D-semble: Efficient Diversity-Guided Search for Resilient ML Ensembles

Abraham Chan
University of British Columbia
Vancouver, Canada
abrahamc@ece.ubc.ca

Arpan Gujarati
University of British Columbia
Vancouver, Canada
arpanbg@cs.ubc.ca

Karthik Pattabiraman
University of British Columbia
Vancouver, Canada
karthikp@ece.ubc.ca

Sathish Gopalakrishnan
University of British Columbia
Vancouver, Canada
sathish@ece.ubc.ca

Abstract

Supervised Machine Learning (ML) is used in many safety-critical applications, such as self-driving cars and medical imaging. Unfortunately, many training datasets have been discovered to contain faults. The accuracy of individual models when trained with faulty datasets can significantly degrade. In comparison, *ensembles*, consisting of multiple models combined through simple majority voting, are able to retain accuracy despite training data faults, due to their classification diversity, and are thus more resilient. However, there are many different ways to generate ML ensembles, and their accuracy can significantly differ. This creates a large search space for ensembles, making it challenging to find ensembles that maximize accuracy despite training data faults. We identify three different ways to generate diverse ML models, and present *D-semble*, a technique that uses Genetic Algorithms and diversity to efficiently search for resilient ensembles. We evaluate D-semble by measuring the balanced accuracies and F1-scores of ensembles it finds. Compared with bagging, greedy search, random selection, and the best individual model, ensembles found by D-semble are on average 9%, 16%, 28%, 32% more resilient respectively.

CCS Concepts

• **Computing methodologies** → **Ensemble methods**; **Discrete space search**; • **Computer systems organization** → **Reliability**.

Keywords

Error Resilience, Machine Learning, Training

ACM Reference Format:
Abraham Chan, Arpan Gujarati, Karthik Pattabiraman, and Sathish Gopalakrishnan. 2025. D-semble: Efficient Diversity-Guided Search for Resilient ML Ensembles. In *The 40th ACM/SIGAPP Symposium on Applied Computing (SAC '25), March 31-April 4, 2025, Catania, Italy*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3672608.3707978

1 Introduction

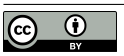
Many safety-critical applications, such as autonomous driving [3] and medical diagnosis [46], use machine learning (ML). Supervised learning, a form of ML, needs well-labelled training data, necessitating techniques such as crowdsourcing [9] and automated labelling by other ML models [17] to collect and label data at scale. Unfortunately, these techniques are vulnerable to *training data faults* [17, 20]. We focus on training data faults that occur unintentionally rather than targeted faults introduced by an adversary (*e.g.* data poisoning attacks), or faults affecting only certain classes.

Many well-used public datasets in various domains have been found to contain training data faults, e.g., labelling errors (Table 1 summarizes discovered faults). The prevalence of such training data faults can seriously degrade the ability of ML models to learn effectively and classify test inputs correctly [41], and cause potentially catastrophic failures. Misclassification in AVs can cause passenger injuries or death, while misclassification in medical diagnosis can lead to incorrect medical procedures, jeopardizing patient safety.

Eliminating training data faults is challenging [7, 8, 41] and hence, we aim to mitigate the effect of such faults (as outlined above). Our goal is to build *resilient* ML models. We define a resilient ML model as one that retains high predictive capability *despite* being trained with faulty training data. *Ensembles*, which consist of multiple, independently trained ML models on the same dataset, have been shown to be more resilient to training data faults than most techniques [7, 8, 57, 60]. This is because their constituent models learn different aspects of the feature space, enabling quorums to outvote misclassifications by a model. However, as each model in an ensemble is usually trained independently, large amounts of computational resources are required to train ensembles. Further, practitioners must experiment with a large number of model combinations to find resilient ensembles, which is time consuming [21].

Our objective is to reduce the computational effort involved in finding the most resilient ensemble for a given task. Unlike existing approaches that focus on finding the ensemble that makes the most correct classifications under fault-free datasets, we focus on searching for *resilient ensembles* that classifies correctly against faulty training datasets. Finding resilient ensembles requires the training dataset to be repeatedly injected with training data faults, and evaluating the ensemble's predictive capability until an optimal combination (*i.e.*, highest predictive capability for a given ensemble size) is found. This is computationally expensive.

We propose D-semble, a metaheuristic search approach, which efficiently finds ensembles that exhibit high resilience to training data faults (for a given task). D-semble reduces the training time during ensemble space exploration by using *heuristics based on increasing diversity among the models.* Specifically, D-semble takes (i) a training dataset, (ii) a set of models, and (iii) a set of *diversity*



SAC '25, March 31-April 4, 2025, Catania, Italy

ACM ISBN 979-8-4007-0629-5/25/03
https://doi.org/10.1145/3672608.3707978

operators as inputs. It returns an ensemble achieving high predictive capability despite the presence of training data faults. *To the best of our knowledge, we are the first technique to efficiently search the space of ensembles to find ensembles resilient to training data faults.*

Diversity operators systematically capture the ways to generate diversity in ensembles. Through our analysis of related work, we came up with three operators: (1) *architecture*, models of different architectures, (2) *data*, models trained with different subsets of the training data, (3) *snapshots*, models composed of different snapshots, taken when a model converges at a local minimum during training.

D-semble's metaheuristic search is based on *Genetic Algorithms* (GAs) [19]. GAs utilize biologically-inspired evolution operations such as crossover, mutation, and selection to efficiently search for near-optimal solutions in large state spaces [2]. We introduce two variations to the standard GA to further speed up the search, for our problem space. First, D-semble applies constraint correction to eliminate invalid ensemble combinations (i.e., those exceeding the desired ensemble size) during crossover and mutation operations. Second, D-semble leverages diversity as a heuristic to guide candidate selection, rather than using only accuracy metrics.

In summary, our principal contributions are:

- Identifying three diversity operators to systematically generate resilient ensembles against training data faults.
- Building D-semble[1], a GA-based approach that is customized to efficiently search for resilient ensembles against training faults, using the above heuristics.
- Studying fault distributions in real datasets, and building a fault injector based on the extracted fault distributions.
- Experimentally evaluating D-semble across three different image-classification datasets using the fault injector, by (1) comparing the predictive capability of generated ensembles with each diversity operator applied independently, and (2) the predictive capability versus time taken by D-semble to generate ensembles with bagging, greedy search and random selection. We use balanced accuracy (BA) and F1-score (F_1) as metrics to measure predictive capability across balanced and imbalanced multi-class datasets.

We have three main results. First, no individual diversity operator consistently provides high resilience, motivating the need to search for resilient ensembles combining the operators. Second, diversity and resilience are reasonably well correlated, thus, making diversity a good heuristic. Third, D-semble generates ensembles that are (on average) 9%, 16%, 28% more resilient than bagging, greedy search and random selection, while its search is only 1.5×, 3× and 4× slower than these baselines (respectively), across fault types.

2 Background

2.1 Motivation and Fault Model

Similar to prior work by Chan et al. [7], we consider three types of faults that commonly occur in training data.

(1) *mislabelling faults* – where data is erroneously labelled,
(2) *repetition faults* – where input-output pairs are repeated,
(3) *removal faults* – where a fraction of data may be deleted.

[1]D-semble is available at https://github.com/DependableSystemsLab/Dsemble

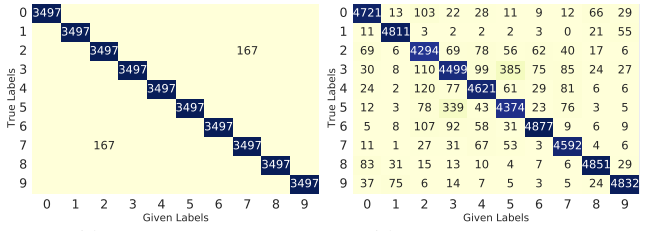
Table 1: Training data faults found in datasets across different domains. Mis stands for mislabelling.

Dataset	Domain	% Faulty	Fault Type(s)	Source
Udacity [55]	AV	33	Mis, Removal	[14]
Lyft Level 5 [24]	AV	70	Mis, Removal	[22]
ChestX-Ray14 [58]	Medical	20	Mis	[53]
ImageNet [13]	Objects	5.83	Mis	[41]
COCO [31]	Objects	45.5	Mis, Removal	[38]
GTZAN [54]	Music	10.6	Mis, Repetition	[51]

We collated a list of real-world datasets, over multiple domains, where training data faults were reported (Table 1). Mislabelling and removal faults (up to 70%) have been reported, even in safety-critical datasets [47], such as the Lyft and Chest X-Ray datasets. Repetition faults have been found in a well-used music dataset GTZAN [51]. *Thus, training data faults are prevalent in real-world datasets.*

Prior work on training data faults [7, 8] has largely focused on symmetrically (uniformly) distributed faults across label classes for simplicity. For example, the fault distribution for mislabelling is represented by a *noise transition matrix*, which is a square matrix where each row represents the instances of samples labelled as a certain class, while each column represents the instances belonging to the actual class. The diagonals represent the number of instances that are correctly labelled, while the off-diagonals are the number of instances that are mislabelled. Because the noise transition matrix is not known without manually inspecting the training dataset, a symmetric fault pattern (Fig. 1a) where the off-diagonals are equal, is usually assumed. In real datasets, however, distributions may not be symmetric because certain label classes can be more easily confused with one another during labelling, *e.g.*, images of cats may be more similar to dogs rather than trucks, resulting in a higher likelihood for mislabelling. For example, fault distributions in CIFAR are asymmetric (Fig. 1b).

It is important to find resilient ML models against varying amounts of training data faults and fault types. During development, practitioners often train ML models on smaller training datasets due to a lack of production data availability [39]. Smaller datasets are easier to inspect and clean. A larger production version of the dataset may only become available during later stages of the ML lifecycle [26], which may contain larger amounts of training data faults. Practitioners may find high accuracy models during development, then retrain and deploy those models on the production data, on the misleading assumption that their accuracy is retained [39].



(a) Symmetric Noise **(b) Observed Fault Pattern**

Figure 1: Noise Transition Matrices for CIFAR-10

2.2 Ensembles

Ensembles consist of multiple ML models trained independently, inspired by N-version programming [10, 36, 61]. During inference, the same input is fed into each ML model in the ensemble, which will in turn make individual predictions. These predictions are then combined through a voting scheme (*i.e.* simple majority voting) to produce a single prediction. Ensembles for classification have been found to be resilient against faulty training data due to the prediction diversity among its constituent models [7, 28]. Compared to other mitigation techniques against faulty training data, ensembles have also been shown to be the most resilient overall, while requiring no additional hyperparameter configuration [8].

2.3 Genetic Algorithms (GAs)

Genetic Algorithms (GAs) [19] use meta-heuristic search inspired by biological evolution. GAs aim to find the candidate with the highest fitness score, which determines how satisfactory a candidate solution is at solving the problem, without exhaustively exploring the entire search space. The GA terminates when any of the following occurs: (1) a candidate reaches a desired fitness score, (2) the algorithm exceeds a given time or number of rounds, or (3) the algorithm satisfies a custom termination condition.

2.4 BA, F_1 - Metrics for Predictive Capability

We use balanced accuracy (BA) [56] to measure predictive performance in multi-class datasets. In imbalanced datasets, where the number of samples belonging to each class is asymmetric in the training dataset, BA avoids a situation where a model's poor performance in underrepresented classes is masked by its superior performance in overrepresented classes. BA is calculated by measuring the recall for each class separately, and averaging them. Furthermore, BA and accuracy are similar for balanced datasets [56]. For certain binary datasets where false positives and negatives (*i.e.* missed medical diagnoses) are more important than true negatives (*i.e.* benign outcomes), we use the F1 score (F_1) [11] instead, which is calculated by the harmonic mean of the precision and recall.

3 Motivating Example

We present an illustrative example based on actual experimental observations to demonstrate the importance of selecting ensembles carefully when the training data contains faults. CIFAR-10 [27] contains more than 50,000 images belonging to 10 different categories of objects, and each class has 5000 images. We inject the original CIFAR-10 training dataset with mislabelling faults at five fault amounts from 10% to 50%, to achieve five faulty training datasets. These fault amounts are consistent with previous studies that show 33% [14] to 70% [22] of AV training datasets are faulty.

We focus on neural networks for ML models due to their ubiquitous use in image classification. Suppose we have seven neural network architectures: ConvNet, DeconvNet, MobileNet, ResNet18, ResNet50, VGG11, VGG16. We train each neural network on each of the five faulty training datasets. We exhaustively create ensembles of three models out of the seven possible neural networks at each fault amount. That is, for a single fault amount, we obtain $\binom{7}{3} = 35$ different ensembles. Each ensemble consists of three independently trained models on the *same faulty* training data. During inference, the same input image is fed into each model, and the models use simple majority voting to decide on a single prediction output. We choose ensemble sizes of three as it is the minimal number of models required for simple majority voting, and provides the maximum number of ensemble combinations.

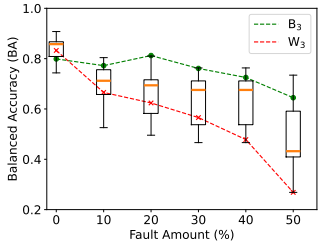


Figure 2: Boxplots of Size 3 Ensembles, CIFAR-10 with Mislabelling. Best and Worst Resilient Ensembles shown. BA Versus Fault amount. BA on y-axis starts at 0.2.

We ask: *Is it important to select ensembles to achieve high BA under training data faults?* We measure the BA of each ensemble and present a boxplot of the BA of 35 ensembles at each fault amount (Fig. 2). The orange line indicates the median resilient ensemble (*i.e.* a randomly chosen ensemble), while the top and bottom whiskers indicate the most and least resilient ensembles at each fault amount.

We show ensemble (B_3) consisting of three models (ConvNet, ResNet50, VGG11), with the highest BA and thus, most resilience, across five fault amounts, denoted by the green line. B_3 is the ensemble with the lowest Euclidean distance between the top whiskers across fault amounts. In contrast, the least resilient model (W_3) is shown as the red line. W_3 has the lowest Euclidean distance from the bottom whiskers. We make two observations from the figure:

(1) There is a significant difference in BA (up to 0.38) between the most and least resilient ensembles.
(2) On average, as the fault amount increases, the BA gap between the whiskers of the curve also increases from 0.16 to 0.47, starting at 10% mislabelling.

We infer that one must carefully select ensembles, especially when faults are present in training data. If not, prediction capabilities degrade significantly.

4 Methodology

4.1 Overview

We show the workflow for D-semble, our search framework for resilient ensembles in Fig. 3. D-semble consists of seven components, namely: (1) Fault Pattern Extraction and Injection, (2) Model Candidates, (3) Diversity Operators, (4) Ensemble Candidates, (5) Diversity-based Heuristics, (6) Fitness Function, (7) Evolutionary Search Optimization.

D-semble uses evolutionary search for ensemble architecture search. Model candidates consist of an initial pool of individual untrained models (*i.e.* ResNet50 or VGG16). Diversity operators are used for ensemble diversity generation (i.e., how to generate different models in an ensemble?). Ensemble candidates are generated by applying different diversity operators on the model candidates.

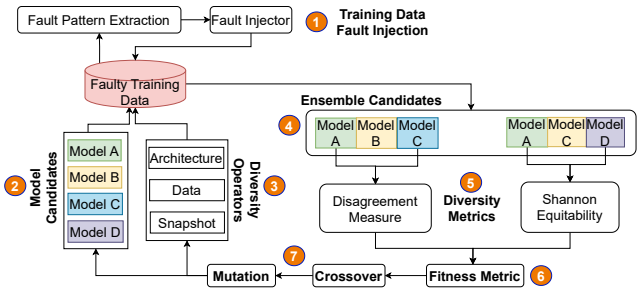


Figure 3: Workflow of D-semble. The numbered circles refer to the components in the workflow.

Diversity metrics measure the diversity between models in an ensemble, and are used as the heuristic to accelerate the search algorithm's convergence. The fitness function is used to determine how close an ensemble candidate is to achieving the search objective.

We use BA as the fitness function. The BA of the ensembles are evaluated across different fault amounts and averaged in each round. After each round, the ensemble candidates with the lowest combination of BA and diversity heuristic are removed. The evolutionary search optimization consists of crossover and mutation to breed the next generation of ensemble candidates. Crossover swaps the ensemble components (*i.e.*, model candidates and diversity operators) between two ensemble candidates. Mutation randomly changes one of the ensemble components (*i.e.* models) to another. The evolutionary search optimization explores different ensemble candidates, guided by the diversity-based heuristic.

In the following subsections, we explain why we use an evolutionary search approach for D-semble, and discuss the (1) training data fault injection, (3) diversity operators, (5) diversity-based heuristic functions, and (7) evolutionary search.

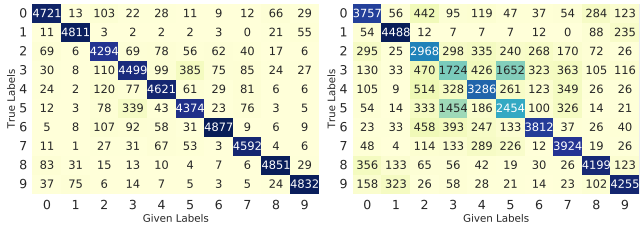
4.2 Fault Pattern Extraction and Injection

We explain how we extract the fault pattern (*i.e.* the noise transition matrix) using CIFAR-10 as an example. We use Cleanlab [42], a tool that deploys statistical methods to identify mislabelled data, to extract the fault pattern for CIFAR-10, resulting in an asymmetric matrix seen in Fig. 4a. Cleanlab identifies mislabelled data based on the confidence (*i.e.* softmax output) of any ML model trained on the dataset, and produces a list of probable mislabelled samples. We convert this list into a noise transition matrix (Fig. 4a).

$$g = \left(\sum_{i=1}^{N} \sum_{j=1}^{N} |x_i - x_j| \right) \Big/ 2N^2 \bar{x}. \tag{1}$$

Our fault injector injects faults based on the distribution in the noise transition matrix. More formally, let $T \in \mathbb{W}^2$ represent the extracted noise transition matrix from a dataset. We express fault injection as a function, $d : \mathbb{W}^2, \mathbb{R} \rightarrow \mathbb{W}^2$. Suppose we wish to perform fault injection based on T. A new noise matrix $T' \in \mathbb{W}^2$ is computed so it reaches a target fault amount, $\mathrm{f_{amt}}$, where $T' = d(T, \mathrm{f_{amt}})$. Synthetic noise is added to T by increasing the off-diagonals (mislabelled instances), while decreasing the diagonals (correctly labelled instances). For example, to inject $\mathrm{f_{amt}} = 30\%$ mislabelling faults into CIFAR-10, the resulting T' is shown in Fig. 4b.

We use the Gini coefficient [43] (g) to represent noise imbalance. The Gini coefficient measures the inequality of values, and ranges from 0 to 1, where 0 is for total equality (symmetric distribution) and 1 is for total inequality. Its equation is shown in Eq. (1) where N is the number of classes, and x is an off-diagonal element in the noise transition matrix. Let $g : \mathbb{W}^2 \rightarrow \mathbb{R}$. Since the noise imbalance must remain constant between the before and after matrices, $g(T) = g(T')$. This is because the off-diagonals in T' increase proportionate to each other while the diagonals decrease, as the mislabelling fault amount rises. In the example, g for both T and T' is 0.6.



(a) Extracted Fault Pattern (T) **(b) 30% Mislabelling (T')**

Figure 4: Noise Transition Matrices for CIFAR-10

4.3 State Space and Choice of Search Algorithm

We considered the scale of the state space to find a resilient ensemble across multiple diversity operators, model architectures, and fault amounts. Suppose we keep the fault type fixed and the ensemble size to be k, and let d represent the number of diversity operators, n represent the number of architectures, f_{amt} represent the number of fault amounts. The state space size is then $f_{\mathrm{amt}} \times \binom{n \times d}{k} \approx (n \times d)^k$, which is exponential, rendering exhaustive search impractical. As shown in Section 3, it is also paramount to pick resilient ensembles carefully. Therefore, we seek a search algorithm that finds the best solution, given some objective function, avoiding exhaustive search. GAs are a good choice, as they have been used to efficiently explore very large state spaces [16] in a variety of applications.

4.4 Diversity Operators

Diversity is a key component of ensembles' resilience against faulty training data by enabling misclassifications from one model to be overcome by others or causing models to fail in different ways, thus aiding failure detection (*i.e.* reverting to manual control in an autonomous vehicle) [7]. *We identify* three diversity operators (techniques to generate diversity in an ensemble) that were most effective at improving predictive capability over single models.

Architecture diversity [7, 61] is where models in the ensemble have different architectures. For instance, an architecturally diverse ensemble of size 3 may consist of MobileNet, ResNet50, VGG16. MobileNet has depthwise convolution layers, while ResNet50 has residual (skip) layers and VGG16 has regular convolution layers. We use a pool of 7 architecturally diverse models to generate diversity.

Data diversity, based on *bagging* [4], a highly resilient method against faulty training data [25], is where the same ML architecture is trained on different subsets of the dataset that are randomly sampled with replacement. We set the bagging percentage to 63% [4].

Snapshot diversity [21] is an ensemble generated from different convergence points (called *snapshots*) at local minima along the

loss function by the same ML architecture. Each time convergence is achieved before converging on a global minimum, the model may focus on certain input features. Snapshot diversity leverages these input feature specializations among snapshots. We use cosine annealing, which varies the learning rate in a cycle, to reach local minima along the loss function [32].

4.5 Heuristics: Ensemble Diversity Metrics

Diversity enables ensembles to be resilient against training data faults [7]. Therefore, we use the diversity value as a heuristic to select ensemble candidates at each round of the GA. Ensemble diversity metrics can be classified [28, 52] as *pairwise* and *nonpairwise*. Pairwise metrics measure diversity between two models, while non-pairwise metrics measure diversity on two or more models.

Our diversity metric is an average of a pairwise and a nonpairwise metric: the disagreement measure (Dis) [28] and the Shannon equitability index (H) [7]. We found that taking an average of both diversity metrics provided more stable values between runs. Suppose there are two models A and B in an ensemble, and there are N total inputs in the test set. $N_{A\bar{B}}$ denotes the number of inputs that were correctly classified by A, but misclassified by B. $N_{\bar{A}B}$ denotes the reverse case. Then, the disagreement measure (Dis) is given by:

$$\text{Dis} = \frac{N_{A\bar{B}} + N_{\bar{A}B}}{N} \tag{2}$$

The Shannon equitability index (H) is calculated on the ensemble prediction of a single test input. The H_{avg} denotes the average H over the entire test set. If S denotes the number of classes in a dataset, and p_i denotes the proportion of predictions belonging to class i, then H is given by:

$$H = -\left(\sum_{i=1}^{S} p_i \ln p_i\right) \Bigg/ \ln S. \tag{3}$$

Both metrics range between 0 and 1, where higher values indicate greater diversity. Hence, the final diversity heuristic formula used by D-semble is given by:

$$\text{Diversity} = 0.5 \times \text{Dis} + 0.5 \times H_{\text{avg}} \tag{4}$$

4.6 D-semble Implementation of the GA

Encoding. We demonstrate how D-semble encodes ensemble combinations for GA. Suppose we want to encode an ensemble of size $k = 3$, with a pool for seven initial architectures, and three diversity operators. The encoding is shown in Fig. 5 and has 21 total elements. The encoding is divided into seven elements for each diversity operator, resulting three divisions. The relative index i in m_i within a division indicates the model architecture. Suppose m_0, m_1, m_4 represent ConvNet, DeconvNet, ResNet50 respectively.

Each element represents the number of times that a diversity operator is applied on a model architecture. Elements can carry any value between '0' and the ensemble size, k. Architecture elements are limited to '0' or '1' as D-semble does not permit repeats of the same architecture and trained weights. Therefore, the encoding in Fig. 5 represents a three model ensemble consisting of a ConvNet, a DeconvNet trained on a data subset, and a snapshot of ResNet50.

Crossover and Mutation. D-semble uses the standard crossover and mutation operations as defined in GA [19]. As an example of

Figure 5: Example ensemble of size 3, encoded by D-semble with 7 initial architectures and 3 diversity operators.

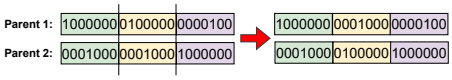


Figure 6: Crossover of 2 ensembles. Dividing lines show randomly chosen crossover points, depending on crossover rate.

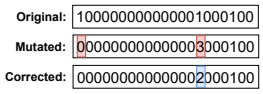


Figure 7: Mutation of an encoded ensemble. Indices are randomly chosen for mutation, depending on mutation rate.

crossover applied to two encoded ensembles (Fig. 6), two crossover points are randomly chosen, depending on the crossover rate, and are indicated by the two lines. D-semble applies two crossover points for every crossover operation. As an example of mutation on a single encoded ensemble (Fig. 7), each index is randomly chosen for mutation, depending on the mutation rate. Then the element value is swapped for another valid value (i.e., 0 to 1, or 1 to 3). The values represent whether a model is selected (non-zero value) and how many times a diversity operator is applied. In Fig. 7, the mutation deselects m_0 (ConvNet, trained for fixed epochs), and selects a snapshot of ConvNet (local convergence).

Constraint Correction. Ensembles created through crossover and mutation may violate the problem constraints. For example, when searching for ensembles of size 3, D-semble can sometimes generate ensemble combinations exceeding three models, as seen in Fig. 7. We implement *constraint correction* based on the concept of repairing constraint-violating candidates [37] by making corrections to candidates rather than discarding them - this also speeds up convergence. Constraint correction corrects the above case by randomly removing additional models. Constraint correction can also randomly add models to the ensemble, by activating elements, when the ensemble is short of the required size.

5 Evaluation

5.1 Research Questions (RQs)

(1) How are real-life faults distributed in datasets?
(2) What is the impact of different diversity operators on the resilience of ensembles?
(3) How do ensembles found by D-semble compare with baseline approaches?
(4) How does D-semble perform across fault types?
(5) How do ensembles found by D-semble perform relative to the search time, compared to other approaches?

5.2 Experimental Setup

Datasets. We use three datasets for our evaluation (Table 2): CIFAR-10, GTSRB, and Pneumonia. The German Traffic Sign Recognition Benchmark (GTSRB) [50], a training dataset for Autonomous Vehicles (AVs), contains more than 50,000 images belonging to 43 different types of traffic signs in Germany. The Pneumonia dataset [23],

Table 2: Image classification datasets and metrics used

Name	Dataset Size		Task (# Classes)	Evaluating
	Training	Test		Metric
CIFAR-10 [27]	50,000	10,000	Objects (10)	BA
GTSRB [50]	39,209	12,630	Traffic signs (43)	BA
Pneumonia [23]	5,239	624	Chest X-rays (2)	F_1

Table 3: Neural network architectures used

Name	Depth	Architecture Summary
ConvNet	Moderate	3 Conv + 3 FC + Max Pooling
DeconvNet	Moderate	4 Conv + 2 FC w/ 0.5 Dropout
VGG11	Deep	13 Conv + 3 FC + Max Pooling
VGG16	Deep	13 Conv + 3 FC + Max Pooling
ResNet18	Deep	17 Conv + 1 FC + Avg Pooling
MobileNet	Deep	27 Conv + 1 FC + Avg Pooling
ResNet50	Deep	49 Conv + 1 FC + Avg Pooling

consists of 5,863 X-ray images of paediatric patients in China. GTSRB and Pneumonia both represent safety-critical applications. Pneumonia is smaller than other datasets, as it is difficult to curate quality medical images [5]. We use F_1 to evaluate Pneumonia, a binary dataset, where false positives and negatives are more important than true negatives. We use BA for CIFAR-10 and GTSRB. We use the pre-defined training and testing splits in all three datasets.

All three datasets are *well-curated*, hence, we assume they have minimal inherent faults (< 5%) other than those that we inject. The golden model is hence trained on the original dataset. For example, images in CIFAR-10 were manually labelled [27], while images in the GTSRB dataset were labelled automatically but later verified by humans [50]. Images in the Pneumonia were verified by physicians.

We considered only image datasets as it was challenging to find adequately well-curated datasets for other applications. Additionally, we used relatively small datasets as we need to evaluate different ensemble search approaches in a reasonable time as we need to retrain the models from scratch after each fault injection.

Models. For our experiments, we used seven popular neural networks for image classification (Table 3) of varying architecture types and depth: ConvNet, DeconvNet, MobileNet, ResNet18, ResNet50, VGG11, and VGG16. We used these specific models for our evaluation as they have varying number of layer depths, and diverse architectural components (*i.e.* depthwise convolution layers in MobileNet and residual layers in ResNet models). Diverse networks provide a favourable starting point for ensemble search.

Fault Injection. We extend the TF-DM fault injector [40] to inject faults into training datasets. TF-DM supports the injection of mislabelling, removal and repetition faults. Mislabelling faults affect some label classes more than others; removal and repetition faults are equally likely to impact any label class. Thus, we use asymmetric fault distributions for mislabelling faults, and symmetric distributions for removal and repetition faults.

TF-DM injects faults of a selected fault type into a fixed proportion, indicated by the fault amount, of the training data, chosen at random. For mislabelling, we modified TF-DM so it performs stratified sampling based on label classes, and selects a fixed proportion of samples corresponding to the off-diagonal entry in the noise transition matrix. Then, for each selected sample, the original label is swapped with the target label.

We inject the three types of training data faults (Section 2.1), with one fault type and one fault amount per run. The fault amounts (10%-50%) resemble approximate observations in real datasets (Table 1).

Each model is first trained with fault-free training data to obtain a golden model. Then, the model is retrained with fault injected training data, resulting in faulty models. The predicted labels by the model against the actual labels in the test dataset to obtain BA.

We focus on ensembles consisting of three models, the minimum number required for simple majority voting, as there is a diminishing return for BA as more models are added [7], while incurring additional training and inference overheads. We experiment with larger ensembles of five and seven models (Section 5.8). All ensembles are constructed using individually trained models. Ensembles and individual models are all trained on identical training datasets, using the pre-defined training and testing splits.

Baselines. We select four baselines to compare with D-semble: bagging, greedy search, random selection, and the best individual model. Since bagged ensembles are very resilient (*e.g.* more than boosting) against faulty training data [25], we compare D-semble with the most resilient ensemble found through bagging, by configuring D-semble to only search ensembles where the data diversity operator is applied on the same candidate model. We select greedy and random search as baselines since they are faster and simpler than D-semble's GA, as they use neither GA nor diversity heuristics. Greedy search generates an ensemble by taking the most resilient individual model at each fault amount (*i.e.* 10%, 30%, 50%), and combining them together. Random selection, as the name suggests, obtains an ensemble by randomly sampling from all possible combinations of ensembles, across different diversity operators and architectures. We also select the best individual model for each fault configuration (dataset, fault type and amount) as a baseline to determine if ensembles provide any significant resilience. Individual models require the least effort to train and deploy.

Experimental Environment. For our experiments, we used a 64-bit AMD Ryzen Threadripper 3960X 24-Core Processor with 256GB RAM and three NVIDIA RTX 3070 GPUs. In total, training and inference took 15 days of computation time, as each ensemble search approach was run 10 times on each fault configuration to reduce its variance (*we computed 95% confidence intervals*).

5.3 D-semble Hyperparameters

D-semble is based on GAs, and hence we need to determine two hyperparameter values: crossover rate, p_c, and mutation rate, p_m. We experiment on CIFAR-10 to determine suitable values for p_c and p_m. However, similar results were found for the other datasets.

The first hyperparameter is p_c, a value between 0 and 1 controlling the rate at which elements between two candidate solutions are randomly swapped (Section 4.6). We set p_c to different values (0.25, 0.5, 0.75), and allow D-semble to search for the most resilient ensemble. We observe that higher values of p_c yield ensembles with higher BA, which are more resilient. Therefore, we set $p_c = 0.75$.

The second hyperparameter is p_m, a value between 0 and 1, which controls the rate at which elements in a candidate solution are randomly mutated (Section 4.6), to support exploration of new candidates. We set p_m to different values: 0.25, 0.5, and 0.75, and found that $p_m = 0.5$ returns ensembles with the highest BA. If p_m is too low, D-semble does not explore enough candidates to converge to a better solution. If p_m is too high, D-semble explores too many candidates and takes much longer to converge.

5.4 RQ1: Real-Life Fault Distributions

We focus on mislabelling faults and analyze their distribution in datasets. We omit removal and repetition faults due to their uniform distributions. We ask: *Does mislabelling occur symmetrically (uniformly) across label classes in a dataset or does there exist more frequent occurrences of mislabelling in certain classes (non-uniformly)?*

We utilize our fault pattern extractor (Section 4.2) to obtain the noise transition matrices and measure their Gini coefficients. The Gini coefficients for CIFAR-10, GTSRB, Pneumonia are 0.6, 0.95, 0.2 respectively. In addition to these datasets, we also analyze datasets that have been found to have large quantities of mislabelling such as Animal-10N [49] and Food-101N [30], and find that the Gini coefficients are 0.5 and 0.9. *Faults in real-life datasets are not symmetrically distributed across classes, but rather concentrated in specific label classes.* Pneumonia is the only exception as it only contains two label classes. We, therefore, base our experimental fault injection for mislabelling on asymmetric fault distributions extracted from the datasets. We manually verified Cleanlab [12] and our fault pattern extractor by comparing the noise transition matrix extracted from CIFAR-10, with that of CIFAR-10N [59], a version of CIFAR-10 manually relabelled by three human Amazon Mechanical Turk workers. We observe similar noise matrices.

Observation 1. *Mislabelling faults in real datasets are not symmetrically distributed, but concentrated in specific classes.*

5.5 RQ2: Resilience by Diversity Operator

To find which diversity operator provides the highest resilience, we applied the diversity operators, one at a time each, to generate ensembles of three models. Since we have 7 different neural network architectures, we produce a total of 49 ensembles, broken down into: $\binom{7}{3} = 35$ architecturally-diverse ensembles, 7 data-diverse ensembles and 7 snapshot-ensembles. The data-diverse and snapshot ensembles are generated with a single architecture each. For instance, a data-diverse ensemble is constructed by training ResNet50 on three random subsets of the training data.

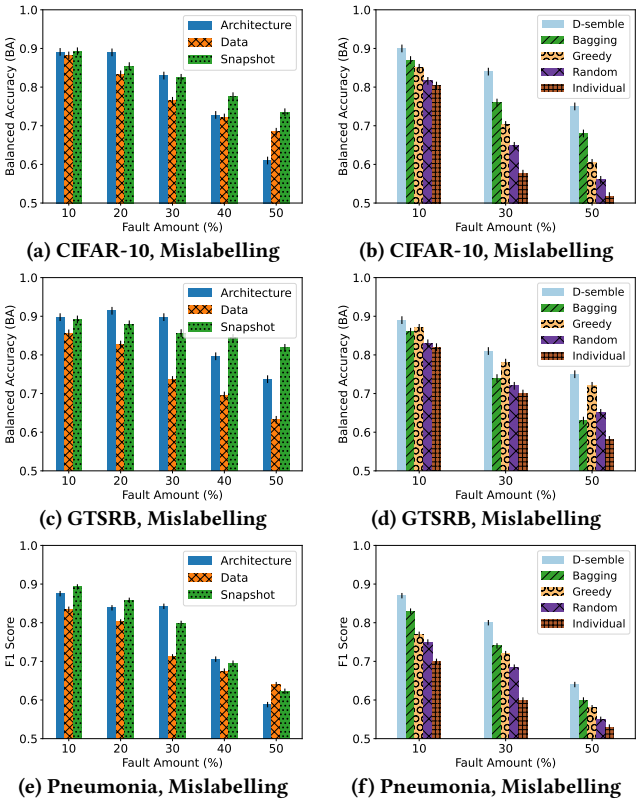
We performed 20 fault injections (for each fault amount and fault type) and measured the BA (F_1 for Pneumonia) of the most resilient ensemble, generated by each diversity operator. We report only mislabelling faults in this experiment (Figs. 8a, 8c and 8e), but the results were similar for other fault types. For example, the BA reported by the architecturally-diverse ensemble is the best out of 35 ensembles. As expected, we observe that both BA and F_1 decrease as the fault amount increases.

We find that architecturally-diverse ensembles are, overall, the most resilient, *i.e.*, highest BA or F_1. However, there are many exceptions, particularly for the Pneumonia dataset (Fig. 8e) and at higher fault amounts (50%) where the data-diverse and snapshot-diverse ensembles outperform the architecturally-diverse ensembles. Thus, *no individual diversity operator consistently generates ensembles of high BA or F_1*. Therefore, we need to search *all* candidate ensembles, across diversity operators, to find the most resilient ensembles.

Next, we examine correlation between ensemble diversity and resilience. We calculate ensemble diversities (Eq. (4) in Section 4.5). We ask: *Is diversity a good predictor of BA and F_1 under faults?*

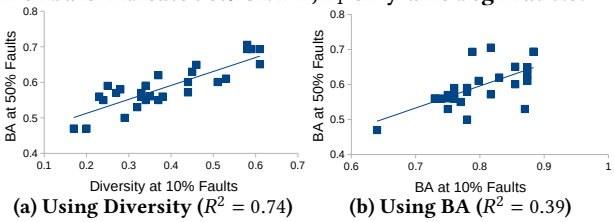
We perform a two-step experiment to answer this question. First, we plot the diversity of every ensemble at 10% faults against its BA at 50% faults, across CIFAR-10 and GTSRB (Fig. 9a). We compute the coefficient of determination, R^2, between diversity and BA. The computed R^2 value is 0.74, which indicates a moderate correlation.

Second, we plot the BA of every ensemble at 10% faults against its BA at 50% faults (Fig. 9b) and find that $R^2 = 0.39$, which denotes a weak correlation. Examining the data, we find that ensembles with high BA at lower fault amounts (*i.e.* 10%) may not also have high BA at higher fault amounts (*i.e.* 50%). In contrast, ensembles with high diversity at lower fault amounts are likelier to have more consistent BA values, thus avoiding drops in BA, even at higher fault amounts. Therefore, diversity is a better predictor of BA at



(a) CIFAR-10, Mislabelling (b) CIFAR-10, Mislabelling
(c) GTSRB, Mislabelling (d) GTSRB, Mislabelling
(e) Pneumonia, Mislabelling (f) Pneumonia, Mislabelling

Figure 8: BA, F_1 of the most resilient ensemble, generated by each diversity operator (a, c, e), trained on each dataset, injected with varying amounts of mislabelling. (b, d, f) BA, F_1 of the most resilient ensemble, found by each approach. Error bars indicate 95% CI. BA, F_1 on y-axis begin at 0.5.



(a) Using Diversity ($R^2 = 0.74$) (b) Using BA ($R^2 = 0.39$)

Figure 9: Scatterplots of Ensemble (a) Diversity at 10% Faults Vs BA at 50% Faults. Trend line: $y = 0.39x + 0.43$. (b) BA at 10% Faults Vs BA at 50% Faults. Trend line: $0.63x + 0.09$.

higher fault amounts, making it a more suitable heuristic for finding resilient ensembles. We find a similar result using F_1 in Pneumonia.

Observation 2. *No individual diversity operator consistently offers ensembles with the highest BA or F_1, across all configurations.*

Observation 3. *Diversity is more effective than BA and F_1 for finding resilient ensembles against faulty training data.*

5.6 RQ3: D-semble Vs Baseline Approaches

In the previous RQ, we established a need to search for resilient ensembles as no single diversity operator universally offers high resilience. We compare D-semble with four baselines (Section 5.2). We ask: *How does the resilience of ensembles generated by D-semble compare to that of ensembles generated by baseline approaches?*

We compare D-semble to baselines across all three datasets (Figs. 8b, 8d and 8f). We average the BA (F_1 for Pneumonia) of ensembles obtained by each approach (from 10 runs), across different fault amounts. As before, we focus on the results for mislabelling, but obtained similar results for other fault types. We bound D-semble's search time for each run to one hour, and report the BA or F_1 of the most resilient ensemble generated by then.

Ensembles generated by D-semble have the highest BA and F_1 across configurations. D-semble outperforms the second best approach, bagging (by 6% on average) in CIFAR-10 and Pneumonia (Figs. 8b and 8f) and greedy search (by 4% on average) in GTSRB (Fig. 8d). For GTSRB, bagging was less effective as extra models are required to achieve comparable performance to other approaches due to the larger number of classes. Random selection is the worst ensemble search approach but still outperforms the best individual model by 4% higher BA and F_1.

To statistically confirm our results, we perform single-tailed unpaired t-tests between D-semble and the two best alternatives, on each dataset separately, against a confidence interval (CI) of 95%. We show our analysis for GTSRB at 10% mislabelling, as it had the smallest range of BA among datasets (the most conservative case).

Let μ_D be the mean ensemble BA returned by D-semble, μ_G be greedy selection. Under the null hypothesis, $H_0 : \mu_D \leq \mu_G$. n, the sample size, is 10 runs. The mean and standard deviations for D-semble and greedy are (0.89, 0.02) and (0.87, 0.02) respectively. The p value is 0.0382, which is less than 0.05 (CI), thus, rejecting H_0. Comparing against bagging, the second best approach, $p = 0.02$, which is less than 0.05. Thus, D-semble is significantly better than both greedy selection and bagging in generating resilient ensembles.

Observation 4. *D-semble generates more resilient ensembles than bagging, greedy search, random selection and best individual model.*

5.7 RQ4: Performance across Fault Types

We examine D-semble's ability to find resilient ensembles against different types of training data faults. Previously, we focused on mislabelling faults due to space. Table 4 summarizes the BA and F_1 of ensembles found by D-semble across fault types, where the fault amount is 30%. D-semble is able to find resilient ensembles, with minor variations across fault types, in all three datasets.

We find D-semble has a lower BA and F_1 against both mislabelling and removal faults, compared to repetition. This observation is similar to how mislabelling and removal have a more pronounced effect on the BA of individual models [7]. Especially, in the case of Pneumonia, where there are fewer training samples, removal faults can have a significant impact on the individual model, and the subsequent ensemble. GTSRB is the exception, in which D-semble appears to find ensembles with consistently high BA across fault types. D-semble finds ensembles with higher diversity on average in GTSRB (0.56) compared to CIFAR-10 (0.37) and Pneumonia (0.17), which explains the higher average BA in GTSRB across fault types.

Table 4: Average BA and F_1 (Pneumonia) of ensembles found by D-semble across fault types, where the fault amount is 30%. Error bars represent 95% Confidence Intervals.

Dataset	Metric	Mislabelling	Removal	Repetition
CIFAR-10	BA	0.83 ± 0.02	0.82 ± 0.01	0.90 ± 0.01
GTSRB	BA	0.81 ± 0.02	0.85 ± 0.01	0.87 ± 0.01
Pneumonia	F_1	0.80 ± 0.02	0.76 ± 0.01	0.89 ± 0.01

Observation 5. *D-semble is able to find ensembles with high resilience across different fault types.*

5.8 RQ5: Performance by Search Time

We answer the question of how much BA can be achieved if the search time is bounded in D-semble. Because fault pattern extraction and training data fault injection is a one-time cost, we only consider the search time of ensembles. D-semble has no preset termination condition, apart from the number of iterations. If D-semble is run longer, more candidate ensembles can be explored, potentially finding more resilient ensembles. However, bagging, greedy search, random selection have a fixed runtime length as they terminate when an ensemble is either found or selected, respectively.

We also compare D-semble with itself without the diversity heuristic applied, denoted as 'D-semble w/o H'. In 'D-semble w/o H', D-semble selects the top candidates in each round, based on their BA (or F_1 for Pneumonia) instead of their diversity. This evaluation is to understand the impact of the diversity heuristic.

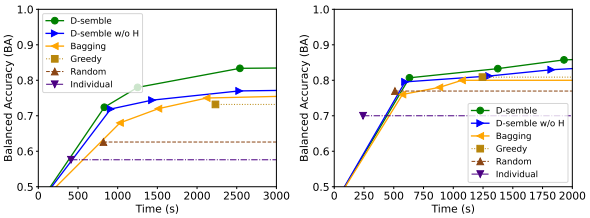
We show the BA, F_1 of the generated ensembles versus the time spent on ensemble search, using each approach (Fig. 10). For space reasons, we show only the results for three fault configurations. However, we obtained similar results for other configurations. We report the BA and F_1 at 30% faults. The best individual, greedy search and random selection results are shown as detached straight lines, as they have fixed times. D-semble, 'D-semble w/o H' and bagging are shown as line plots with multiple points, each representing the best ensemble found at different termination times.

In most cases, if D-semble is run for longer, the BA and F_1 of the found ensemble improves. However, there is a saturation point, beyond which increasing search time no longer improves the BA. *Overall, D-semble offers a much more resilient ensemble than the other baselines.* The ensemble found by D-semble is 9%, 16%, 28%, 32% more resilient than bagging, greedy search, random selection, the best individual model at 1.5×, 3×, 4×, 7× the search time respectively. Moreover, we see that D-semble always outperforms the baselines, especially the best individual model, when capped at a similar search time (Fig. 10a).

Furthermore, D-semble offers more resilient ensembles compared to the other techniques (Fig. 10), even without the diversity heuristic, as shown by 'D-semble w/o H'. However, 'D-semble w/o

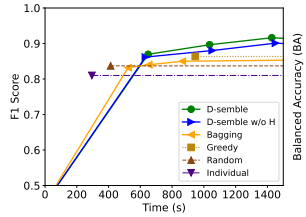
H' takes longer to reach saturation than D-semble. On average, 'D-semble w/o H' takes 1.4× time of D-semble to reach saturation, with a similar BA. Thus, the diversity heuristic enables better prediction of ensemble BA across different fault amounts rather than directly using the BA. Further, the diversity heuristic helps D-semble find more resilient ensembles with less search time.

While we considered ensembles consisting of only three models, we analyze how D-semble performs when searching for ensembles of different sizes in Fig. 10d. We configured D-semble so it searches for the most resilient ensemble from size 3 up to size k (=5,7). Due to the diversity heuristic, D-semble favours larger ensembles over smaller ones. Larger ensembles also exhibited higher BA compared to smaller ensembles initially. However, smaller ensembles enable searching for higher BA ensembles more quickly. Given sufficient search time, ensembles of any size converge to a similar BA.

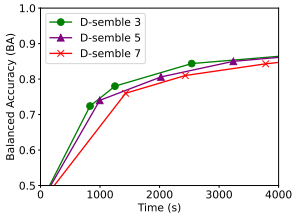


(a) CIFAR-10, Mislabelling 30%

(b) GTSRB, Removal 30%



(c) Pneumonia, Repetition 30%



(d) CIFAR-10, Mislabelling 30%, by Maximum Ensemble Size

Figure 10: BA and F_1 of generated ensemble by each approach versus search time spent, for each fault type and dataset. D-semble w/o H indicates the use of D-semble without the diversity heuristic applied. BA and F_1 on y-axis begin at 0.5.

Observation 6. *D-semble has higher search time than greedy and random search, but finds much more resilient ensembles.*

Observation 7. *The diversity heuristic allows D-semble to find more resilient ensembles faster.*

6 Discussion

Threat to Internal Validity. We assume labels in the test dataset match ground truth, which may not always hold. While we assume faults in training data, we do not make the same assumption for test data. As in software engineering, one does not assume that a program *and* its tests are faulty [18].

Threats to External Validity. We evaluated D-semble and other approaches separately against each fault type. In reality, datasets may contain multiple fault types. Evaluating D-semble against multiple fault types is an avenue for future work. Further, we evaluated D-semble on datasets that are much smaller (*i.e.* fewer than 100K training images) than most production datasets. As D-semble has to repeatedly train models on the fault injected datasets, D-semble will take longer to find resilient ensembles on larger datasets. Scaling D-semble to larger datasets is also direction for future work.

Threats to Construct Validity. Genetic algorithms (GAs) can occasionally be stuck at local optima. While this can be remedied by mutation, local optima can still occur if the mutation rate is set too low (Section 5.3). Alternatives to GAs such as simulated annealing may overcome this problem. This is a subject for future work.

D-semble relies on Cleanlab to correctly identify mislabelled data, but Cleanlab may report false positives and false negatives. We mitigate this risk by raising the reporting confidence threshold, and manually verifying the samples of reported mislabelling.

Applicability to Other ML Tasks and Data Modalities. While our focus is on image classification, ensembles have been applied in object detection and text sentiment analysis [1, 6], so we believe our results will be more broadly applicable. We focus on image classification-based ML tasks for three reasons. First, these datasets are highly mature [13]. Second, unlike object detection or image segmentation, each entry in image classification datasets only consists of an image and a single label, making it ideal to inject faults for evaluation. Finally, image classification is supported by a wider variety of ML architectures than other tasks. Thus, there is a stronger motivation to search for resilient ensembles.

7 Related Work

ML with Faulty Training Data. Faulty training data has been identified as a major threat to ML deployment, as it can seriously degrade accuracies of trained models [45]. While data cleaning has been proposed to remove faults [30], faulty training data can persist. Prior work has instead focused on tolerating training data faults [8], which include knowledge distillation [62], label smoothing [34], robust loss functions [35], and ensembles [7, 61]. Ensembles were found to be the most resilient among these techniques [8]. However, finding the combinations of models for ensembles still requires practitioner effort and heavy computational resources for training.

Neural Architecture Search (NAS). NAS automates the search for a single neural network architecture for a given task, which is related to ensemble search [33, 63]. While NAS has been largely explored under fault-free training data, NAS returns architectures with significantly lower accuracy under faulty training data [48]. Improved NAS methods incorporating robust regularization, and injecting noise during NAS [29] have been proposed. However, these techniques are dependent on the specific fault type and fault amount. In contrast, D-semble makes no assumptions on the fault amount, and is effective across common types of training data faults.

Ensemble Optimization. Ensemble optimization has focused on reducing training redundancy of networks, while boosting accuracy, with fewer models [21]. In contrast, we aim to efficiently find ensembles that maximize resilience against faulty training data. To do so, models in the ensemble have to be repeatedly trained against fault injected datasets, rendering the search space much larger than finding accurate ensembles against fault-free datasets.

Genetic Programming (GP). GP using diversity as a heuristic has been explored for traditional software engineering (*i.e.* non-ML programs), including fault tolerance [15] and program repair [44]. Feldt [15] found that GP yields the most diverse three-version programs, though, not always the lowest failure rates. In contrast, we

apply genetic algorithms to find resilient ML ensembles. Additionally, Qi et al. [44] find random search can occasionally outperform GP - this is why we use random search as one of our baselines.

8 Conclusions

Faulty training data can reduce the predictive capability of ML applications. Thus, ML applications need to be resilient against training data faults. Ensembles have been found to be highly resilient to training data faults but finding resilient ensembles is time-consuming due to the large search space. We introduce D-semble, a genetic-algorithmic approach to efficiently construct resilient ensembles. D-semble generates 9%, 16%, 28%, 32% more resilient ensembles compared to bagging, greedy search, random selection, and the best individual model respectively within reasonable time. D-semble's ensembles are also resilient across fault types.

9 Acknowledgements

We thank the anonymous reviewers of SAC'25. This work was funded in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) and a Four Year Fellowship from UBC.

References

[1] Ankit and Nabizath Saleena. 2018. An Ensemble Classification System for Twitter Sentiment Analysis. In *Proc. of ICCIDS*.
[2] Emmanuel Asamoah. 2023. *Genetic Algorithm-Based Improved Availability Approach for Controller Placement in SDN*. Ph. D. Dissertation.
[3] S. S. Banerjee et al. 2018. Hands Off the Wheel in Autonomous Vehicles?: A Systems Perspective on over a Million Miles of Field Data. In *Proc. of DSN*.
[4] L. Breiman. 1996. Stacked regressions. *Mach Learn* (1996).
[5] Canadian Assoc. of Radiologists. 2019. CAR White Paper on Ethical and Legal Issues Related to Artificial Intelligence in Radiology. *CAR Journal* (2019).
[6] A. Casado-García and J. Heras. 2020. Ensemble Methods for Object Detection. In *Proc. of ECAI*.
[7] Abraham Chan et al. 2021. Understanding the Resilience of Neural Network Ensembles against Faulty Training Data. In *Proc. of QRS*.
[8] Abraham Chan et al. 2022. The Fault in Our Data Stars: Studying Mitigation Techniques against Faulty Training Data in Machine Learning Applications. In *Proc. of DSN*.
[9] Joseph Chee Chang et al. 2017. Revolt: Collaborative Crowdsourcing for Labeling Machine Learning Datasets. In *Proc. of CHI*.
[10] Liming Chen and Algirdas Avizienis. 1978. N-version programming: A fault-tolerance approach to reliability of software operation. In *Proc. of FTCS'78*.
[11] Nancy Chinchor et al. 1993. MUC-5 evaluation metrics. In *Proc. of MUC*.
[12] Cleanlab. 2024. Cleanlab. https://github.com/cleanlab/cleanlab.
[13] Jia Deng et al. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *Proc. of CVPR*.
[14] Brad Dwyer. 2020. A popular self-driving car dataset is missing labels for hundreds of pedestrians.
[15] R. Feldt. 1998. Generating Diverse Software Versions with Genetic Programming: an Experimental Study. *IEE Proceedings - Software* (1998).
[16] Patrice Godefroid and Sarfraz Khurshid. 2002. Exploring Very Large State Spaces Using Genetic Algorithms. *STTT* (2002).
[17] Muhammad Hamzah. 2020. https://github.com/mdhmz1/Auto-Annotate.
[18] Dan Hao et al. 2013. Is This a Bug or an Obsolete Test?. In *Proc. of ECOOP*.
[19] John H. Holland. 1992. Genetic Algorithms. *Scientific American* (1992).
[20] Pei-Yun Hsueh et al. 2009. Data Quality from Crowdsourcing: A Study of Annotation Selection Criteria. In *Proc. of HLT*.
[21] Gao Huang et al. 2017. Snapshot Ensembles: Train 1, get M for free. In *Proc. of ICLR*.
[22] Daniel Kang et al. 2022. Finding Label and Model Errors in Perception Data With Learned Observation Assertions. In *Proc. of SIGMOD*.
[23] Daniel Kermany et al. 2018. Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images for Classification. https://data.mendeley.com/datasets/rscbjbr9sj/2.
[24] R. Kesten et al. 2020. Lyft L5 Perception Dataset. https://level5.lyft.com/dataset/
[25] Taghi Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. 2011. Comparing Boosting and Bagging Techniques With Noisy and Imbalanced Data. *Trans. on Sys, Man, and Cyb.* (2011).
[26] Dominik Kreuzberger, Niklas Kühl, and Sebastian Hirschl. 2022. MLOps: Overview, Definition, and Architecture. *IEEE Access* (2022).
[27] Alex Krizhevsky. 2009. *Learning multiple layers of features from tiny images*. Technical Report.
[28] Ludmila Kuncheva et al. 2003. Measures of Diversity in Classifier Ensembles and Their Relationship with the Ensemble Accuracy. *Mach Learn* (2003).
[29] Mathias Lecuyer et al. 2019. Certified Robustness to Adversarial Examples with Differential Privacy. In *Proc. of S&P*.
[30] Kuang-Huei Lee et al. 2018. CleanNet: Transfer Learning for Scalable Image Classifier Training with Label Noise. In *Proc. of CVPR*.
[31] Tsung-Yi Lin et al. 2014. Microsoft COCO: Common Objects in Context. arXiv:1405.0312.
[32] Ilya Loshchilov and Frank Hutter. 2017. SGDR: Stochastic Gradient Descent with Warm Restarts. In *Proc. of ICLR*.
[33] Zhichao Lu et al. 2019. NSGA-Net: Neural Architecture Search Using Multi-Objective Genetic Algorithm. In *Proc. of GECCO*.
[34] Michal Lukasik, Srinadh Bhojanapalli, Aditya Krishna Menon, and Sanjiv Kumar. 2020. Does label smoothing mitigate label noise? arXiv:2003.02819.
[35] Xingjun Ma et al. 2020. Normalized Loss Functions for Deep Learning with Noisy Labels. In *Proc. of ICML*.
[36] F. Machida. 2019. N-Version Machine Learning Models for Safety Critical Systems. In *Proc. of DSNW19*.
[37] Zbigniew Michalewicz et al. 1996. Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation* (1996).
[38] Jamie Murdoch. 2022. https://medium.com/p/79d382edf22b.
[39] Sasu Mäkinen et al. 2021. Who Needs MLOps: What Data Scientists Seek to Accomplish and How Can MLOps Help?. In *Proc. of WAIN*.
[40] Niranjhana Narayanan and Karthik Pattabiraman. 2021. TF-DM: Tool for Studying ML Model Resilience to Data Faults. In *Proc. of DeepTest*.
[41] Curtis G. Northcutt et al. 2021. Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks. arXiv:2103.14749.
[42] Curtis G. Northcutt et al. 2022. Confident Learning: Estimating Uncertainty in Dataset Labels. In *Proc. of NeurIPS*.
[43] Graham Pyatt. 1976. On the Interpretation and Disaggregation of Gini Coefficients. *The Economic Journal* (1976).
[44] Yuhua Qi, Xiaoguang Mao, Yan Lei, Ziying Dai, and Chengsong Wang. 2014. The Strength of Random Search on Automated Program Repair. In *Proc. of ICSE*.
[45] Thomas C Redman. 2018. If Your Data is Bad, Your Machine Learning Tools Are Useless. *Harvard Business Review* (2018).
[46] Jonathan G Richens et al. 2020. Improving the accuracy of medical diagnosis with causal machine learning. *Nature Communications* (2020).
[47] Nithya Sambasivan et al. 2021. "Everyone wants to do the model work, not the data work": Data Cascades in High-Stakes AI. In *Proc. of CHI*.
[48] Christian Simon et al. 2022. Towards a Robust Differentiable Architecture Search under Label Noise. In *Proc. of WACV*.
[49] Hwanjun Song, Minseok Kim, and Jae-Gil Lee. 2019. SELFIE: Refurbishing Unclean Samples for Robust Deep Learning. In *Proc. of ICML*.
[50] J. Stallkamp et al. 2012. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks* (2012).
[51] Bob L. Sturm. 2013. The GTZAN dataset: Its contents, its faults, their effects on evaluation, and its future use. (2013). arXiv:1306.1461.
[52] E. Tang et al. 2006. An analysis of diversity measures. *Mach Learn* (2006).
[53] Siyi Tang et al. 2021. Data Valuation for Medical Imaging Using Shapley Value: Application on A Large-scale Chest X-ray Dataset. *Scientific Reports* (2021).
[54] G. Tzanetakis et al. 2002. Musical genre classification of audio signals. *IEEE Trans. on Speech and Audio Processing* (2002).
[55] Udacity. 2016. Self-Driving Car. https://github.com/udacity/self-driving-car
[56] Ryan Urbanowicz and Jason Moore. 2015. ExSTraCS 2.0: Description and Evaluation of a Scalable Learning Classifier System. *Evolutionary Intelligence* (2015).
[57] Kazuya Wakigami, Fumio Machida, and Tuan Phung-Duc. 2024. Empirical architecture comparison of two-input machine learning systems for vision tasks. *Form. Asp. Comput.* (2024).
[58] Xiaosong Wang et al. 2017. ChestX-Ray8: Hospital-Scale Chest X-Ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases. In *Proc. of CVPR*.
[59] Jiaheng Wei et al. 2022. Learning with Noisy Labels Revisited: A Study Using Real-World Human Annotations. In *Proc. of ICLR*.
[60] Qiang Wen, Júlio Mendonça, Fumio Machida, and Marcus Völp. 2024. Enhancing Autonomous Vehicle Safety through N-version Machine Learning Systems. In *Proc. of AISafety*.
[61] Hui Xu et al. 2019. NV-DNN: Towards Fault-Tolerant DNN Systems with N-Version Programming. In *Proc. of DSNW*.
[62] Linfeng Zhang et al. 2019. Be Your Own Teacher: Improve the Performance of Convolutional Neural Networks via Self Distillation. arXiv:1905.08094.
[63] Barret Zoph and Quoc V. Le. 2017. Neural Architecture Search with Reinforcement Learning. In *Proc. of ICLR*.