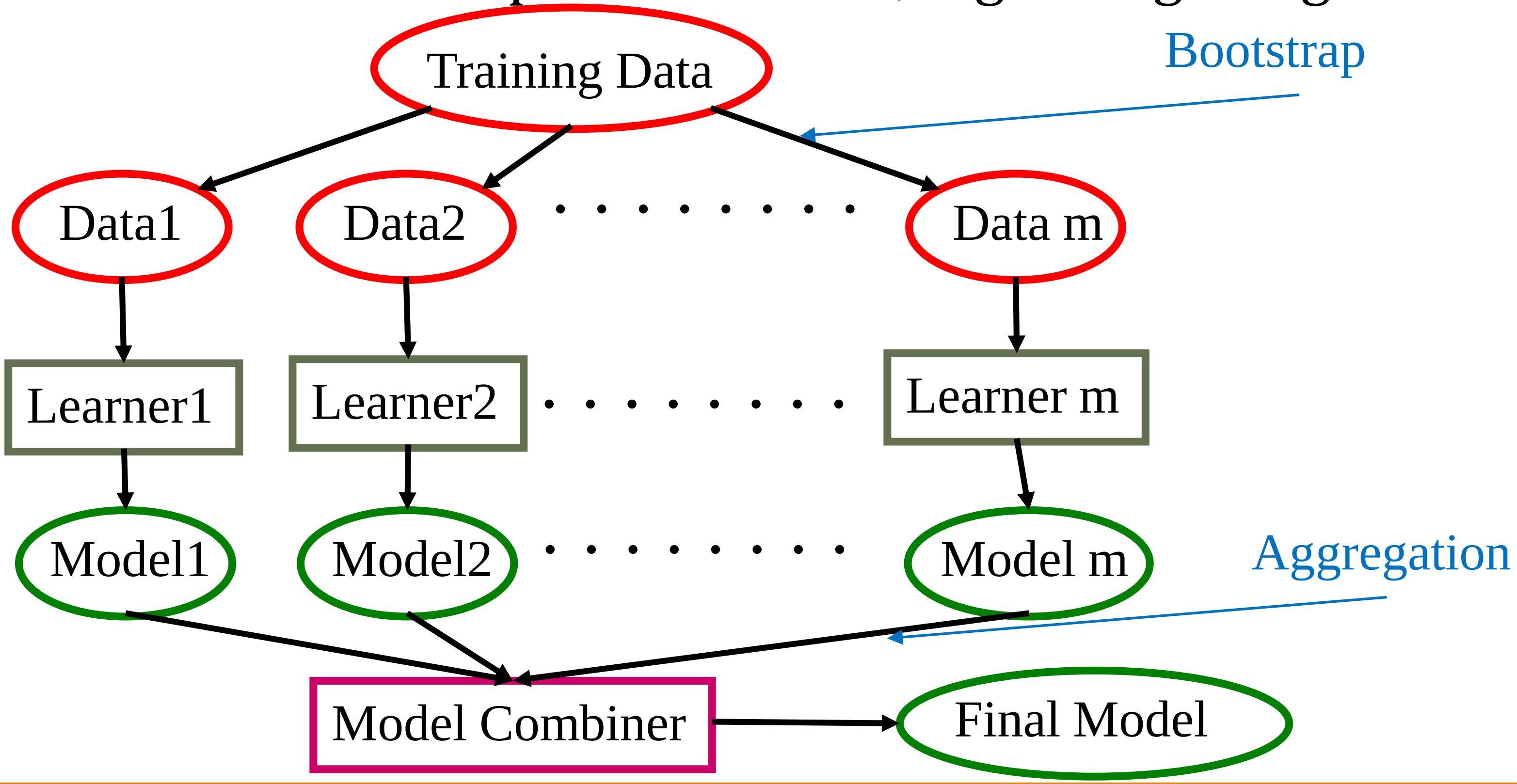
Ensemble Learning Based Classifiers

DR. DINESH KUMAR VISHWAKARMA

Learning Ensembles

- Learn multiple alternative definitions of a concept using different training data or different learning algorithms.
- Combine decisions of multiple definitions, e.g. using weighted voting.



Value of Ensembles

- When combing multiple ***independent*** and ***diverse*** decisions each of which is at least more accurate than random guessing, random errors cancel each other out, correct decisions are reinforced.
- Human ensembles are demonstrably better
  - How many jelly beans in the jar?: Individual estimates vs. group average.
  - Who Wants to be a Millionaire: Expert friend vs. audience vote.

Homogenous Ensembles

- Use a single, arbitrary learning algorithm but manipulate training data to make it learn multiple models.
  - Data1 ≠ Data2 ≠ … ≠ Data m
  - Learner1 = Learner2 = … = Learner m
- Different methods for changing training data:
  - **Bagging: Resample training data**
  - **Boosting: Reweight training data**
  - **DECORATE: Add additional artificial training data**
- In WEKA, these are called ***meta-learners***, they take a learning algorithm as an argument (***base learner***) and create a new learning algorithm.

Bagging vs Boosting

- **Sequential ensemble,** popularly known as ***boosting***, here the weak learners are sequentially produced during the training phase. The performance of the model is improved by assigning a higher weightage to the previous, incorrectly classified samples. An example of boosting is the AdaBoost algorithm.
- **Parallel ensemble**, popularly known as ***bagging***, here the weak learners are produced parallelly during the training phase. The performance of the model can be increased by parallelly training a number of weak learners on bootstrapped data sets. An example of bagging is the Random Forest algorithm.

Bagging

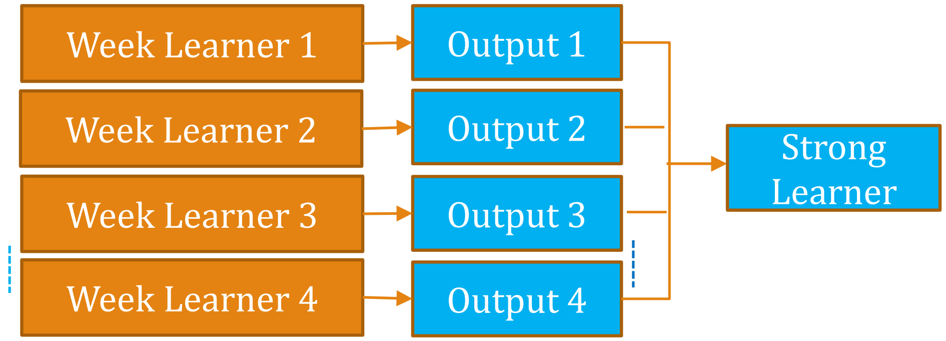
- Create ensembles by repeatedly randomly resampling the training data (Brieman, 1996).
- Given a training set of size n, create m samples of size n by drawing n examples from the original data, ***with replacement***.
  - Each ***bootstrap sample*** will on average contain 63.2% of the unique training examples, the rest are replicates.
- Combine the m resulting models using simple majority vote.
- Decreases error by decreasing the variance in the results due to ***unstable learners***, algorithms (like decision trees) whose output can change dramatically when the training data is slightly changed.

Boosting

- Originally developed by computational learning theorists to guarantee performance improvements on fitting training data for a ***weak learner*** that only needs to generate a hypothesis with a training accuracy greater than 0.5 (Schapire, 1990).
- Revised to be a practical algorithm, AdaBoost, for building ensembles that empirically improves generalization performance (Freund & Shapire, 1996).
- Examples are given weights. At each iteration, a new hypothesis is learned and the examples are reweighted to focus the system on examples that the most recently learned classifier got wrong.

Why Is Boosting Used?

- Consider a dataset of images of **cats and dogs.**
- Identifying these images by using some rules like:
  - The image has pointy ears: Cat
  - The image has cat shaped eyes: Cat
  - The image has bigger limbs: Dog
  - The image has sharpened claws: Cat
  - The image has a wider mouth structure: Dog



- These rules are helping to identify whether an image is a Dog or a cat.
- If we classify using individual (single) rule, the prediction would be flawed. Each of these rules, individually, are called weak learners because these rules are not strong enough to classify an image as a **cat or dog**.
- To improve prediction, we can combine the prediction from each of these weak learners by using the majority rule or weighted average.
- Above Ex. Has 5 weak learners & the majority of these rules (i.e. 3 out of 5 learners predict the image as a cat) gives us the prediction that the image is a cat

Boosting: Basic Algorithm

- **Step 1:** The base algorithm reads the data and assigns equal weight to each sample observation.
- **Step 2:** False predictions made by the base learner are identified.
  - In the next iteration, these false predictions are assigned to the next base learner with a higher weightage on these incorrect predictions.
- **Step 3:** Repeat step 2 until the algorithm can correctly classify the output.
- Therefore, the main aim of Boosting is to focus more on miss-classified predictions.

Types of Boosting

- There are three main ways through which boosting can be carried out:
  - Adaptive Boosting or AdaBoost
  - Gradient Boosting
  - XGBoost

Adaptive Boosting

- Implemented by combining several weak learners into a single strong learner.
- The weak learners in AdaBoost take into account a single input feature and draw out a single split decision tree called the decision stump. Each observation is weighed equally while drawing out the first decision stump.
- The results from the first decision stump are analyzed and if any observations are wrongfully classified, they are assigned higher weights.
- Post this, a new decision stump is drawn by considering the observations with higher weights as more significant.
- Again if any observations are misclassified, they're given higher weight and this process continues until all the observations fall into the right class.

AdaBoost Pseudocode

```
TrainAdaBoost(D, BaseLearn)
  For each example $d_i$ in $D$ let its weight $w_i$=1/|$D$|
  Let $H$ be an empty set of hypotheses
  For $t$ from 1 to $T$ do:
      Learn a hypothesis, $h_t$, from the weighted examples: $h_t$=BaseLearn($D$)
      Add $h_t$ to $H$
      Calculate the error, $\varepsilon_t$, of the hypothesis $h_t$ as the total sum weight of the
          examples that it classifies incorrectly.
      If $\varepsilon_t > 0.5$ then exit loop, else continue.
      Let $\beta_t = \varepsilon_t / (1 - \varepsilon_t)$
      Multiply the weights of the examples that $h_t$ classifies correctly by $\beta_t$
      Rescale the weights of all of the examples so the total sum weight remains 1.
  Return $H$
TestAdaBoost($ex$, $H$)
     Let each hypothesis, $h_t$, in $H$ vote for $ex$'s classification with weight log(1/ $\beta_t$)
     Return the class with the highest weighted vote total.
```

Gradient Boosting

- Gradient Boosting is based on sequential ensemble learning. The base learners are generated sequentially in such a way that the present base learner is always more effective than the previous one, i.e. the overall model improves sequentially with each iteration.
- The difference in this type of boosting is that the weights for misclassified outcomes are not incremented, instead, Gradient Boosting method tries to optimize the **loss function** of the previous learner by adding a new model that adds weak learners in order to reduce the loss function.
- The main idea is to overcome the errors in the previous learner's predictions. This type of boosting has three main components:
  - **Loss function** that needs to be ameliorated.
  - **Weak learner** for computing predictions and forming strong learners.
  - An **Additive Model** that will regularize the loss function.

XGBoost

- XGBoost is an advanced version of Gradient boosting method, it literally means eXtreme Gradient Boosting. XGBoost developed by Tianqi Chen, falls under the category of Distributed Machine Learning Community (DMLC).
- The main aim of this algorithm is to increase the speed and efficiency of computation. The Gradient Descent Boosting algorithm computes the output at a slower rate since they sequentially analyze the data set, therefore XGBoost is used to boost or extremely boost the performance of the model.
- XGBoost is designed to focus on computational speed and model efficiency. The main features provided by XGBoost are:
  - Parallelly creates decision trees.
  - Implementing distributed computing methods for evaluating large and complex models.
  - Using Out-of-Core Computing to analyze huge datasets.
  - Implementing cache optimization to make the best use of resources.

Learning with Weighted Examples

- Generic approach is to replicate examples in the training set proportional to their weights (e.g. 10 replicates of an example with a weight of 0.01 and 100 for one with weight 0.1).
- Most algorithms can be enhanced to efficiently incorporate weights directly in the learning algorithm so that the effect is the same (e.g. implement the WeightedInstancesHandler interface in WEKA).
- For decision trees, for calculating information gain, when counting example *i*, simply increment the corresponding count by w_i rather than by 1.

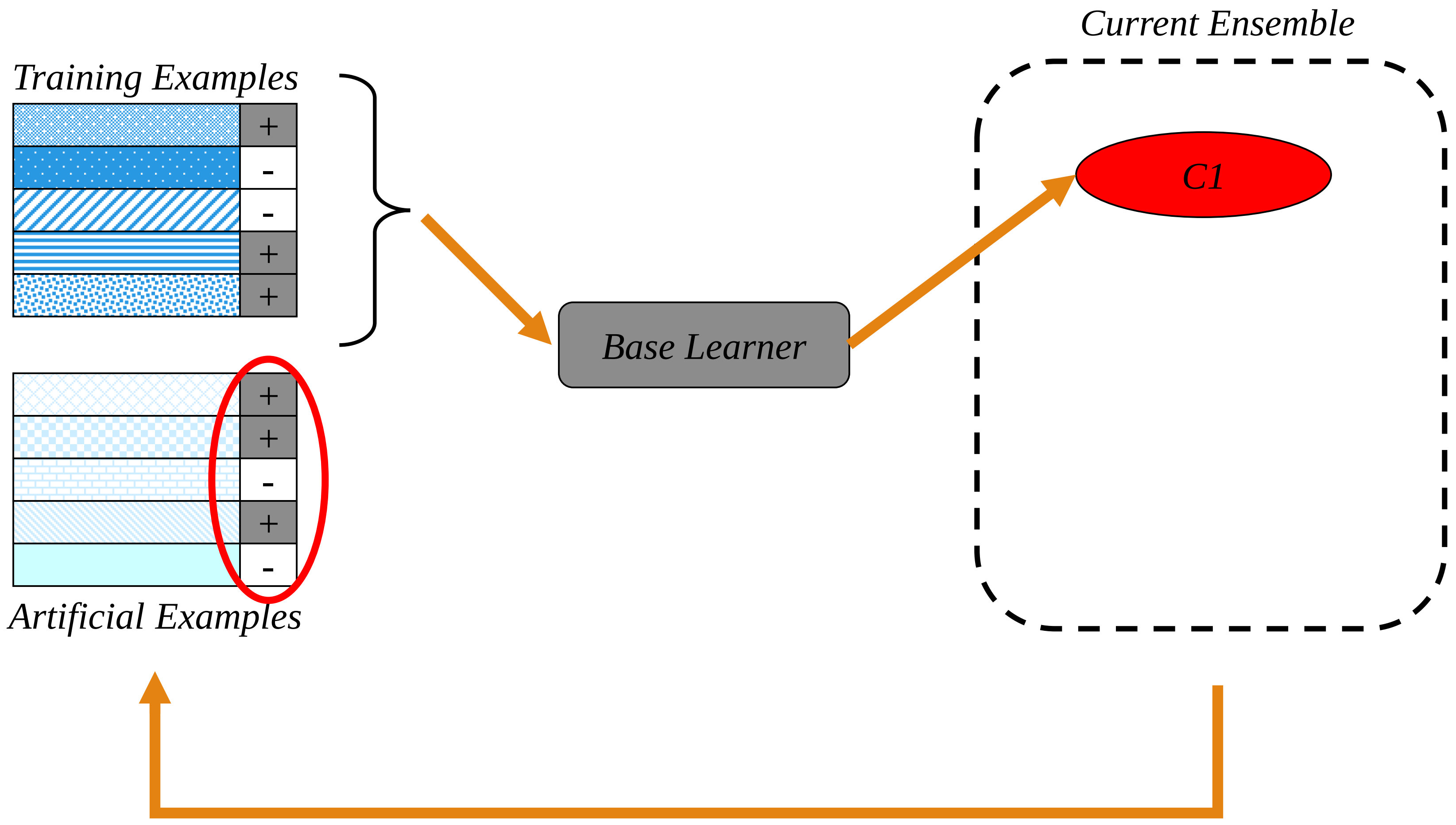
Experimental Results on Ensembles (Freund & Schapire, 1996; Quinlan, 1996)

- Ensembles have been used to improve generalization accuracy on a wide variety of problems.
- On average, Boosting provides a larger increase in accuracy than Bagging.
- Boosting on rare occasions can degrade accuracy.
- Bagging more consistently provides a modest improvement.
- Boosting is particularly subject to over-fitting when there is significant noise in the training data.

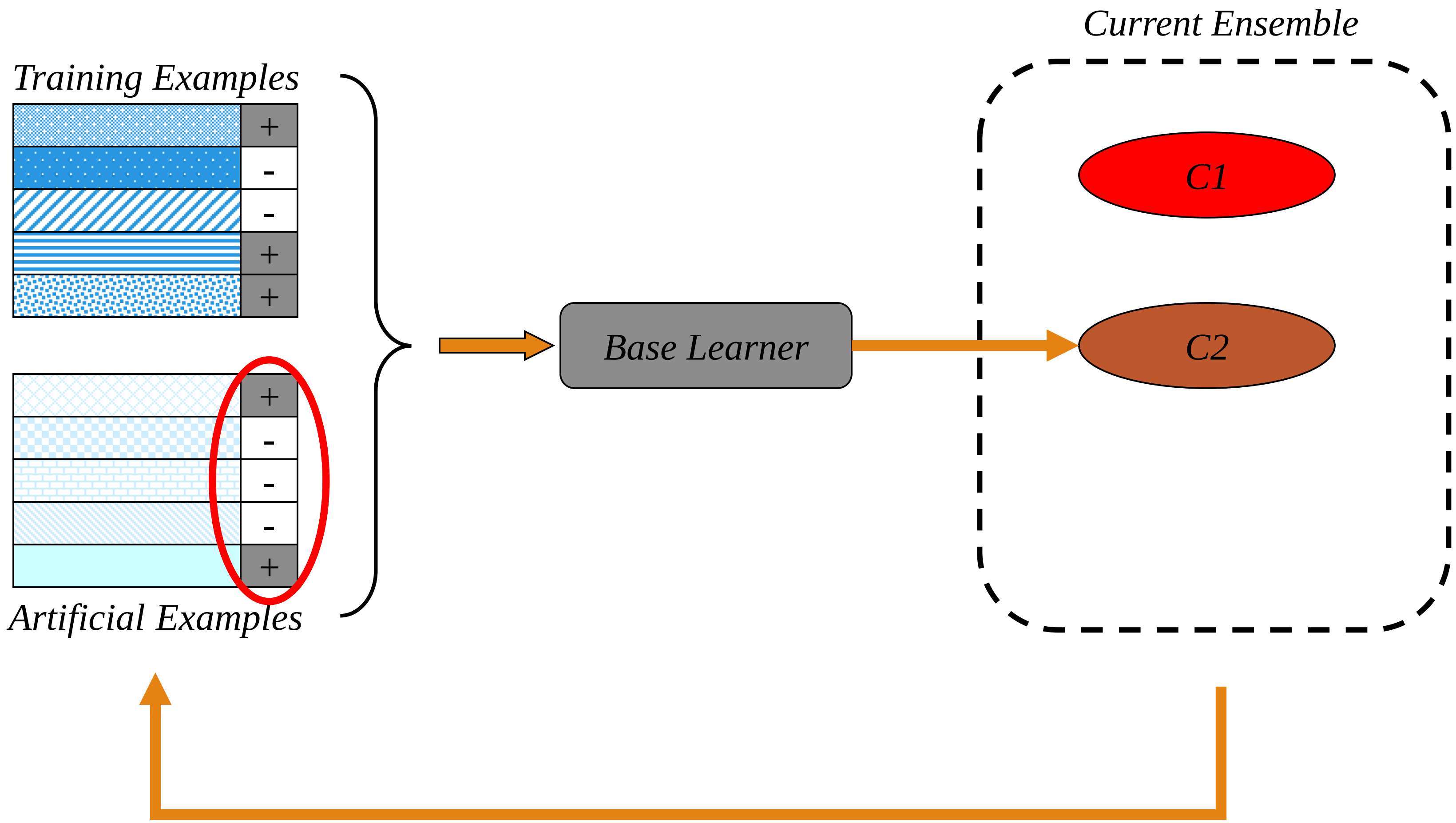
DECORATE (Melville & Mooney, 2003)

- Change training data by adding new artificial training examples that encourage diversity in the resulting ensemble.
- Improves accuracy when the training set is small, and therefore resampling and reweighting the training set has limited ability to generate diverse alternative hypotheses.

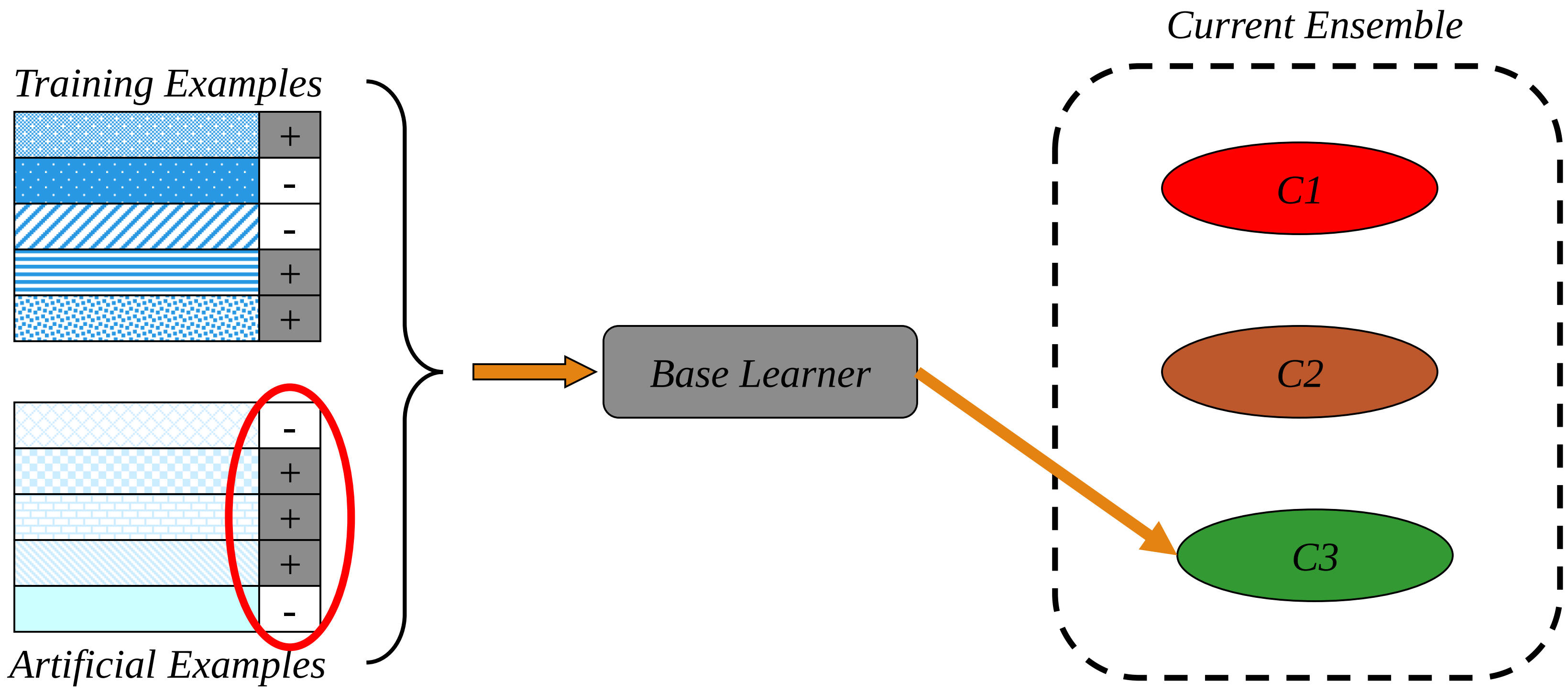
Overview of DECORATE



Overview of DECORATE



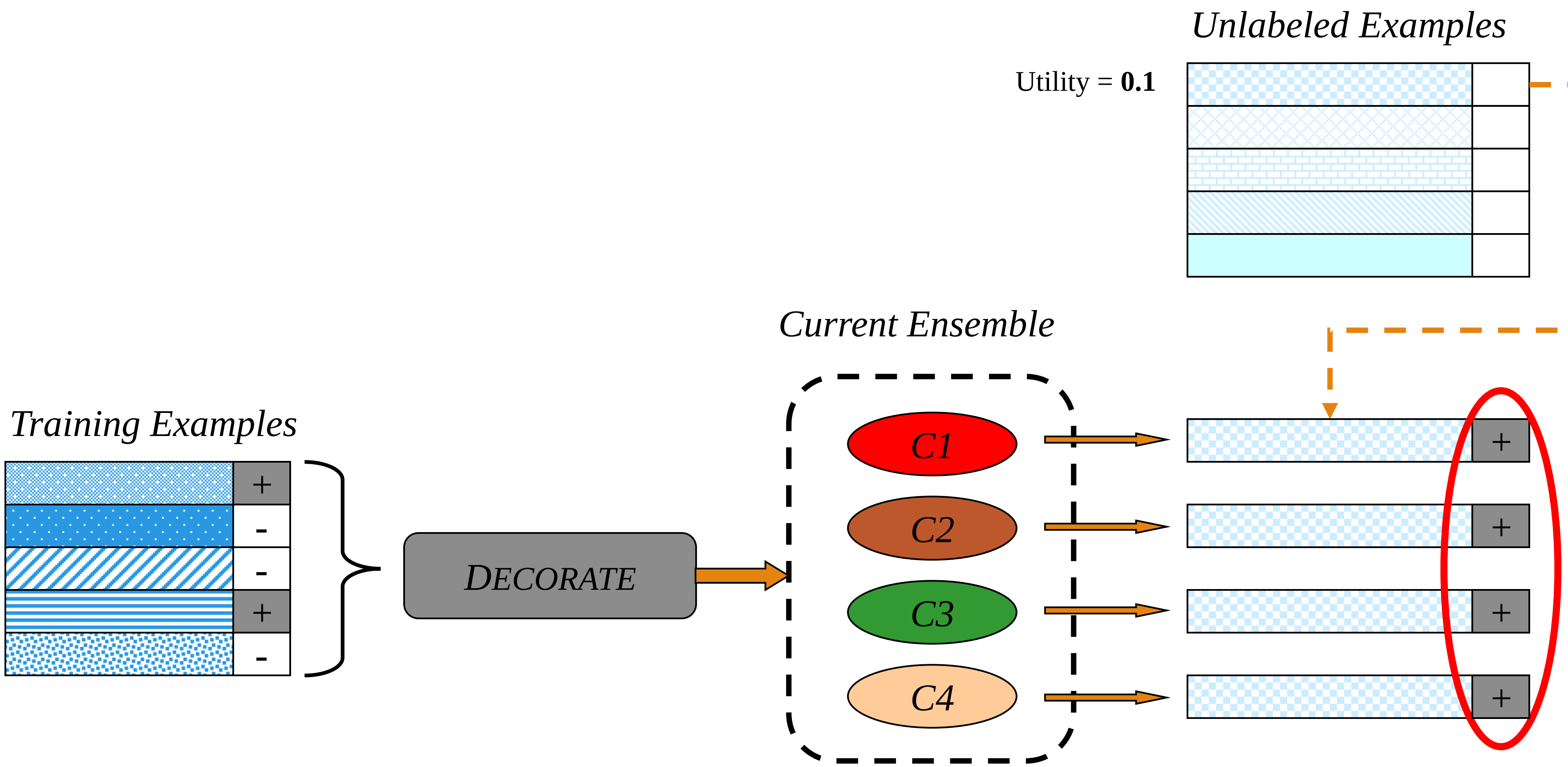
Overview of DECORATE



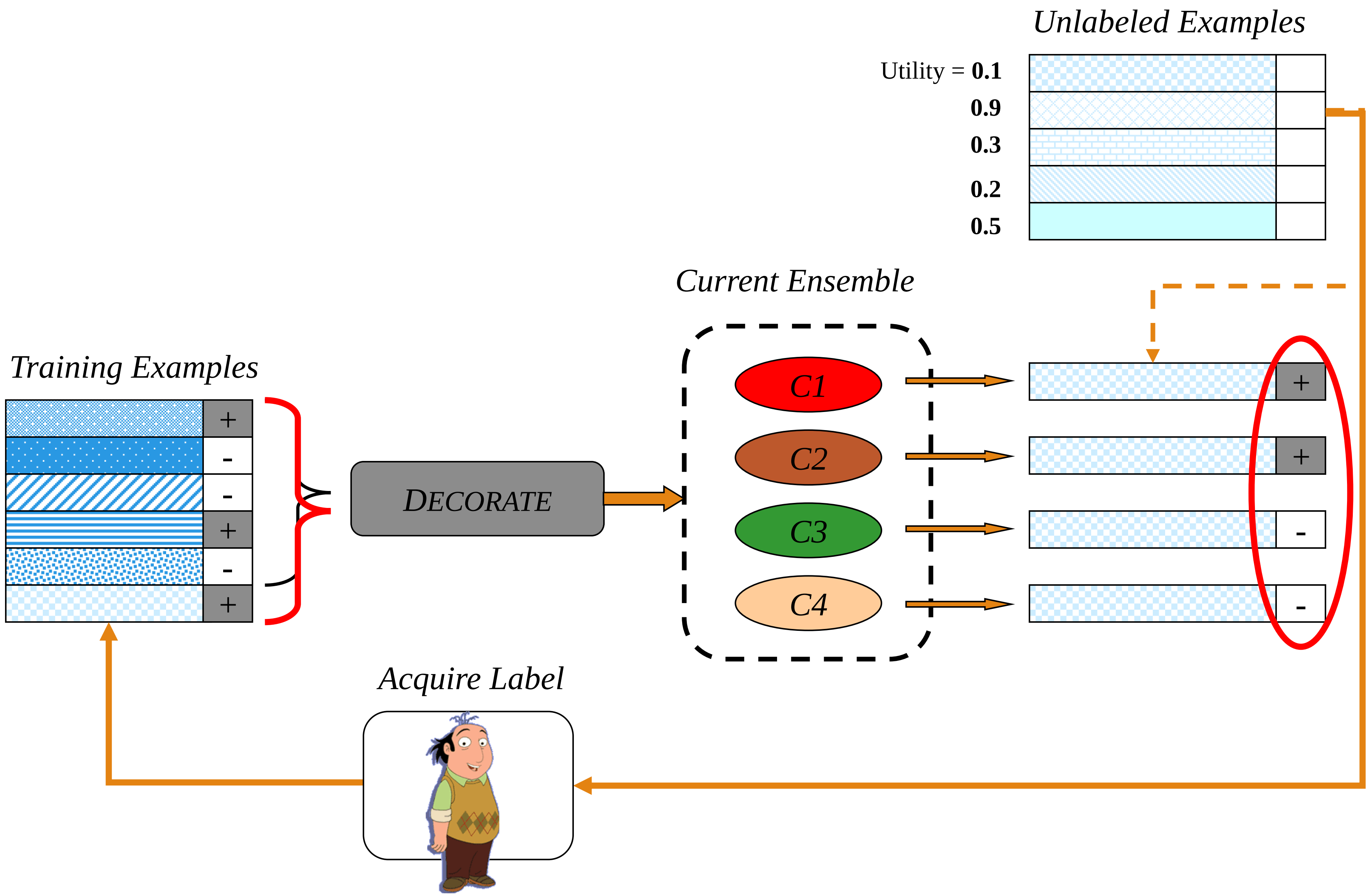
Ensembles and Active Learning

- Ensembles can be used to actively select good new training examples.
- Select the unlabeled example that causes the most disagreement amongst the members of the ensemble.
- Applicable to any ensemble method:
  - **QueryByBagging**
  - **QueryByBoosting**
  - **ActiveDECORATE**

Active-DECORATE



Active-DECORATE



Issues in Ensembles

- Parallelism in Ensembles: Bagging is easily parallelized, Boosting is not.
- Variants of Boosting to handle noisy data.
- How “weak” should a base-learner for Boosting be?
- What is the theoretical explanation of boosting’s ability to improve generalization?
- Exactly how does the diversity of ensembles affect their generalization performance.
- Combining Boosting and Bagging.