dataset, which is a simple dataset to model for DT-GP. Once verifying AL-GP is applicable to DT-GP we applied it to the task of cell classification to aid another lab at MSU accelerate their research by partially automating the task of classifying cells by using a human-in-the-loop method where the human researcher supplies labels for the cells identified as most informative by

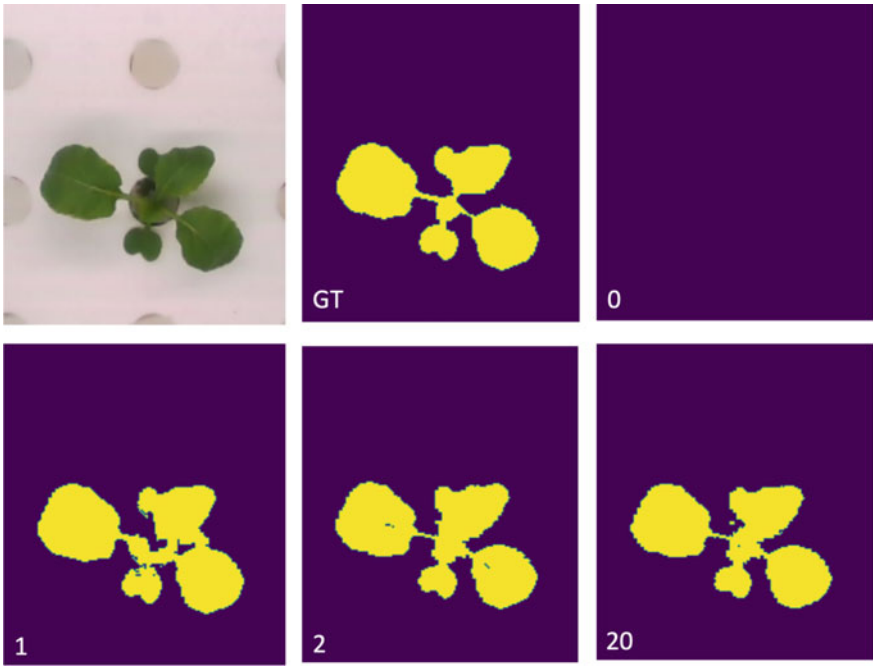


Fig. 3.14 Shown here is the progress from an example run using active learning on the KOMATSUNA plant data. The top left image shows the original image and the top middle shows the ground truth segmentation pattern. The top right begins the active learning iterations and continues down then to the right. After the initial random image, we see that the best model is terrible on an image from the test set. We see quick improvement and convergence to a good solution. The segmentation patterns shown are from the 0th, 1st, 2nd, and 20th iterations of active learning, where the 0th represents training on the first randomly selected image used to seed the training set

the GP system. This results in just a few samples requiring human labelling before finding models that could then be deployed to automatically label the rest of the cells.

When applying AL-GP to SEE-Segment we explored how active learning compares to random sampling and ordered sampling. We also explored how well it can overcome biased data and compared two different methods for selecting ensembles to be used in the uncertainty computation. We observed that both active learning methods outperformed random and ordered sampling by finding better solutions more quickly and consistently across 40 repeated trials. We also observed that biasing the data did not have a significant impact on the active learning approaches, still outperforming random sampling in finding good solutions more quickly and consistently. The ensemble selection method that used phenotypic diversity was found to perform better than the method that used genotypic diversity. This shows that a diverse ensemble is an essential part of the AL-GP approach.

Additional work is currently underway exploring how AL-GP impacts SEE-Segment's model development on a more challenging dataset where near perfect

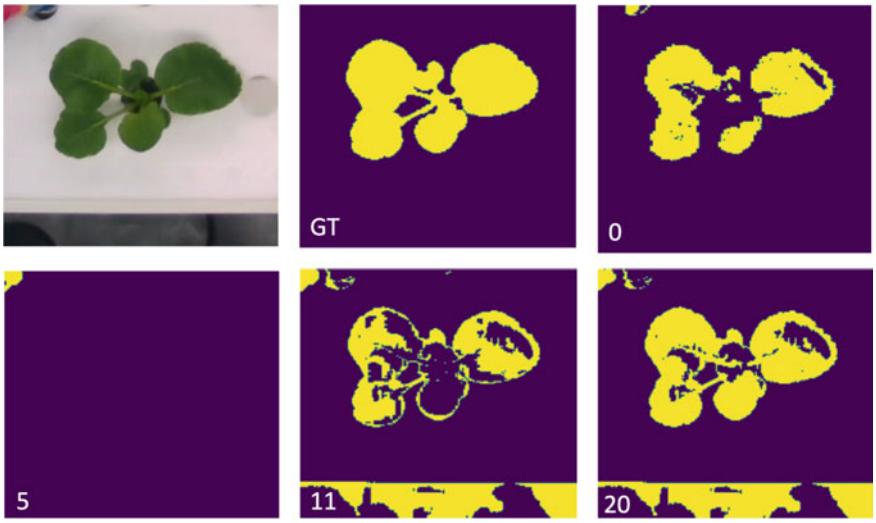


Fig. 3.15 Shown here is the progress from an example run using ordered selection on the KOMATSUNA plant data. The top left image shows the original image and the top middle shows the ground truth (GT) segmentation pattern. The top right begins the ordered selection iterations and continues down then to the right. After the initial random image, we see that the best model is actually not too bad. By the 5th iteration, we actually see that the performance on this example test image actually worsens significantly. By the 20th iteration, we see some improvement but the model is still performing poorly

solutions are not achievable. The preliminary results indicate that AL is developing better models with fewer training samples, but more runs are still needed to confirm these results.

This work confirms that AL-GP can be successfully applied to population-based ML systems outside of StackGP and further that AL-GP is not restricted to regression tasks. It also shows how AL-GP can be applied to help accelerate research projects by reducing the time and cost of collecting and labelling training samples.

Acknowledgements Computer support by MSU's iCER high-performance computing centre is gratefully acknowledged.

References

1. Cohn, D.A.: Neural network exploration using optimal experiment design. Neural Netw. **9**, 1071–1083 (1996)
2. Cohn, D.A., Ghahramani, Z., Jordan, M.I.: Active learning with statistical models. J. Artif. Intell. Res. **4**(1), 129–145 (1996)
3. Colbry, D.: See-Segment. https://github.com/see-insight/see-segment

4. Haut, N., Banzhaf, W., Punch, B.: Active learning improves performance on symbolic regression tasks in StackGP. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO '22, pp. 550–553, New York, NY, USA. Association for Computing Machinery (2022)
5. Kotanchek, M., Smits, G., Vladislavleva, E.: Exploiting trustable models via pareto gp for targeted data collection. In: Riolo, R., Soule, T., Worzel, B. eds., Genetic Programming Theory and Practice VI, pp. 145–162. Springer (2009)
6. Kremer, J., Pedersen, K.S., Igel, C.: Active learning with support vector machines. WIREs Data Mining Knowl. Dis. **4**(4), 313–326 (2014)
7. Lewis, D.D., Gale, W.A.: A sequential algorithm for training text classifiers. In: Bruce, W.C., van Rijsbergen, C.J. (eds.), SIGIR '94, London, pp. 3–12. Springer London (1994)
8. Lindley, D.V.: On a measure of the information provided by an experiment. Ann. Math. Stat. **27**(4), 986–1005 (1956)
9. Ren, P., Xiao, Y., Chang, X., Huang, P.-Y., Li, Z., Gupta, B.B., Chen, X., Wang, X.: A survey of deep active learning. ACM Comput. Surv. **54**(9), 1–40 (2021)
10. Ricker, B., Mitra, S., Castellanos, E.A., Grady, C.J., Pelled, G., Gilad, A.A.: Proposed three-phenylalanine motif involved in magnetoreception signaling of an actinopterygii protein expressed in mammalian cells (2022)
11. Shen, D., Guorong, W., Suk, H.-I.: Deep learning in medical image analysis. Ann. Rev. Biomed. Eng. **19**(1), 221–248 (2017)
12. Suk, H.-I., Shen, D.: Deep Learning in Diagnosis of Brain Disorders, pp. 203–213. Springer Netherlands, Dordrecht (2015)
13. Uchiyama, H., Sakurai, S., Mishima, M., Arita, D., Okayasu, T., Shimada, A., Taniguchi, R.I.: KOMATSUNA Dataset
14. Wäldchen, J., Mäder, P.: Plant species identification using computer vision techniques: a systematic literature review. Arch. Comput. Methods Eng. **25**(2), 507–543 (2018)