Article Type: Data

Efficient Data Augmentation for Machine Learning Classifiers Using Horizontal and Vertical Contractions

Abdul Majeed, Noor Munir, and Seong Oun Hwang, *Gachon University, Seongnam 13120, Republic of Korea*

Abstract—To yield better performance, machine learning (ML) classifiers are typically trained with augmented data (i.e., combining real data with synthetically generated data). While the amalgamation of real data with synthetic data makes intuitive sense and can be useful in enhancing the performance of ML classifiers, the addition of synthetic records beyond a certain limit destroys the truthfulness of the real data and can lead to performance bottlenecks. To address these problems, we propose and implement an efficient data augmentation technique based on vertical and horizontal contractions, which adds as few records as possible (under salient features only), whereas existing methods augment everything, leading to high computation costs. Our technique identifies suitable features and regions for augmentation without losing guarantees on performance, rather than needlessly adding more records. Experiments with three real-world datasets prove the efficacy of our technique in terms of computing time, accuracy, and model complexity.

Machine learning (ML) classifiers have demonstrated their effectiveness in solving many real-world classification and prediction problems. For instance, ML classifiers can be used to determine if a person is healthy or not based on data collected in hospital environments. Similarly, they can predict the onset of multiple types of disease by leveraging diverse medical data. In many sectors, a vast amount of multidimensional and multi-faceted data is generated, offering many opportunities for downstream tasks (e.g., classifying healthy individuals from unhealthy ones based on statistical features, classifying transactions into fraud versus legitimate ones, etc.) when analyzed with advanced data mining or ML techniques. However, the proper use of complex data in ML classifiers is very challenging owing to quality-related factors such as incomplete tuples, missing labels, wrong values for some features, and redundant tuples. ML classifiers developed with low-quality data cannot correctly classify/predict unseen data, leading to poor generalizations. Also, the use of ML classifiers trained on poor-quality data in some high-stakes applications (e.g., medical diagnosis, dangerous event prediction) can be very risky. Recently, ensuring the quality of data in the entire life cycle of an ML-based project has become a very hot research topic.

Motivation and Research Gaps

Since the inception of data-centric artificial intelligence (DC-AI), data quality enhancement has received considerable attention from the AI community. The DC-AI concept was introduced by Andrew Ng, and its essence is to pay higher attention to the data, rather than fiddling only with the code of AI models [1]. By using DC-AI, Hegde [2] achieved 100% accuracy with time series data in an anomaly detection scenario. Jeczmionek and Kowalski [3] adopted the DC-AI concept and accomplished the task of model training using fewer data without degrading accuracy. Similarly, many such solutions have been proposed for diverse domains where either the data quality is poor or better quality data does not exist [4]. DC-AI encompasses many sophisticated techniques to make the data representative of the problems being solved.

Motivated by the DC-AI concept, we explore the invisible risks of data augmentation techniques in terms of computing burden and corresponding performance improvements when used as a supplement in ML classifiers. We found that most of the existing data augmentation techniques focus on more records addition

XXXX-XXX © 2024 IEEE
Digital Object Identifier 10.1109/XXX.0000.0000000

in order to yield higher performance from ML classifiers [5], [6]. Although the addition of more records without any legitimate criteria yields better performance in terms of accuracy from ML classifiers, data truthfulness (e.g., each sample in training data should correspond to an existing phenomenon in reality) cannot be sustained when abundant synthetic records are added to real data. Maintaining data truthfulness while adding more records is a critical problem and requires more robust solutions to prevent negative consequences in terms of data reliability and computing overhead. The existing data augmentation methods have four problems while balancing the data.

- Many methods add more records in all classes of the data, leading to a significant rise in volume [6]. Adding more records in all classes can lead to serious performance problems while minimally enhancing accuracy, especially when the data volume is large.
- Some methods add records only to minority classes but without identifying suitable regions, leading to poor separability and noise in the augmented data [7].
- Some data augmentation methods consider the curse of dimensionality but only after adding a large # of records, leading to high cost with only minimal improvements in results [8].
- Many of the existing methods do not explore ways to reduce the # of features before augmentation, leading to higher computing costs while marginal improvements in accuracy results.

Solutions to the above problems are imperative because ML models are employed in many high-stakes and safety-critical scenarios nowadays.

Major Contributions

In this work, we extend our previous work [9] by making it well-suited to high-dimensional and large datasets. In [9], only vertical contraction was applied and impacts of data refinements in terms of the classifier's complexity were not assessed. In this work, we make the previous implementation more customized and investigate the impact of data augmentation in terms of classifiers' complexity while sustaining accuracy close to our previous work [9]. This work reduces the data dimensionality before the augmentation process; therefore, the subsequent steps are more lightweight than our prior work [9]. Our major contributions are as follows.

- We delve into performance bottlenecks caused by vertical data augmentation (e.g., fewer records addition, but under all columns) when it is used to supplement ML models, and we explore opportunities to develop an efficient data augmentation technique that significantly reduces computing overhead while improving the accuracy of ML models.
- We introduce the concepts of vertical contractions (e.g., feature selection) and horizontal contractions (e.g., adding fewer records) to overcome the computing burdens in data augmentation without compromising accuracy.
- We amalgamate the Boruta algorithm and a conditional generative adversarial network (CGAN) to address the curse of dimensionality and good-quality synthetic data generation problems to accomplish data augmentation so the resulting data are complete, perfect, and representative of the problem being solved.
- We integrate a noise removal mechanism in the data augmentation process to ensure that bad samples are filtered out before the training process to foster the learning ability of classifiers.
- We conducted detailed experiments on three benchmark datasets. The experimental analyses and comparisons reveal better results from our technique than from state-of-the-art methods.

Our technique yielded superior results with a significantly reduced parameter size, and therefore, it can be used in resource-constrained environments.

Preliminaries and Problem Overview

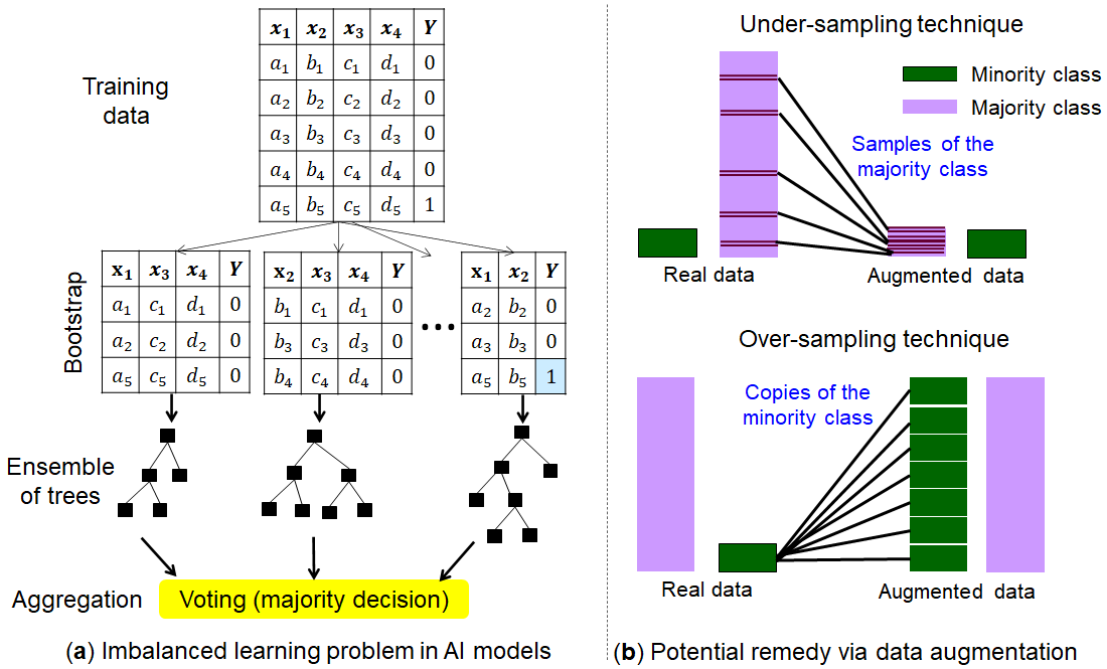
Preliminaries

In this section, we discuss the imbalanced learning problem and its conventional solution. Fig. 1 demonstrates an example of imbalanced learning, where 0 is a majority class, and 1 is a minority class. In this example, set X where $X = \{x_1, x_2, \ldots, x_n\}$ denotes the predictors, and Y is the target class having a cardinality of 2 (i.e., $Y = y_1, y_2$). The generic data model considered in this work is Eq. 1.

$$T = \begin{pmatrix} & x_1 & x_2 \cdots & x_n & Y \\ 1 & x_1^1 & x_1^2 \cdots & x_1^n & y_1 \\ 2 & x_2^1 & x_2^2 \cdots & x_2^n & y_2 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ N & x_N^1 & x_N^2 \cdots & x_N^n & y_1 \end{pmatrix} \quad (1)$$

As shown in Fig. 1(a), most of the samples drawn from the data have an imbalanced distribution of Y that can lead to imbalanced learning of classifiers. For example, out of three samples, only one has representation from both classes. In large-size samples, there is a possibility of wide gaps in the distribution of both classes, leading to improper learning during training. It is worth

This article has been accepted for publication in Computing in Science & Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MCSE.2025.3563355



(**a**) Imbalanced learning problem in AI models (**b**) Potential remedy via data augmentation

FIGURE 1. (a) Example of the imbalanced learning problem in classifiers (0 is a majority class, and 1 is a minority class), and (b) a potential remedy via data augmentation.

noting that using a large number of trees or optimized parameters cannot address these data-oriented problems. The classifiers trained on an imbalanced dataset can yield poor results in some applications, such as medical diagnoses, especially when a binary decision is involved [7]. A concrete solution for such data-tailored problems is necessary when ML techniques are used within digital twins. In the literature, two common solutions, illustrated in Fig. 1(b), have been suggested to address these problems: random over-sampling (ROS) and random under-sampling (RUS). These solutions are defined as follows.

1) ROS: Upgrade the minority class distribution by expanding the data (e.g., acquire more samples by leveraging the information of the existing samples, or curate synthetic data via generative AI).
2) RUS: Downgrade the samples from the majority class by removing samples until the # of samples equals the # of samples in the minority class.

Many improved versions of these techniques have been developed, but most of the recent developments have the problems explained in the previous section. In some cases, an ML model can yield a significantly higher accuracy (e.g., $\geq$ 90%), but the corresponding confusion matrix can be centered toward the majority class only, leaving the minority class learned only minimally. Based on our experimental findings from a benchmark stroke prediction dataset[1] in medical research, we found accuracy to be $\geq$ 95% but only true positives were high (e.g., 94.46% of the contributions to accuracy came from one class). We believe this is common with datasets encompassing target classes of a binary nature (i.e., two values only). The situation is serious when such a classifier is deployed in cloud environments for medical data analytics, or deployed by hospitals for medical diagnosis of cancer patients. Furthermore, in some cases, pre-processing is applied to the data, but only general problems (e.g., outliers or missing values) are fixed, leading to very small improvements in the results.

All of the above data-related challenges necessitate more practical and robust data-quality enhancement techniques in the AI era. In recent years, many data augmentation techniques have been developed, but most of them are less efficient and only marginally improve the results. Since the quality of data is directly related to ML model performance, more efforts are required for ML-ready data production. State-of-the-art techniques to address data quality-related problems

[1] https://www.kaggle.com/code/ahmedashrafahmed/stroke-prediction-dataset

have been proposed [10], [11], [12]. Although these techniques help balance the data, they can lead to changes in semantics from augmented data, which might lead to biased results. Furthermore, these techniques do not curate good-quality data with better diversity to repair only problematic portions of the data. To the best of our knowledge, none of these techniques have considered selective augmentation to lower computing overhead while sustaining accuracy. Consequently, the above-cited technical problems cannot be resolved in a fine-grained manner by using these techniques. Our technique attempts to fill this gap by amalgamating Boruta and the CGAN model with a noise reduction method.

Problem formulation

The key problem we solve with our technique is formally expressed as follows.

Problem 1. *Given a real-life dataset, D, with $n \times d$ dimensions that encompass various features (name, age, race, glucose level, etc.) and target classes (e.g., disease/income), there can be various quality-related challenges in D such as a limited # of samples, incomplete samples, outliers, duplicate records, highly skewed target class distributions, etc. How to curate a best-quality representative dataset, $\mathcal{D}$, from D where (a) $\mathcal{D} \subseteq D$, where $\mathcal{D}$ encompasses salient features only, rather than all features; (b) $\mathcal{D}$ has excellent quality (e.g., balanced distributions, complete tuples, no outliers, sufficient records, no duplicate records, and consistent values under each feature); (c) $\mathcal{D} \sim D$ (e.g., the knowledge/conclusions drawn from $\mathcal{D}$ are nearly the same as they are from D); (d) $\mathcal{D}$ yields higher accuracy than D; and (e) $\mathcal{D}$ significantly lowers the computing burden when classifiers are trained on it?*

Proposed Technique: Boruta and CGAN-based Data Augmentation

In many real-world applications, the data can encompass many features and some of those can be irrelevant, owing to poor correlation with the target class. These irrelevant features do not contribute to prediction accuracy and instead increase the computing cost. Similarly, in some cases, the data is inadequate to train ML classifiers, and more data is required before training them. To solve dimensionality reduction (removing less useful features/columns) and to increase data size (e.g., increasing # of rows), many tools have been recently developed. This work adopts two technical tools for accomplishing this task: Boruta for dimensionality reduction, and CGAN to curate more data for compensating the data deficiency. Boruta analyzes the correlation between features and the target class and figures out the features that exhibit poor correlation with the target class. The CGAN analyzes the distributions of the real data under some conditions and produces synthetic records that are very similar to real data.

The Boruta and CGAN-based data augmentation technique is imperative in order to address the class imbalance problem but with minimal overhead. In our technique, a noise removal method is also used, which increases the robustness of ML models in terms of generalization, learning, and balanced sampling. Fig. 2 demonstrates the conceptual overview of our proposed technique. This technique is quite different from our previous work, which only considers the vertical contractions [9]. In contrast, this technique first reduces the dimensions of data which makes the remaining steps highly customized and lightweight. For example, the data generation process is customized to curate data for some features only whereas our existing method curates data for all classes as well as features, leading to performance bottlenecks when data dimensions are high. In addition, the comparison for each record at noise removal time is higher in our previous technique. Lastly, this work offers opportunities to reduce hyperparameters of ML classifiers by reducing data in both directions, which was not possible in our previous work. There are five key steps, which are discussed below.

Data engineering: In the first step (Fig. 2 Part 1a,1b), data are cleaned through sophisticated preprocessing techniques, and vulnerability analysis is performed. We apply seven techniques to pre-process D. At the start, outliers are removed from the numerical features encompassed in D via the $min-max$ method. Later, outliers from categorical features are removed by analyzing each feature's domain values and identifying illegitimate values. For example, the gender feature can have two legitimate values (male or female). We also regard values occurring with very low frequency as outliers in order to improve their frequency later. Missing values are a very common problem with many real-life examples of D. To solve this problem, we propose Algorithm 1.

In Algorithm 1, N and D are given as input, and $\mathcal{D}$ is obtained as output. The missing values in numerical features are imputed via the mean of their respective columns, whereas the missing columns in non-numerical features are imputed via minority values to increase their size. Later, $\mathcal{D}$ with complete information is curated. The cost of steps 7 and 8 in Algorithm 1 depends on the size of $\mathcal{D}$ as well as the # of unique values in categorical features. However, it is not very high when computed with the help of the count function enclosed in built-in R packages of *plyr* and *vctrs*.

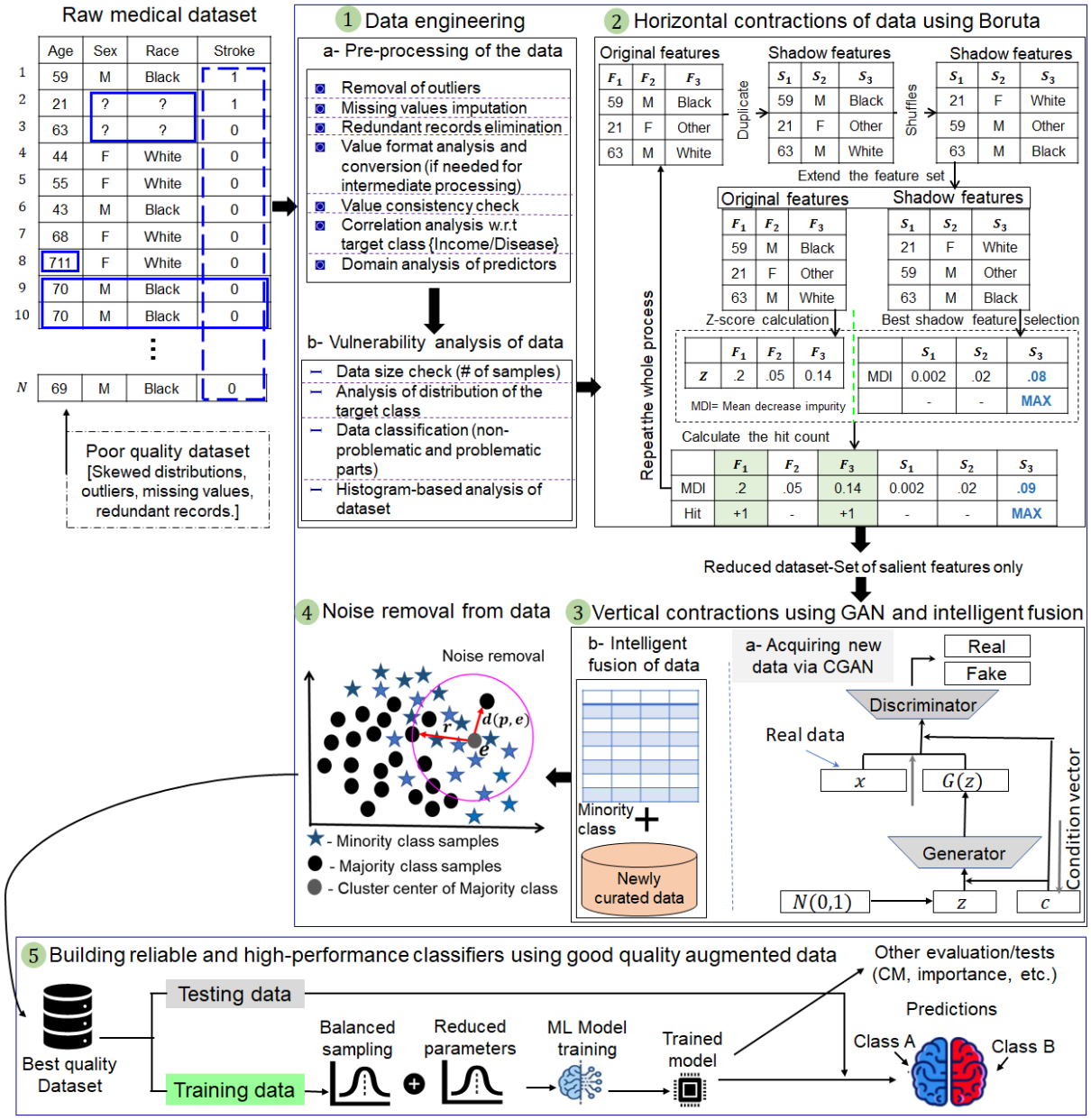


FIGURE 2. Conceptual overview of the proposed horizontal- and vertical-contractions-based data augmentation technique (Extension of Majeed et al. [9]). The major difference is step # 2, which makes the subsequent steps highly optimized.

Through experiments, we found that the worst-case cost of both these steps for three chosen datasets is 0.16s (Stroke), 0.4s (Adults), and 1.92s (Census).

To remove redundant tuples, we use the cosine similarity measure. We compute similarities among features and remove tuples with a similarity score of 1 that are located next to each other. Via the *typeof*(x_i) command, we also perform a consistency check to ensure that values for each feature are consistent with the feature type. Some feature values (e.g., dates) might not be in a unified format; therefore, they can be converted into a unified format at this stage. We perform a correlation analysis of X with Y to prevent the removal of salient features in later stages. These steps help improve D's quality.

In the next step (Fig. 2 Part 1b), we inspect the types of vulnerabilities that exist in D. There can be many advanced types of vulnerabilities in D, i.e., an

Algorithm 1 Imputing missing values in D (adopted from [9]).

Require: D, N
Ensure: $\mathcal{D}$ ▷ $\mathcal{D}$ has complete values of features.
$\mathcal{D} \leftarrow 0$, $X \leftarrow \{x_1, x_2, ... x_n\}$ ▷ initializing $\mathcal{D}$ and X.
for $i = 1$ *to* n **do**
  if (x_i == *numeric*) **then**
    $\bar{x}_i \leftarrow \sum_1^N x_i / N$ ▷ calculate mean.
    Impute missing numerical values with $\bar{x}_i$
  else if (x_i == *discrete*) **then**
    $D_{x_i} \leftarrow unique(D(x_i))$ ▷ Find unique values.
    $f(D_{x_i}) \leftarrow |D|$, where $x_j == x_i$ ▷ x_i frequency.
    $\chi_1 \leftarrow max(f(D_{x_i}))$ ▷ Majority values.
    $\chi_2 \leftarrow min(f(D_{x_i}))$ ▷ Minority values.
    Impute missing categorical values with χ_2
While (D have no missingness in all features) do
$\mathcal{D} \leftarrow \mathcal{D} \cup D$ ▷ Curating complete data.
End While
Return $\mathcal{D}$ ▷ Data for further processing.

inadequate number of samples, an imbalance w.r.t. target class, big gaps in values, and skewed distributions. Specifically, we implement an algorithm to check for class imbalance problems, classifying data into good and bad parts for augmentation. The imbalance in the target class can be determined by counting the class examples and using $\mathcal{IR}$, as shown below.

$$\mathcal{IR} = c_M / c_m \tag{2}$$

where c_M and c_m denote the total # of samples in the major and minor classes, respectively. The $\mathcal{IR}$ close to 1 indicates that the data is balanced w.r.t classes.

The vulnerability related to D's size can be determined by setting up a threshold and comparing N with it. The above process can help identify the types of vulnerability in D, subsequently leading to an effective solution.

Horizontal contractions of the data using Boruta: Since not all features are required in classification/regression tasks, some features can have a poor correlation with Y. The removal of less informative features enhances the accuracy of classifiers and reduces computing overhead. In this step, we leverage the R implementation of Boruta algorithm[2] to solve the curse of dimensionality (i.e., horizontally contracting $\mathcal{D}$). Boruta finds highly informative features by contrasting the original features' importance with the importance accomplished at random, estimated via their permuted versions (shadows). It uses Z score, mean decrease impurity (MDI), and hit count during the evaluation process. The basic formula for the Z score is: $Z = {}^{x-\mu}/_{\sigma}$. If x represents the score of features, then this formula simply tells how many standard deviations this score is above/below the mean score in D. MDI counts the times a feature is leveraged to split a tree node, weighted by the # of samples it splits. Hit count simply records the number of hits for each feature when the Z score is higher than the shadow features. The higher hit count indicates that a feature is important and vice versa. The holistic process of extracting relevant features from $\mathcal{D}$ is illustrated in Fig. 2 Part 2. With the help of this process, the horizontal dimension of $\mathcal{D}$ is reduced, and only salient features are retained. The horizontal contraction process (e.g., data refinement) is the reduction of features in data as demonstrated in Figure 3. In this example, there are five features (e.g., x_1 to x_5) and a target class (y), and when we apply Boruta algorithm to this data, and if turns out that two features (e.g., x_2 and x_4) are irrelevant, then they will be simply removed from the data. As features are plotted along the x-axis (a.k.a. horizontal axis), and therefore, we name this process as horizontal contractions of the data.

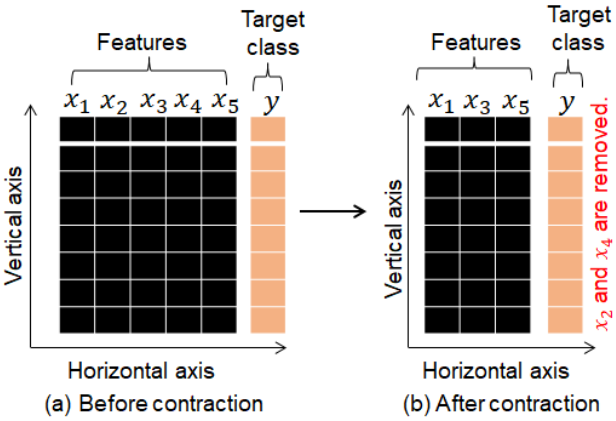


FIGURE 3. Workflow of the horizontal contraction process.

In most practical cases, there are constraints (e.g., value ranges, uniqueness) on the data, and ignoring such constraints might impact the sanctity of data characteristics to the problem at hand. However, the default setting of the shadowing and shuffling step of Boruta considers these constraints in the first step (e.g., extended feature matrix creation) by respecting the global information of each feature from real data [13]. For example, each real feature is randomly disordered only in shadow features without any modifications in the actual values, and therefore, the value ranges remain the same as real features. Similarly, the uniqueness of

[2]https://cran.r-project.org/web/packages/Boruta/index.html

values is sustained in the shadowing and shuffling step because the features' values remain unique column-wise only the index information changes.

Vertical contractions of the data using CGAN and intelligent fusion: In this step, we generate more data using a CGAN and intelligently fuse it in $\mathcal{D}$ by ensuring that only a minimal number of records get added. The implementation architecture of the CGAN is illustrated in Fig. 2 Part 3. To curate data of good quality, we adopted an open-source implementation of CGAN[3] model and modified it to fit our purpose (e.g., using pre-processed data, curating values only for salient features, developing condition vectors for salient features only, and applying modified losses to maintain stability during the training process). In the naive implementation of the CGAN, the data is fed directly to the model by downloading it from the URL/directory without pre-processing. However, we found that not pre-processing real data before feeding it to CGAN can induce various performance bottlenecks. In the adults' dataset, the country feature has many values of '?', and this is regarded as one unique value at value analysis time. Therefore, we used pre-processed data by fixing null values or wrong values to limit technical issues. In this work, the data is horizontally contracted, and therefore, synthetic data is generated for some salient features only whereas existing methods curate data for all features, leading to significant overheads while minimally improving the accuracy. Since features are reduced in the second step, the condition array encloses only relevant features' labels in the condition of CGAN in step 3. Similarly, due to optimization in other steps, the loss function is also adjusted accordingly to achieve faster convergence in CGAN. To the best of our knowledge, these optimization/changes are jointly applied for the first time to accomplish data augmentation tasks with the least overheads. After curating the synthetic data, we amalgamate them in $\mathcal{D}$ to produce complete data for training classifiers. Specifically, we add more records only in the minority class, c_m, whereas existing methods add samples to both major and minor classes, leading to higher computing overhead [6]. Our technique adds fewer records and only in c_m, leading to improved performance. In the next step (Fig. 2 Part 4), we apply noise removal to further improve data quality.

Noise removal: Owing to the new samples added in the former step, there is a possibility that some samples may lie within regions of c_M (the majority class). Hence, we apply a coin-throwing algorithm to remove noisy samples from $\mathcal{D}$ [14]. This algorithm jointly uses distance, probability, and k-means to identify and remove noisy samples. The key steps of the algorithm are: (i) apply a k-means algorithm to divide samples of c_M into different clusters; (ii) draw a sphere of radius r by employing a sample of majority class as the center (c); (iii) compute the distance from the center and minority samples located in c_M; (iv) compute the probability of the sample being noisy using Eq. 3, and set up a threshold to classify noisy and non-noisy samples in [0-1] vector encoding; and (v) remove noisy samples from $\mathcal{D}$.

$$P = 1 - \frac{d(v, c)}{r} \tag{3}$$

By using the above process, a refined $\mathcal{D}$ is curated that improves the learning dynamics of classifiers in terms of separability and learning.

In the k-mean application, it is vital to first determine the optimal # of k in order to divide samples into appropriate clusters. There are three commonly used approaches to find optimal k: silhouette co-efficient, gap statistics, and sum of squares (SS) inflection point method (a.k.a. elbow method). In this work, we implemented the elbow method to find the optimal # of k for each dataset. Figure 4 presents an example of finding k for a given dataset (stroke) using the elbow method. In Figure 4, we chose 5 as it falls on the inflection point.

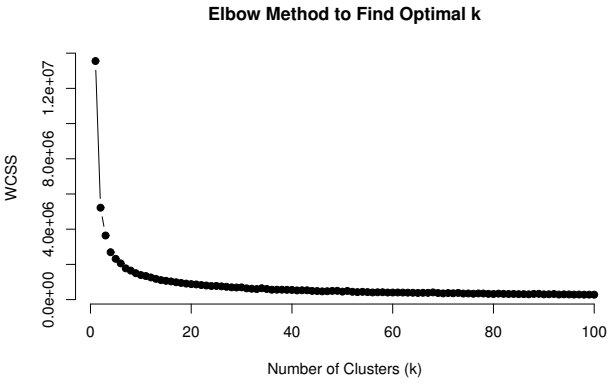


FIGURE 4. Example of finding optimal k via elbow method.

If the optimal value of k is not determined, then there is a higher risk of leftover records, and convergence time (# of iterations) can be prolonged. The value of r is determined by finding the distance between the cluster center and the furthest away sample of the majority class. The large value of r assists in reducing the computing burden and vice versa.

[3]https://learnopencv.com/conditional-gan-cgan-in-pytorch-and-tensorflow/

Building ML classifiers from a good-quality augmented $\mathcal{D}$: In the last step (Fig. 2 Part 5), classifiers are trained using the augmented $\mathcal{D}$ with significantly reduced parameters. In this work, we used random forest (RF)[4] as a classifier to measure the performance of our technique. RF is an ensemble method and has been widely used in many classification/regression tasks. It has two key parameters: *ntree* (the # of trees to be built), and *mtry* (the # of variables used in tree splits). A snapshot of the RF training and testing code is given below.

```
  \# Perform RF training.
  rf_classifier = randomForest(stroke~.,
data=training, ntree=150, mtry=6)
  \# Perform testing with unseen data.
  y_pred = predict(rf_classifier,
newdata = test)
```

The main rationale for employing the RF model is its ability to handle complex and multidimensional data. Furthermore, it yields higher accuracy than existing ensemble methods. Lastly, it has only a few parameters, which require less tuning.

Special Considerations: In HC, an important consideration is to prevent any appropriate feature from being deleted. In VC, ample attention is required to add sufficient and diverse records only under minority classes to address data imbalance problems. Lastly, in noise removal and ML classifier training, it is vital to determine the optimal values of all hyperparameters.

Experiments and Discusion

This section discusses the results that were obtained through experiments on three benchmark datasets: the Adult[5] dataset, the Stroke[6] Prediction dataset, and the Census-income[7] dataset. These datasets have been extensively used in testing the performance of ML models. In the Adult dataset, there are 32,561 samples and 15 mixed-type features (numerical and categorical). Salary/income is the target class in the Adult dataset. In the Stroke dataset, there are 5,510 samples and 12 mixed-type features. Stroke probability is the target class. In the Census-income dataset, there are 199,523 samples and various mixed-type features. We used fifteen relevant demographics as predictors and income as the target class. Next, we compare the results of our technique based on three criteria. Firstly, we computed and compared the performance of our technique in terms of accuracy (*Acc*). The value of *Acc* can be computed via Eq. 4:

$$Acc = \frac{T_p + T_n}{F_p + F_n + T_p + T_n} \tag{4}$$

where F_p and F_n are false positive and false negative, respectively. T_p and T_n are true positive and true negative, respectively.

Table 1 presents *Acc* results obtained from extensive experiments on the benchmark datasets. From the results, we can see that our technique has a higher *Acc* than both $\mathcal{D}$ and the augment everything (AET) approach [6] that is mostly followed in AI research [15]. By AET we mean that if there are two classes in $\mathcal{D}$, synthetic records will be added under each class and for all features, that cannot favorably solve the class imbalance problem in high dimensional data. In contrast, our work adds some synthetic records only in minor classes and under horizontally contracted data rather than all features. The addition of some synthetic records only under salient features makes our method highly customized without losing performance. A clear difference between the three augmentation methods (AET [6], vertical contraction [9], and horizontal and vertical (this paper)) is shown in Figure 5.

TABLE 1. Average *Acc*: ours versus $\mathcal{D}$ and AET approach.

Dataset	$\mathcal{D}$	AET	Ours
Stroke	94.63%	96.38%	99.63%
Adult	85.86%	90.01%	100%
Census	89.78%	93.81%	97.38%

Note: Our technique uses only 6, 10, and 11 features, whereas AET and $\mathcal{D}$ use 12, 15, and 15 features.

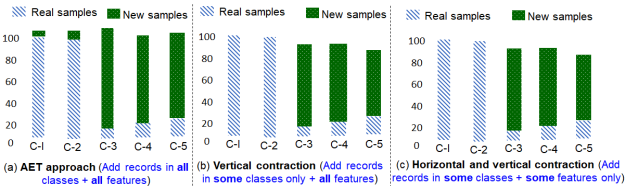


FIGURE 5. Difference between AET, vertical contraction, and vertical and horizontal contraction.

The AET is one of the latest and most practical data augmentation schemes that use synthetic samples curated with CTGAN for balancing datasets [6]. However, it does not explore ways to reduce augmented data sizes in both horizontal and vertical directions, which can lead to higher overheads in the case of large

[4]https://github.com/imbs-hl/ranger

[5]https://archive.ics.uci.edu/dataset/2/adult

[6]https://www.kaggle.com/code/ahmedashrafahmed/stroke-prediction-dataset

[7]http://archive.ics.uci.edu/dataset/117/census+income+kdd

datasets. Since our technique advances this scheme, therefore, we compared our results with this relevant baseline.

Secondly, we compared the model size (number of *ntree*) of our technique with the $\mathcal{D}$ and the AET approach. Table 2 lists the results obtained from using the same three datasets. The results show our technique outperformed the existing techniques on three datasets, and the model size is lower.

TABLE 2. Average model size: Ours versus $\mathcal{D}$ and AET.

Dataset	$\mathcal{D}$	AET	Ours
Stroke	395	501	150
Adult	595	700	350
Census	1009	1489	575

Note: Fewer trees $\rightarrow$ reduced model size.

Thirdly, we compared the time of our technique against $\mathcal{D}$ and the AET approach. The results in Table 3 were obtained from the same datasets, showing that our technique had lower time than $\mathcal{D}$ and AET.

TABLE 3. Average computing time: Ours versus $\mathcal{D}$ and AET.

Dataset	$\mathcal{D}$	AET	Ours
Stroke	4.24 s	5.83 s	0.89 s
Adult	62.04 s	78.4 s	7.18 s
Census	279.46 s	504.32s	129.72 s

To benchmark our technique, the results were compared with five SOTA augmentation techniques: random over-sampling (ROS), random under-sampling (RUS), the synthetic minority over-sampling technique (SMOTE), *k*-means clustering SMOTE (*K*-means SMOTE), fair SMOTE (Fair-SMOTE), and CTGAN sampling (CTGANSamp). We performed repeated experiments to compute and compare the results of our technique with the baseline approaches. Table 4 shows the *Acc* values obtained from the experiment on the three benchmark datasets. These results are an average of five rounds on each dataset.

TABLE 4. Avg. *Acc* (%): our technique versus SOTA schemes.

SOTA Method	Stroke	Adult	Census
RUS	89.12	76.35	83.77
ROS	95.54	75.41	87.54
SMOTE	94.61	70.32	88.91
K-means-SMOTE	94.03	74.43	90.09
Fair-SMOTE	95.23	85.09	91.87
CTGANSamp	96.03	90.03	93.89
Our Technique	99.63	100.00	97.38

From the results, we can see that our technique resulted in higher *Acc* (e.g., 99.63%, 100%, and 97.38%) compared to the SOTA schemes. Although some schemes yielded better performance, confusion matrices were highly imbalanced (e.g., most contributions in *Acc* came from majority classes). From the Stroke dataset, the contribution from the majority class using *k*-means SMOTE was 93.79%, which is 99.79% of the total *Acc*. In contrast, our scheme ensures a balanced contribution to *Acc* from both classes. From the Stroke dataset, the contribution to *Acc* from the majority class using our technique was 58.38%. The ML classifiers trained with data resulting from our technique can be more stable and robust. These results fortify the importance of our technique for deriving fair and informed decisions from ML classifiers in real-world cases.

Ablation study: In this work, we applied three different techniques: horizontal contraction (HC), vertical contraction (VC), and noise removal (NR). The results of ablation experiments that demonstrate the differential impact of each technique on the overall *Acc*, model size, and computing time results are in Table 5. From Table 5, it can be seen that VC has a greater impact on *Acc* because it increases the $\mathcal{D}$'s size. In contrast, HC sustains the *Acc* or slightly improves it as it decreases $\mathcal{D}$'s dimension. The NR has a distinct impact in each $\mathcal{D}$, owing to a variable degree of noise. However, the VC has higher overheads (avg. model size and computing time) than the HC and NR. These results confirm that our technique is efficient while yielding higher *Acc*.

TABLE 5. Differential impact of each technique on avg. *Acc*, avg. model size (# of trees), and avg. computing time.

Evaluation	Default	VC	HC	NR
Acc (S)	94.63%	+4.41%	-0.16%	+0.75%
Size (S)	395	+73	-282	-36
Time (S)	4.24s	+1.22s	-3.93s	-0.64s
Acc (A)	85.86%	+9.13%	+0.39%	+4.62%
Size (A)	595	+85	-271	-59
Time (A)	62.04s	+13.92s	-55.67s	-13.11s
Acc (C)	89.78%	+3.45%	+2.02%	+2.13%
Size (C)	1009	+451	-731	-154
Time (C)	279.46s	+188.86 s	-298.6s	-40.0s

Note: S, A, and C denote Stroke, Adult, and Census.

Impact of data characteristics on results: To further demonstrate the effectiveness of our technique, we conducted experiments by varying data characteristics in terms of data size, class imbalance, the quality of initial data with different degrees of noise and incompleteness, and data quality-related factors reported in the earlier parts of the paper. To assess the impact

of data size on *Acc*, computing time, and model size, we divided each dataset into five partitions (e.g., p1 ∼ p5). Figure 6 presents the results that were achieved by varying data sizes. We chose 20% of the data for the first time and varied it with the same factor for comparison and analysis. The Census dataset encloses the highest # of records, and therefore, the overheads are higher compared to other datasets.

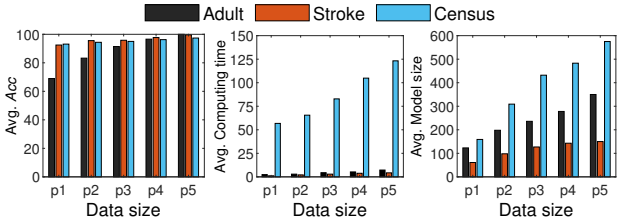
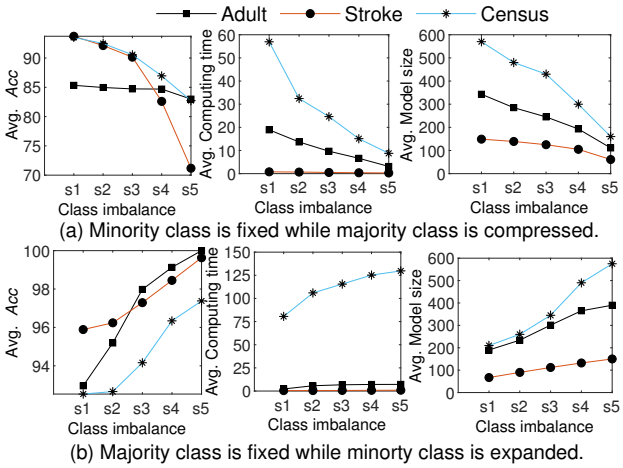


FIGURE 6. Impact of varying data size on results.

To assess the impact of class imbalance on results, we conducted experiments in two phases. In the first phase, the class imbalance was judged by decreasing samples from the majority class while keeping the minority class fixed. In the second phase, we kept the majority class fixed and expanded the minority class via synthetic samples. The experimental results of both phases are given in Figure 7. We conducted



(a) Minority class is fixed while majority class is compressed.

(b) Majority class is fixed while minorty class is expanded.

FIGURE 7. Impact of varying class imbalance on results.

experiments in five settings (s1 ∼ s5) for within-class imbalance analysis by varying imbalance by 20%. In the Stroke dataset, there is a big drop in *Acc* results because the minority class has only 249 samples. In other datasets, this issue did not occur as the majority/minority class has enough samples.

Lastly, we analyzed the impact of other factors such as the degrees of noise, incompleteness, and redundancy of the *D* on *Acc* results. Figure 8 presents the results of experiments that were obtained from three datasets and the impact of each factor. In Figure 8, A, S, and C denote the datasets (Adults, Stroke, and Census). From the results, it can be seen that noise has a higher impact while missing values have the least impact on *Acc* in these datasets. It is worth noting that the Census dataset has 21.50% missings values, but when data is refined, the number reduces to just 0.14 because two features with many 0 values were regarded as irrelevant. In the stroke dataset, there were no quality-related issues, and for analysis, we deliberately added 1.96% noise, 1.76% incompleteness, and 1.00% redundant tuples. On average, these characteristics have a 3.40% impact on the *Acc* results.

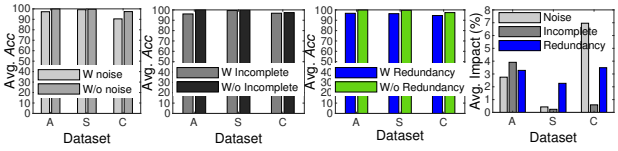


FIGURE 8. Impact of noise, incompleteness, and redundancy on *Acc* results, and Avg. impact of each factor.

The proposed technique has yielded better results on three benchmark datasets having a reasonable number of columns/features. However, if the number of columns in the data increases significantly, the computing overheads for horizontal contraction can be slightly higher due to more operations during feature selection. Also, the number of salient features can increase and synthetic data needs to be curated for those features. However, these overheads can be restrained by using advanced hardware architectures or by using more optimized implementations than the Boruta and CGAN algorithms. Lastly, the proposed technique is expected to maintain better performance in terms of *Acc*, classifier building time, and model size in reduced data, even when original data contain more columns, meeting the intended performance goals in real scenarios.

Conclusion and Future Work

In this paper, we implemented an efficient data augmentation technique by amalgamating the Boruta algorithm and a CGAN model to address imbalanced learning problems. By combining these models with noise removal, our technique adds fewer records with salient features only, whereas most of the existing techniques needlessly add more records than required. The salient features are selected in an automated manner that captures a closer relationship with the target

class. Good-quality data were curated by leveraging the CGAN model with a condition vector to achieve higher diversity in the synthetic data. This is the first approach that addresses the curse of dimensionality and that adds fewer records compared to previous schemes. Different and extensive experiments on benchmark datasets demonstrated the effectiveness and efficiency of the proposed technique. Specifically, our technique outperformed the existing techniques by enhancing accuracy, reducing computing time, reducing the model size, and providing greater balance in confusion matrices. For future work, we plan to investigate fairness issues while augmenting data and providing corresponding solutions in different AI environments. Lastly, we intend to investigate the impact of data characteristics (e.g., value ranges, uniqueness) in the shadowing and shuffling step of the Boruta algorithm to further improve data refinement results.

ACKNOWLEDGMENTS

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) (RS-2024-00340882). Seong Oun Hwang and Abdul Majeed are co-corresponding authors of this article.

REFERENCES

1. E. Strickland, "Andrew Ng, AI minimalist: The machine-learning pioneer says small is the new big," *IEEE Spectrum*, vol. 59, no. 4, pp. 22–50, 2022. [Online]. Available: 10.1109/MSPEC.2022.9754503
2. C. Hegde, "Anomaly detection in time series data using data-centric AI," in *2022 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*. IEEE, 2022, pp. 1–6. [Online]. Available: 10.1109/CONECCT55679.2022.9865824
3. E. Jeczmionek and P. A. Kowalski, "Input reduction of convolutional neural networks with global sensitivity analysis as a data-centric approach," *Neurocomputing*, vol. 506, pp. 196–205, 2022. [Online]. Available: https://doi.org/10.1016/j.neucom.2022.07.027
4. S. Eyuboglu, B. Karlaš, C. Ré, C. Zhang, and J. Zou, "dcbench: a benchmark for data-centric AI systems," in *Proceedings of the Sixth Workshop on Data Management for End-To-End Machine Learning*, 2022, pp. 1–4. [Online]. Available: https://doi.org/10.1145/3533028.3533310
5. Q. Dai, J.-w. Liu, and Y.-h. Shi, "Class-overlap undersampling based on Schur decomposition for class-imbalance problems," *Expert Systems with Applications*, vol. 221, p. 119735, 2023. [Online]. Available: https://doi.org/10.1016/j.eswa.2023.119735
6. A. S. Dina, A. Siddique, and D. Manivannan, "Effect of balancing data using synthetic data on the performance of machine learning classifiers for intrusion detection in computer networks," *IEEE Access*, vol. 10, pp. 96 731–96 747, 2022. [Online]. Available: 10.1109/ACCESS.2022.3205337
7. N. A. Azhar, M. S. M. Pozi, A. M. Din, and A. Jatowt, "An investigation of SMOTE based methods for imbalanced datasets with data complexity analysis," *IEEE Transactions on Knowledge and Data Engineering*, 2022. [Online]. Available: 10.1109/TKDE.2022.3179381
8. X. Zhang, L. Yu, H. Yin, and K. K. Lai, "Integrating data augmentation and hybrid feature selection for small sample credit risk assessment with high dimensionality," *Computers & Operations Research*, vol. 146, p. 105937, 2022. [Online]. Available: https://doi.org/10.1016/j.cor.2022.105937
9. A. Majeed and S. O. Hwang, "CTGAN-MOS: Conditional generative adversarial network based minority-class-augmented oversampling scheme for imbalanced problems," *IEEE Access*, 2023. [Online]. Available: 10.1109/ACCESS.2023.3303509
10. S. Maldonado, C. Vairetti, A. Fernandez, and F. Herrera, "FW-SMOTE: A feature-weighted oversampling approach for imbalanced classification," *Pattern Recognition*, vol. 124, p. 108511, 2022. [Online]. Available: https://doi.org/10.1016/j.patcog.2021.108511
11. G. Douzas, F. Bacao, and F. Last, "Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE," *Information Sciences*, vol. 465, pp. 1–20, 2018. [Online]. Available: https://doi.org/10.1016/j.ins.2018.06.056
12. J. Chakraborty, S. Majumder, and T. Menzies, "Bias in machine learning software: Why? how? what to do?" in *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2021, pp. 429–440. [Online]. Available: https://doi.org/10.1145/3468264.3468537
13. H. Zhou, Y. Xin, and S. Li, "A diabetes prediction model based on boruta feature selection and ensemble learning," *BMC bioinformatics*, vol. 24, no. 1, pp. 1–34, 2023. [Online]. Available: https://doi.org/10.1186/s12859-023-05300-5
14. H. Zhu, M. Zhou, G. Liu, Y. Xie, S. Liu, and C. Guo, "NUS: Noisy-sample-removed undersampling scheme for imbalanced classification and application to credit card fraud detection," *IEEE Transactions on Computational Social Systems*,

2023. [Online]. Available: 10.1109/TCSS.2023.3243925

15. A. S. Tarawneh, A. B. Hassanat, G. A. Altarawneh, and A. Almuhaimeed, "Stop oversampling for class imbalance learning: A review," *IEEE Access*, vol. 10, pp. 47 643–47 660, 2022. [Online]. Available: 10.1109/ACCESS.2022.3169512

Abdul Majeed is currently working as an Assistant Professor at the Department of Computer Engineering, Gachon University, Korea.

Noor Munir is currently working as an Assistant Professor at the Department of Smart Security, Gachon University, Korea.

Seong Oun Hwang is currently working as a Professor at the Department of Computer Engineering, Gachon University, Korea. He is a senior member of IEEE.