Enhancing Supervised Classifications with Metamorphic Relations

Liming Xu
International Doctoral Innovation Centre
University of Nottingham Ningbo China
Ningbo, Zhejiang, China
liming.xu@nottingham.edu.cn

Dave Towey*
AIOP Group, School of Computer Science
University of Nottingham Ningbo China
Ningbo, Zhejiang, China
dave.towey@nottingham.edu.cn

Andrew P. French
School of Computer Science
University of Nottingham
Nottingham, UK
andrew.p.french@nottingham.ac.uk

Steve Benford
School of Computer Science
University of Nottingham
Nottingham, UK
steve.benford@nottingham.ac.uk

Zhi Quan Zhou
School of Computing and Information Technology
University of Wollongong
Wollongong, NSW, Australia
zhiquan@uow.edu.au

Tsong Yueh Chen
Dept. of Computer Science and Software Engineering
Swinburne University of Technology
Hawthorn, VIC, Australia
tychen@swin.edu.au

ABSTRACT

We report on a novel use of metamorphic relations (MRs) in machine learning: instead of conducting metamorphic testing, we use MRs for the augmentation of the machine learning algorithms themselves. In particular, we report on how MRs can enable enhancements to an image classification problem of images containing hidden visual markers ("Artcodes").

Working on an original classifier, and using the characteristics of two different categories of images, two MRs, based on *separation* and *occlusion*, were used to improve the performance of the classifier. Our experimental results show that the MR-augmented classifier achieves better performance than the original classifier, algorithms, and extending the use of MRs beyond the context of software testing.

CCS CONCEPTS

• **Software and its engineering → Software testing and debugging**; • **Computing methodologies → Supervised learning by classification**; *Bagging*;

KEYWORDS

Metamorphic testing, metamorphic relations, supervised classification, Artcodes, random forests

ACM Reference Format:
Liming Xu, Dave Towey, Andrew P. French, Steve Benford, Zhi Quan Zhou, and Tsong Yueh Chen. 2018. Enhancing Supervised Classifications with Metamorphic Relations. In *MET'18: MET'18:IEEE/ACM International Workshop on Metamorphic Testing , May 27, 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, Article 4, 8 pages. https://doi.org/10.1145/3193977.3193978

*Corresponding author.

MET'18, May 27, 2018, Gothenburg, Sweden

ACM ISBN 978-1-4503-5729-6/18/05…$15.00
https://doi.org/10.1145/3193977.3193978

1 INTRODUCTION

Over the past two decades, machine learning techniques have been increasingly adopted by the research community to solve a range of practical problems. For researchers in the machine learning and software testing communities, building accurate learning models and verifying their quality are major topics. Due to the nature of machine learning programs, test oracles (mechanisms to determine if software behaviour is correct) are generally hard to define. Metamorphic testing (MT) has been used to alleviate the oracle problem in testing machine learning software [25, 31, 32]. Machine learning techniques have also been used to identify metamorphic relations (MRs) [17]. MT has been used to further analyse classification results of machine learning systems [4].

In the literature, MRs have been used in software verification and validation, and to assess the quality of software [34] — in this paper, we report on expanding the existing role of MRs to use as a kind of *post adjustor* to a machine learning program, to build a more accurate learning model. Compared to the reported use of MT in [4], in this paper, MR is used to adjust the both the inputs and outputs of a machine learning system. To the best of our knowledge, this is the first time MRs have been extended to such a use. Using an example of the Artcodes classification problem [33] — similar to QR codes [30], Artcodes are visual codes where bespoke designs can be scanned — we identify MRs for each category of inputs, and use them to augment the original classifier, improving its performance.

The rest of this paper is organised as follows. Section 2 gives a brief explanation of metamorphic relations. Section 3 describes the Artcode classification. Section 4 presents the details of the MR-augmented classifier. The experimental evaluation of the MR-augmented classifier's performance is given in Section 5. Finally, Section 6 concludes the paper.

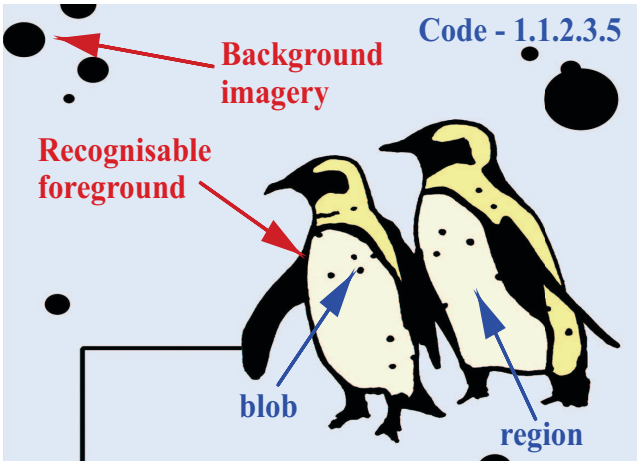


Figure 1: Illustration of the components of an Artcode.

2 METAMORPHIC RELATIONS

In software testing, an inability to determine if software is behaving correctly, or producing the correct output, is called the oracle problem [1]. Metamorphic testing (MT) is an approach that can alleviate the oracle problem [6, 10], MT has been investigated and adopted by a growing number of researchers and practitioners [7, 16, 20, 21, 28], successfully uncovering software problems, even in extensively tested systems [8, 9, 19]. Central to MT is a set of metamorphic relations (MRs), which are expected relations among the inputs and outputs of multiple executions of the intended program's functionality. Instead of examining the behaviour or output for an individual input, MT checks the SUT against selected MRs, with violations of an MR indicating the presence of a fault.

An example MR for a database management system is that the system should return the same results for a query with the search condition "A and B" and a query with the search condition "B and A".

3 ARTCODE CLASSIFICATION

Artcodes[1] (Figure 1) are human-designable topological visual markers, developed based on work in D-touch [11]. These computer-readable visual codes are embedded into images, allowing a designer to create codes that are machine-readable and meaningful to humans. They combine the visibility of the QR codes and the secrecy of these "invisible" markers [2, 23]. As an augmented reality artefact, Artcodes can adorn everyday objects with decorative patterns that enhance their beauty while triggering digital interactions when scanned — interested readers are referred to Benford et al. [3] for more details of Artcodes applications.

An Artcode includes two parts: a *recognisable foreground* and some *background imagery*, as shown in Figure 1. The recognisable part of an Artcode contains a closed boundary that is split into several regions (usually five), with each region containing one or more *blobs* – solid objects disconnected from the region edge, as shown in Figure 2. Additionally, background imagery can be added to the core part of an Artcode to enhance the aesthetics, but only if the background does not break the Artcode's topological structure. Moreover, Artcodes allow for *redundancy*, where multiple Artcodes with the same topology (but different geometry) can appear in an Artcode image. More information about Artcodes can be found, for example, in the work of Meese et al. [23].

As can be seen from the examples in Figures 2 and 3, visually, there is no obvious difference in geometrical shape or appearance between Artcodes and non-Artcodes. The geometrical variations associated with Artcodes are very different to, and more relaxed than, those of other well-known markers, such as QR codes [30], or ARTags [13]. Identification of the presence of Artcodes is not possible through visual inspection alone (as may be the case for QR codes). To trigger people's scan action and read the digital materials embedded in the Artcodes, it is necessary to detect their presence in the images or video sequence. This issue is referred to *Artcode classification* or *detection* [33].

Artcode classification is a binary classification problem, classifying an input image or video sequence as either containing an Artcode or not — labelled *Artcode* or *non-Artcode* classes, in this study. The Artcode class follows the topological definition of Artcodes, whereas the *non-Artcode* class comprises images that do not conform to these topological rules.

4 AUGMENTED CLASSIFIER

Typically, the first step with conventional classifiers involves creating feature vectors that *distinctively* describe each class. Machine learning models can then be used to predict the class of individual inputs. To date, to the best of our knowledge, no attempt has been made to make use of the metamorphic properties or MRs inherent in classification problems to enhance or rectify the classification outputs. Inspired by the various successes of MT, we examined the Artcode classification domain to identify MRs which we then used to augment and enhance the original classifier.

The MRs were identified through observation of the impact on classification results among different classes (or labels, e.g. Artcode or non-Artcode) when feeding in predefined inputs. In particular, the MRs allowed us to express probabilistically the likelihood of a modified input being Artcode or not based on the original classifier's classification (after performing the operations). This use of MRs in classification is different from that usually found in MT, which examines for MR violations to decide whether or not faults exist in the SUT; in contrast, our use of MRs *helps make a probabilistic classification decision* as to whether the input is an Artcode or not.

In the rest of this section, we describe the non MR-augmented classifier (the original classifier) that we used for Artcodes classification. We then explain how to augment this classifier with metamorphic relations identified from the classification model and input categories.

4.1 Original classifier

For Artcode classification, we built a classifier based on the shape of orientation histograms (SOH) [33] of input images and random forests [5]. The classifier makes use of SOH feature vectors, which describe the symmetry and smoothness of the orientation

[1] https://www.artcodes.co.uk/

Figure 2: Artcode examples from the Artcodes Dataset.

Figure 3: Non-Artcode examples from the Artcodes Dataset

(a) Separation masks **(b) Occlusion masks**

Figure 4: Separation and occlusion masks

histogram [15] of input images. Random forests were then trained using these feature vectors.

SOH is constructed from the orientation histogram, and was first proposed by Freeman et al. [15] for hand gesture recognition. The orientation histogram is computed using steerable filters [14], where orientations with weak magnitude (below the predefined threshold) are suppressed. Unlike previous feature sets used for describing the geometry or structure of fixed objects, SOH is used to describe the topological structure of images through analysis of the symmetry and smoothness of the orientation histogram. The SOH is constructed by quantifying these two aspects of the orientation histogram using similarity measurements such as procrustes [24] or Chi-squared (χ^2) distance [26].

After calculation of the SOH feature vector of each image, random forests were trained and used to predict the newly input image. As an ensemble learning method, a random forest has a number of attractive features. It is accurate, robust, and interpretable, and with little tuning required [5]. The effect of overfitting is seldom an issue, and it only requires a small amount of parameter tuning – the original classifier only tunes one parameter, the number of decision trees (nTrees) in the forests. Therefore, it is an appropriate method to be used in Artcode classification.

4.2 Metamorphic relations

As described in Section 3, Artcodes are composed of a number of connected regions. Each region is an independent entity that contains several solid blobs, and therefore has a complete topological structure. Additionally, an Artcode image might contain several independent Artcodes, which means that parts of Artcode images are likely to be complete regions or Artcodes themselves. In other words, parts of Artcodes are "*simplified*" Artcodes which will be classified as "*Artcodes*" by the original classifier with a relatively high probability. On the other hand, non-Artcode images (ideally) should not have those characteristics: parts of non-Artcode images do not have the predefined topology, and they will be treated as "*non-Artcodes*" by the original classifier. Therefore, parts of Artcode images are more likely to be classified as "*Artcodes*" than parts of non-Artcode images. Based on this observation, and the characteristics of the original classifier, we identified two MRs: *Separation* and *Occlusion*.

MR1-Separation. Separation involves splitting the input image uniformly into a number of sections, or *blocks*. For example, Figure 4(a) presents separation masks to generate four uniform blocks by intersecting them with input images. This MR is based on the observation that the blocks of Artcodes could be classified as "*Artcode*" with a higher probability than the blocks of non-Artcode images. If we select the number of blocks appropriately, this difference in the total probability of all blocks may provide more clues for classification. MR1-Separation can be formulated as:

$$\sum_{i=1}^{n} \Pr(B^i_{S_a}) \geq \sum_{i=1}^{n} \Pr(B^i_{S_n}) \tag{1}$$

where n is the number of image blocks; Pr() is the probability to be classified as Artcode by the original classifier; and $B^i_{S_a}$ and $B^i_{S_n}$ denote the ith block of the Artcode and non-Artcode image generated after MR-Separation, respectively.

MR2-Occlusion. Occlusion is similar to Separation, but the image blocks are not separated uniformly — blocks with overlapped areas are permitted. As shown in Figure 4(b), four occlusion masks are provided to intersect with the input image and output the image blocks outlined by white regions. Based on this, we have:

$$\sum_{i=1}^{m} \Pr(B^i_{O_a}) \geq \sum_{i=1}^{m} \Pr(B^i_{O_n}) \tag{2}$$

Occluded images generally keep half of the properties of the input images: half Artcode images have a high probability to be

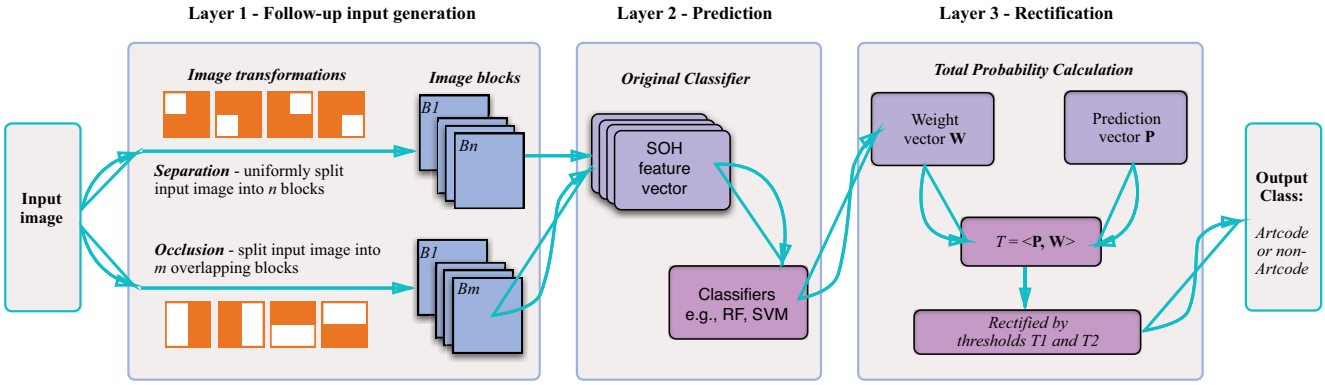


Figure 5: The MR-augmented classifier framework. This framework includes three stages: Follow-up input generation, Prediction, and Rectification. Follow-up input generation involves splitting the input image into blocks based on the two MRs; Prediction makes predictions of these image blocks using the original classifier; and Rectification involves the calculation of the total probability (p) of belonging to the Artcode, and then rectifies the predictions of the original classifier on the basis of the p value and the two predefined thresholds t_1 and t_2.

classified as "*Artcode*" by the original classifier, occluded images of non-Artcodes are still as likely to be labeled as "*non-Artcode*."

$$\Pr(B^i_{O_a}) \geq \Pr(B^i_{S_a}) \tag{3}$$

where m is the number of masks; $B^i_{O_a} = \cap(I_a, M_i)$ and $B^i_{O_n} = \cap(I_n, M_i)$ outputs the overlapped areas of Artcode and non-Artcode images I_a and I_n and the ith mask M_i; and $B^i_{O_a}$ and $B^i_{O_n}$ denote the ith block of the Artcode and non-Artcode images generated after MR-Occlusion, respectively.

We next explain how to use these relations to enhance the classification performance.

4.3 MR-augmented classifier

Unlike most deterministic software, classification is based on statistics, or is learned from prior experience. Given an input, the output of the classifier is a probabilistic classification of belonging to a class or not. In other words, after execution of the classifier, we only learn the probability of an input to be classified as a particular class or not. Therefore, to enable incorporation of the MRs described above, we designed an augmented classifier integrating the identified MRs, and adding an adjustor (or rectifier) to the original classifier. As shown in Figure 5, the augmented classifier first separates the input image into a number of blocks following the rules of MR1-Separation and MR2-Occlusion, and then predicts the label for each block using the original classifier, producing the *prediction vector*. As defined in Equations 1, 2 and 3, the probability of each class generated by separation and occlusion is different, and therefore we give different *weights* to them, thereby constructing a *weight vector*, which has the same dimensionality as the prediction vector. Given a prediction vector $\mathbf{v} = (c_1, ..., c_N)$ and weight vector $\mathbf{w} = (w_1, ..., w_N)$, where each c_i is the predicted class of the ith block according to the original classifier; w_i is the weight assigned to the ith block; and N is the dimensionality of both vectors (and is equal to the total amount of blocks in separation and occlusion), the inner product of $\mathbf{v}$ and $\mathbf{w}$ ($p = \mathbf{v} \cdot \mathbf{w} = \sum_{i=1}^{N} c_i \times w_i$) is the probability of belonging to the Artcode class (p value). The augmented classifier predicts the class of the input depending on the value of p and the given thresholds t_1 and t_2, using the following decision rules: if $p \geq t_2$, then it is Artcode; if $p < t_1$, then it is non-Artcode; otherwise, the input retains the original classifier's predicted class.

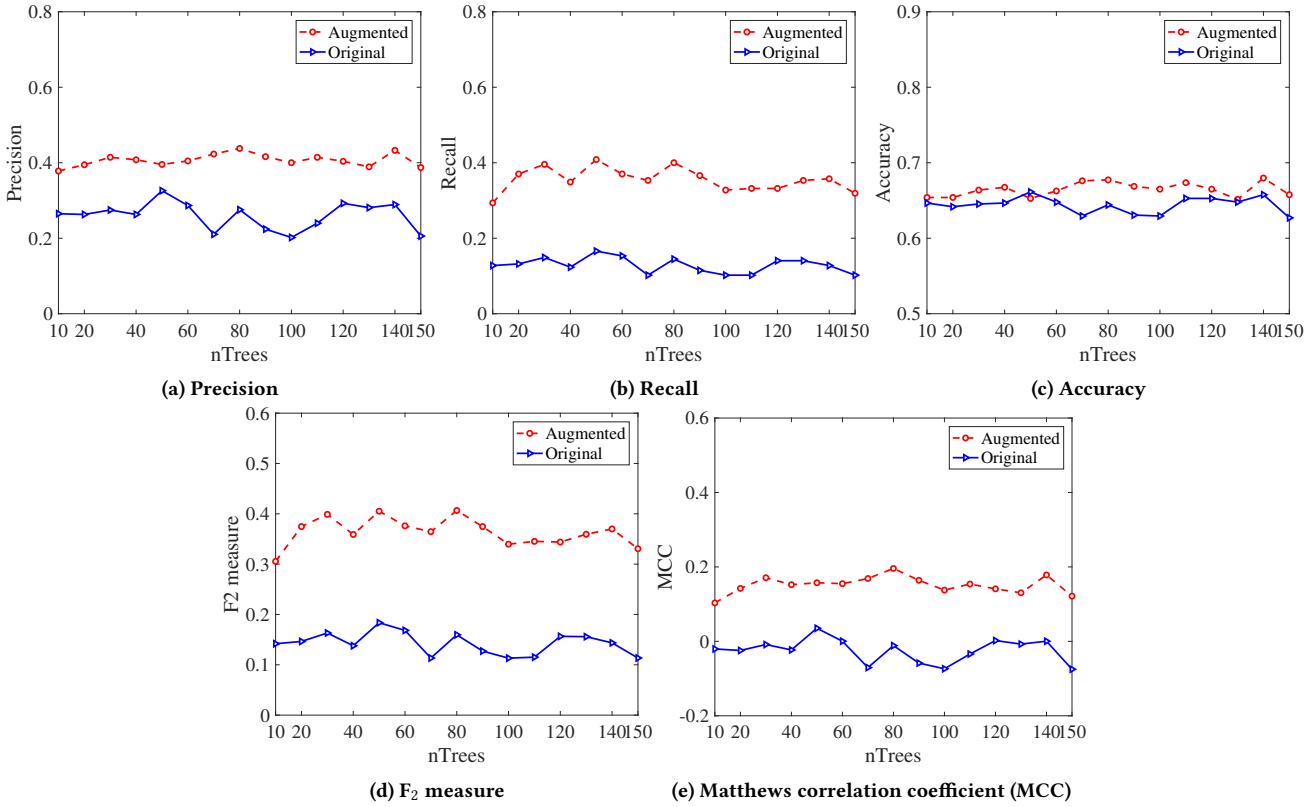
5 EXPERIMENTAL EVALUATION

5.1 Dataset

In order to study the Artcodes classification problem, we created a dataset containing 47 Artcode and 116 non-Artcode images. The non-Artcode images (comprising logos, drawings, and graphics) were all created by humans, and were deliberately selected such that they would appear very similar to actual Artcode images. This means that this dataset is very challenging for Artcodes classification. Because Artcodes are manually created by designers, the number of available Artcodes is currently small, but work is ongoing to extend the dataset.

5.2 Cross validation

Cross-validation is a commonly used model validation technique for assessing how a learning model will generalize to a dataset [12, 18]. One of the main reasons for using cross-validation rather than conventional validation (partitioning the dataset into two sets of 70% for training and 30% for testing) is that there is not sufficient data available to partition into separate training and test sets without losing significant modeling or testing capability. In these cases, a fair way to properly estimate model prediction performance is to use cross-validation [29]. We used k-fold cross-validation, which involves randomly partitioning a dataset into k equally-sized subsets, and keeping one single subset as the validation data for testing the trained model, and using the remaining $k-1$ subsets as training data. The cross-validation process is then repeated k times (the *folds*). Considering the limited number of samples in the Artcodes dataset,



(a) Precision (b) Recall (c) Accuracy

(d) F_2 measure (e) Matthews correlation coefficient (MCC)

Figure 6: Performance comparison between the original and MR-augmented classifier with different *nTrees* values and $t_1, t_2 = 0.2$.

a 5-fold cross-validation was used to ensure sufficient training and testing set sizes for the performance evaluation.

Table 1: Confusion matrix

		Predicted	
		Artcode	*non-Artcode*
Actual	*Artcode*	True Positive (TP)	False Negative (FN)
	non-Artcode	False Positive (FP)	True Negative (TN)

5.3 Experimental setting

We implemented an augmented classifier based on the framework shown in Figure 5 using Matlab, and evaluated its performance using cross-validation on the Artcodes dataset. As there is no existing research on Artcodes classification, we only compare the MR-augmented classifier with the original classifier presented in Section 4.1. Because random forests are used in the classifier, the performance exhibits a certain level of variation on each execution due to the random variable selection from the feature vector. Ten runs of cross-validation were therefore conducted to calculate the average performance.

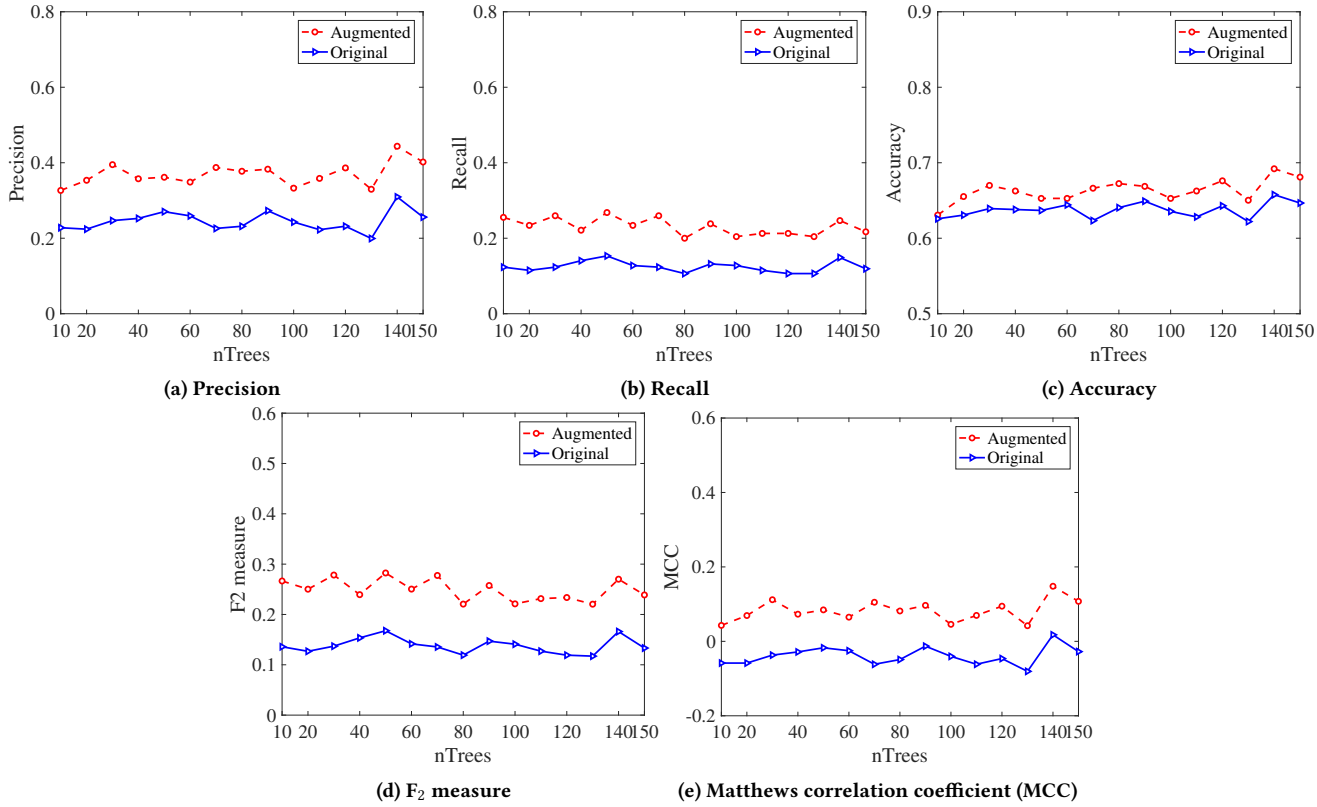
Considering the imbalance of the dataset (with more non-Artcode images) we selected five performance metrics to provide an informative view of the augmented classifier's performance: *Precision*, *Reall*, *Accuracy*, F_2 *measure*, and *MCC* (Matthews correlation coefficient) [22]. These five measures are calculated based on a confusion matrix (Table 1). A confusion matrix, also known as an error matrix, is a specific table layout that allows visualization of the performance of an algorithm — each row represents the instances in an actual class while each column represents the instances in a predicted class (or vice versa) [27].

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{6}$$

Accuracy (as defined in Equation 6) is the overall proportion of correct predictions, for both the Artcodes and non-Artcodes class, and is a simple way of describing a classifier's performance on a given dataset. However, Accuracy is sensitive to the dataset's imbalance. F_2 measure is a special case of the F_β measure:



(a) Precision **(b) Recall** **(c) Accuracy**

(d) F_2 measure **(e) Matthews correlation coefficient (MCC)**

Figure 7: Performance comparison between the original and MR-augmented classifier with different *nTrees* values and $t_1 = 0.15, t_2 = 0.3$.

$$F_\beta = \frac{(1+\beta^2)\cdot Precision \cdot Recall}{\beta^2 \cdot Precision + Recall} \tag{7}$$

where $\beta = 2$. As shown in Equations 4 and 5, Precision is the proportion of true positives among the all predicted positives, and Recall is the proportion of true positives over the total amount of actual positives. The F_2 measure uses a weighted average of Precision and Recall to evaluate the classification effectiveness, giving twice as much importance to recall as to precision.

Compared with Accuracy, the F_2 measure provides more insight into the performance of a classifier, but can be sensitive to data distributions. MCC (Equation 8) is in essence a correlation coefficient between the observed and predicted classifications, incorporating true and false positives and negatives.

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{8}$$

MCC is generally regarded as one of the best measures for classifier performance evaluation [27], and remains effective even if the dataset is imbalanced. The tuning parameters — the number of decision trees (nTrees) in the random forests, and the thresholds t_1 and t_2 — were studied in the experiment, as was their impact on the classifier.

The values of t_1 and t_2 are strongly related to the given values in the weight vector, and, according to Equation 3, the weights of blocks generated by Occlusion are greater than those for Separation. We separated the input images uniformly into four blocks, and overlapped with four occlusion masks, and thus had 8-dimensional prediction and weight vectors. Assuming we assign x to $\Pr(B^i_{O_a})$ and $\Pr(B^i_{O_n})$ and y to $\Pr(B^i_{S_a})$ and $\Pr(B^i_{S_n})$ $(x > y)$, as the total amount of weight vector is 1, then we have $x + y = 0.25, (x > y)$. Any combinations of values of x and y satisfying this condition can be used as the weights. For computational simplicity, in this experiment, we assigned both $\Pr(B^i_{S_a})$ and $\Pr(B^i_{S_n})$ a value of 0.1, and $\Pr(B^i_{O_a})$ and $\Pr(B^i_{O_n})$ both a value of 0.15, which gave the resulting weight vector:

$$\mathbf{w} = (0.1, 0.1, 0.1, 0.1, 0.15, 0.15, 0.15, 0.15)$$

To simplify calculations, we also used 1 and 0 in the prediction vector $\mathbf{p}$ to represent the "Artcode" and "non-Artcode" classes, respectively.

5.4 Experimental results

All performance metric values reported are the average values calculated from ten executions of k-fold cross-validation [18]. As explained in Section 5.2, because of the limited number of samples

in the Artcodes dataset, a 5-fold cross-validation was used to ensure sufficient training and testing set sizes for the performance evaluation. Two combinations of the thresholds t_1 and t_2 combined with different numbers of nTrees were used to study the classifier's tuning parameters. In all graphs in Figures 6 and 7, higher values indicate better performance.

The impact of nTrees on the augmented classifier's performance is illustrated in Figures 6 and 7, which show a stable performance across different numbers of nTrees in terms of the five evaluation metrics. This means that the augmented classifier is not sensitive to changes in the number of nTrees, a property it inherits from the original classifier.

For various numbers of nTrees and fixed thresholds t_1 and t_2, the augmented classifier outperforms the original classifier in terms of all five metrics. The augmented classifier performs better in terms of both precision (Figures 6(a) and 7(a)) and recall (Figures 6(b) and 7(b)) than the original classifier, with an average of about 10-15% improvement for both threshold combinations. As shown in Figures 6(c) and 7(c), for both threshold combinations, the augmented classifier has slightly better Accuracy than the original classifier, about 2-3% improvement on average. Although the MR-augmented classifier shows improved performance with the Artcodes class, the small percentage of artcodes in the dataset does not contribute strongly to the overall evaluation of Accuracy, which is influenced by both true positives and true negatives.

In contrast, F_2 measure and MCC are more informative measures of overall performance, even when the dataset is imbalanced. As shown in Figures 6(d)(e) and 7(d)(e), the augmented classifier obtains better values (approximately 15-20% improvement) than the original for different numbers of nTrees, showing an overall improved performance of the MR-augmented classifier. However, due to the imbalance of the dataset, the MCC values for the two classifiers are relatively low.

Overall, the MR-augmented classifier achieves improved performance according to the five evaluation metrics. This improved performance is sensitive to the threshold values for t_1 and t_2, but not to nTrees. The impact of nTrees on the classifier's performance is relatively small, but larger numbers of nTrees require more time to train the classifier and make the classification predictions. Thus, careful selection of the tuning parameter values is necessary to ensure the performance improvement of the original classifier.

6 CONCLUSION

In this paper, we have reported on a study using metamorphic relations (MRs) to improve binary classification in machine learning. To the best of our knowledge, this is the first use of MRs in such an application. Two MRs were identified based on properties of the input data and the usage of the classification model, and an augmented classifier using these two MRs was designed to show the applicability of the technique. Experimental evaluation showed the performance improvement across certain aspects of the original classifier, demonstrating the potential to apply MT theories and techniques to machine learning applications. The experimental evaluation also showed the importance of the tuning parameters t_1 and t_2 on the performance of the augmented classifier. Our future work will include further examination of other parameters, including the number of blocks in the separation and occlusion MRs, the given values of the weight vector, and the adaptive values of thresholds t_1 and t_2.

Although this has been a preliminary study, the results are very promising, and clearly demonstrate the potential for MR-augmentation of classifiers. More practical and theoretical work will be necessary to fully investigate this new research direction, including more case studies examining application of MRs to other well-known machine learning problems, such as face and object detection.

ACKNOWLEDGMENTS

The authors acknowledge the financial support from the International Doctoral Innovation Centre, Ningbo Education Bureau, Ningbo Science and Technology Bureau, and the University of Nottingham. This work was supported by grants from the UK Engineering and Physical Science Research Council (Grant No. EP/L015463/1), the National Natural Science Foundation of China (Grant No. 71471092) and the Ningbo Science and Technology Bureau (Grant No. 2014A35006).

REFERENCES

[1] Earl T. Barr, Mark Harman, Phil McMminn, Muzammil Shahbaz, and Shin Yoo. 2015. The Oracle Problem in Software Testing: A Survey. *IEEE Transactions on Software Engineering* 41, 5 (2015), 507–525.
[2] Steve Benford, Adrian Hazzard, and Liming Xu. 2015. The Carolan guitar: a thing that tells its own life story. *Interactions* 22, 3 (April 2015), 64–66.
[3] Steve Benford, Boriana Koleva, Anthony Quinn, Emily-Clare Thorn, Kevin Glover, William Preston, Adrian Hazzard, Stefan Rennick-Egglestone, Chris Greenhalgh, and Richard Mortier. 2017. Crafting Interactive Decoration. *ACM Transactions on Computer-Human Interaction (TOCHI) Comput.-Hum. Interact.* 24, 4, Article 26 (August 2017), 39 pages.
[4] Wing Kwong Chan, Jeffrey C. F. Ho, and T. H. Tse. 2010. Finding failures from passed test cases: Improving the pattern classification approach to the testing of mesh simplification programs. *Software Testing, Verification and Reliability* 20, 2 (2010), 89–120.
[5] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.
[6] Tsong Yueh Chen, Shing Chi Cheung, and Shiu Ming Yiu. 1998. Metamorphic testing: a new approach for generating next test cases. *Technical Report HKUST-CS98-01, Dept. of Computer Science, Hong Kong Univ. of Science and Technology* (1998).
[7] Tsong Yueh Chen, Fei-Ching Kuo, Huai Liu, Pak-Lok Poon, Dave Towey, T. H. Tse, and Zhi Quan Zhou. 2018. Metamorphic testing: A review of challenge and opportunities. 51, 1 (2018), 4:1-4:27. ACM Computing Surveys.
[8] Tsong Yueh Chen, Fei-Ching Kuo, Wenjuan Ma, Willy Susilo, Dave Towey, Jeffrey Voas, and Zhi Quan Zhou. 2016. Metamorphic testing for cybersecurity. *Computer* 49, 6 (2016), 48–55.
[9] Tsong Yueh Chen, Fei-Ching Kuo, Dave Towey, and Zhi Quan Zhou. 2015. A revisit of three studies related to random testing. *SCIENCE CHINA Information Sciences* 58, 5, Article 52104 (2015), 052104:1–052104:9 pages.
[10] Tsong Yueh Chen, T. H. Tse, and Zhi Quan Zhou. 2003. Fault-based testing without the need of oracles. *Information and Software Technology* 45, 1 (2003), 1–9.
[11] Enrico Costanza and Jeffrey Huang. 2009. Designable visual markers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1879–1888.
[12] Pierre A. Devijver and Josef Kittler. 1982. *Pattern recognition: A statistical approach*. Prentice Hall, London, UK.
[13] Mark Fiala. 2005. ARTag, a fiducial marker system using digital techniques. In *Proceedings of 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 2. IEEE, San Diego, CA, USA, 590–596.
[14] William T. Freeman and Edward H. Adelson. 1991. The design and use of steerable filters. *IEEE Transactions on Pattern analysis and machine intelligence* 13, 9 (1991), 891–906.
[15] William T. Freeman and Michal Roth. 1995. Orientation histograms for hand gesture recognition. In *International workshop on automatic face and gesture recognition*, Vol. 12. 296–301.
[16] Darryl C. Jarman, Zhi Quan Zhou, and Tsong Yueh Chen. 2017. Metamorphic Testing for Adobe Data Analytics Software. In *Proceedings of the 2nd International*

Workshop on Metamorphic Testing (MET '17, in conjunction with ICSE 2017). IEEE Press, Piscataway, NJ, USA, 21–27.

[17] Upulee Kanewala and James M Bieman. 2013. Using machine learning techniques to detect metamorphic relations for programs without test oracles. In *Proceedings of the 24th International Symposium on Software Reliability Engineering (ISSRE'13)*. IEEE, Pasadena, CA, USA, 1–10.

[18] Ron Kohavi. 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th international joint conference on Artificial intelligence (IJCAI'95)*, Vol. 2. Morgan Kaufmann Publishers Inc., Montreal, Quebec, Canada, 1137–1143.

[19] Vu Le, Mehrdad Afshari, and Zhendong Su. 2014. Compiler Validation via Equivalence Modulo Inputs. In *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI '14)*. ACM, New York, NY, USA, 216–226.

[20] Mikael Lindvall, Dharmalingam Ganesan, Ragnar Árdal, and Robert E. Wiegand. 2015. Metamorphic Model-based Testing Applied on NASA DAT: An Experience Report. In *Proceedings of the 37th International Conference on Software Engineering (ICSE'15)*, Vol. 2. IEEE, Florence, Italy, 129–138.

[21] Huai Liu, Fei-Ching Kuo, Dave Towey, and Tsong Yueh Chen. 2014. How Effectively does Metamorphic Testing Alleviate the Oracle Problem? *IEEE Transactions on Software Engineering* 40, 1 (Jan 2014), 4–22.

[22] Brian Matthews. 1975. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure* 405, 2 (1975), 442–451.

[23] Rupert Meese, Shakir Ali, Emily-Clare Thorne, Steve Benford, Anthony Quinn, Richard Mortier, Boriana Koleva, Tony Pridmore, and Sharon L. Baurley. 2013. From codes to patterns: designing interactive decoration for tableware. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'13)*. ACM, Paris, France, 931–940.

[24] Jürgen Moser. 1965. On the volume elements on a manifold. *Transactions of the American Mathmatical. Society.* 120, 2 (1965), 286–294.

[25] Christian Murphy, Gail Kaiser, Lifeng Hu, and Leon Wu. 2008. Properties of Machine Learning Applications for Use in Metamorphic Testing. In *Proceedings of the 20th International Conference on Software Engineering and Knowledge Engineering (SEKE'08)*. Redwood City, CA, USA, 867–872.

[26] Karl Pearson. 1900. X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 50, 302 (1900), 157–175.

[27] David M. W. Powers. 2011. Evaluation: From precision, recall and f-measure to roc., informedness, markedness & correlation. *Journal of Machine Learning Technologies* 2, 1 (2011), 37–63.

[28] Sergio Segura, Gordon Fraser, Ana Sanchez, and Antonio Ruiz-Cortés. 2016. A Survey on Metamorphic Testing. *IEEE Transactions on Software Engineering* 42 (2016), 805–824.

[29] Giovanni Seni and John Elder. 2010. Ensemble methods in data mining: improving accuracy through combining predictions. *Synthesis Lectures on Data Mining and Knowledge Discovery* 2, 1 (2010), 1–126.

[30] D Wave. 2015. Information technology automatic identification and data capture techniques QR code bar code symbology specification. *International Organization for Standardization, ISO/IEC* 18004 (2015).

[31] Xiaoyuan Xie, Joshua Ho, Christian Murphy, Gail Kaiser, Baowen Xu, and Tsong Yueh Chen. 2009. Application of metamorphic testing to supervised classifiers. In *Proceedings of the 9th International Conference on Quality Software (QSIC'09)*. IEEE, Jeju, Korea, 135–144.

[32] Xiaoyuan Xie, Joshua WK Ho, Christian Murphy, Gail Kaiser, Baowen Xu, and Tsong Yueh Chen. 2011. Testing and validating machine learning classifiers by metamorphic testing. *Journal of Systems and Software* 84, 4 (2011), 544–558.

[33] Liming Xu, Andrew P. French, Dave Towey, and Steve Benford. 2017. Recognizing the Presence of Hidden Visual Markers in Digital Images. In *Proceedings of the on Thematic Workshops of ACM Multimedia 2017 (Thematic Workshops '17)*. ACM, Mountain View, California, USA, 210–218.

[34] Zhi Quan Zhou, Shaowen Xiang, and Tsong Yueh Chen. 2016. Metamorphic testing for software quality assessment: A study of search engines. *IEEE Transactions on Software Engineering* 42, 3 (2016), 264–284.