Information and Software Technology xxx (2011) xxx–xxx

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Automating image segmentation verification and validation by learning test oracles

Kambiz Frounchi [a,*], Lionel C. Briand [b], Leo Grady [c], Yvan Labiche [a], Rajesh Subramanyan [d]

[a] Dept. of Systems and Computer Engineering, Carleton University, Ottawa, Canada
[b] Simula Research Laboratory and The University of Oslo, Oslo, Norway
[c] Dept. of Imaging Analytics and Informatics, Siemens Corporate Research, Princeton, USA
[d] Dept. of Software Engineering, Siemens Corporate Research, Princeton, USA

ARTICLE INFO

Article history:
Received 11 November 2010
Received in revised form 7 June 2011
Accepted 24 June 2011
Available online xxxx

Keywords:
Verification and validation
Machine learning
Image segmentation
Image processing

ABSTRACT

An image segmentation algorithm delineates (an) object(s) of interest in an image. Its output is referred to as a segmentation. Developing these algorithms is a manual, iterative process involving repetitive verification and validation tasks. This process is time-consuming and depends on the availability of experts, who may be a scarce resource (e.g., medical experts). We propose a framework referred to as Image Segmentation Automated Oracle (ISAO) that uses machine learning to construct an oracle, which can then be used to automatically verify the correctness of image segmentations, thus saving substantial resources and making the image segmentation verification and validation task significantly more efficient. The framework also gives informative feedback to the developer as the segmentation algorithm evolves and provides a systematic means of testing different parametric configurations of the algorithm. During the initial learning phase, segmentations from the first few (optimally two) versions of the segmentation algorithm are manually verified by experts. The similarity of successive segmentations of the same images is also measured in various ways. This information is then fed to a machine learning algorithm to construct a classifier that distinguishes between consistent and inconsistent segmentation pairs (as determined by an expert) based on the values of the similarity measures associated with each segmentation pair. Once the accuracy of the classifier is deemed satisfactory to support a consistency determination, the classifier is then used to determine whether the segmentations that are produced by subsequent versions of the algorithm under test, are (in)consistent with already verified segmentations from previous versions. This information is then used to automatically draw conclusions about the correctness of the segmentations. We have successfully applied this approach to 3D segmentations of the cardiac left ventricle obtained from CT scans and have obtained promising results (accuracies of 95%). Even though more experiments are needed to quantify the effectiveness of the approach in real-world applications, ISAO shows promise in increasing the quality and testing efficiency of image segmentation algorithms.

1. Introduction

Image segmentation is the act of grouping and localizing 2D or 3D image content and is widely used in the many applications involving image processing [32]. Identifying groups of pixels as belonging to the same object can have many practical uses, such as object measurement or object recognition. Examples range from using segmentation to help the blind by identifying surrounding objects and converting this information into coded sound [32], to Document Image Analysis (DIA), where segmentation is used to interpret the different components of a document (text, drawings, maps, formulas, etc.) for document classification or other applications [2]. Although image segmentation algorithms may be used to localize multiple objects, our focus in this study will be on the segmentation of a single object of interest within an image. A common representation of the image segmentation is a labeling of each pixel in the image as object or background. Image segmentation algorithms are typically designed to automatically segment an image without the need for an expert to manually delineate the objects of interest in the image (an example is shown in Fig. 1).

In many areas of medical imaging, automatic image segmentation of a target object is a critical task to provide measurements of an object that may be used for diagnosis or treatment planning [22,25]. Therefore, we adopt medical image segmentation as our focus area for this study. However, before a segmentation algorithm can be used clinically (or commercially), it is imperative to know how well the segmentation algorithm performs on a wide range of real image data, comprising possibly hundreds or thousands of test cases. If a ground truth segmentation is available

* Corresponding author.
E-mail addresses: kambiz@sce.carleton.ca (K. Frounchi), briand@simula.no (L.C. Briand), leo.grady@siemens.com (L. Grady), labiche@sce.carleton.ca (Y. Labiche), rajesh.subramanyan@siemens.com (R. Subramanyan).

0950-5849/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.infsof.2011.06.009

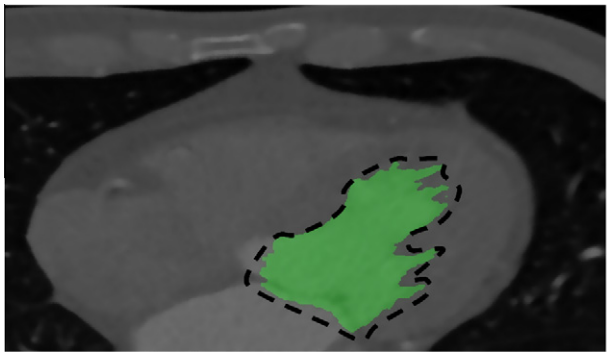

Fig. 1. Cardiac left ventricle segmentation depicted by the green labeling (we added an outline of the segmentation to the picture—dashed line—for black and white printouts).

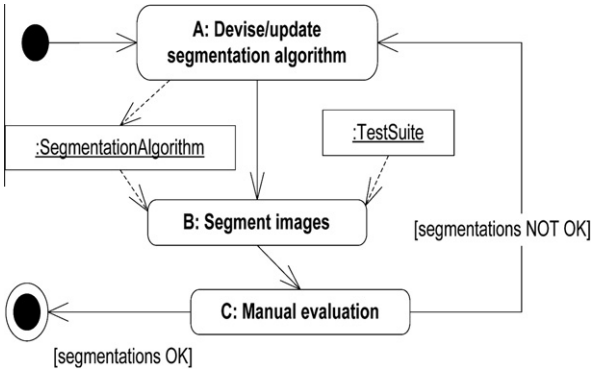

Fig. 2. Manual image segmentation algorithm evaluation process.

for each of these test cases (e.g., manual segmentations from a clinician) then it is possible to use these segmentations as a basis to evaluate the segmentation performance of the automatic algorithm. However, for any clinical task there is typically a range of segmentations which are considered to be correct (e.g., two clinicians often provide slightly different manual segmentations) but, at the same time, even seemingly small deviations from the ground truth segmentations can have clinical importance. Consequently, a visual inspection is often necessary to assess whether differences between segmentations are medically significant. More often, the costs of acquiring ground truth segmentations for a large test set can be too high to be practical, since a well-paid clinician is unlikely to spend the enormous amounts of time required to manually segment each case. Interactions with domain experts indicate that a more reasonable expectation is that a clinician might be willing to quickly examine the segmentation results generated by an automatic algorithm and score the quality of the produced segmentation. The term *score* here describes the evaluation of quality for a segmentation, for example Correct or Incorrect.

The usual way of verifying and validating a medical image segmentation algorithm begins by devising and applying a first version of the segmentation algorithm (activities A and B in Fig. 2) to a set of images. From our standpoint, the images constitute the test suite, and each image is representing a test case. The results obtained from the segmentation algorithm are then manually evaluated by medical experts (activity C). If a predetermined[1] number of segmentations are correct the algorithm is deemed adequate for clinical or commercial use. After deployment of the software into a clinical environment, the image segmentations are typically reviewed visually by a clinician during use of the software. Therefore, the primary consideration in determining correctness of the algorithm is whether the segmentation algorithm will consistently provide benefit to the clinical workflow by saving the clinician time. The algorithm saves the clinician time if making manual object measurements is slower than waiting for the algorithm to run and then possibly editing the results. If a segmentation algorithm is not ready for deployment, the algorithm must be modified. When a modification is required, the revised algorithm is re-applied to the image set and the same evaluation procedure is repeated until a satisfactory version of the segmentation algorithm is reached (Fig. 2). Practice shows that the number of iterations can be large, in many cases requiring dozens of iterations. Indeed, medical experts are required to visually inspect the correctness of the segmentations for each iteration and this often results in long waiting times to assess the consequence of any change to the image segmentation algorithm code (or algorithm parameters), delaying the software development cycle significantly and thus increasing software development costs. Manual evaluation of such large numbers of segmentations is also error-prone.

We treat this evaluation problem for image segmentation systems as an instance of the *oracle problem* in software testing, which is the problem of finding a procedure to assess the correctness of test results (in our case image segmentations) [19]. Our proposed solution leads to the partial automation of segmentation oracles, thus making verification and validation more time-efficient and less reliant on medical experts. We use machine learning to build a classifier that determines the consistency of segmentation pairs (segmentations obtained from different versions of the segmentation algorithm but extracted from the same patient image) and then use this information to predict the correctness of segmentations. Two segmentations are considered consistent if they lead to the same score, otherwise they are considered inconsistent.

To teach the machine learning algorithm to distinguish between consistent and inconsistent segmentation pairs, the (dis)similarity between different segmentation pairs are quantified using several measures and their consistency is determined from expert evaluations of the first few versions of the segmentation algorithm. Ideally (in terms of time spent to find an accurate oracle), as in our case study, two versions of the algorithm would be sufficient to find an accurate classifier. We refer to our solution for image segmentation evaluation as Image Segmentation Automated Oracle (ISAO). Though our case study focuses on a specific segmentation application, the methodology of ISAO is re-usable in other image segmentation verification and validation contexts.

An automated oracle for testing image segmentation algorithms helps designers:

(1) Reduce the long wait times for getting responses back from clinicians for every version of the segmentation algorithm.
(2) Understand quickly how the changes made in a new version of the segmentation algorithm have generally impacted the accuracy of the segmentation algorithm.
(3) Efficiently find the specific test cases, where the new version of the segmentation algorithm has failed to correctly segment the image rather than visually inspecting numerous test cases.
(4) Systematically test the performance of different parameter configurations of the image segmentation algorithm.

An important side benefit of finding the automated oracle is that the resulting classifier provides insight into which combination of segmentation measures leads to a better discriminator of

[1] Agreed between the image segmentation algorithm designer and medical experts.

Please cite this article in press as: K. Frounchi et al., Automating image segmentation verification and validation by learning test oracles, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.06.009

Table 1
Similarity measures.

Measure	Type	Description
AVD	VD	Absolute value of volume difference
ANVD	VD	Absolute value of normalized volume difference
DSC	O	Dice similarity coefficient
TC	O	tanimoto coefficient
TPVF	O	True positive volume fraction
FPVF	O	False positive volume fraction
ADBD	G	Average distance to boundary difference
HD	G	Hausdorff distance
BD	G	Baddeley distance
PMME	G	Peli Malah mean error
PMMSE	G	Peli Malah mean squared error
PFOM	G	Pratt's figure of merit
SODI	G	Scalable ODI (Odet's ODI_n)
SUDI	G	Scalable UDI (Odet's UDI_n)
ODI	G	Odet's ODI
UDI	G	Odet's UDI
PAD	G	Principal axis difference
RMMSD	G	Root mean square surface distance

the similarity between two segmentations. This insight is significant in image processing research since there is no specific way of knowing which measures allow us to compare two segmentations in a clinically relevant manner for a particular segmentation task, or how deviant a segmentation can be from a ground truth segmentation before it is no longer considered correct. In this regard, Deng et al. [11] conducted a study in which they tried to correlate numerical methods of comparing ground truth with algorithm segmentation to subjective clinical scoring of a segmentation relative to ground truth. We go further than Deng et al. by learning to predict score *changes* from segmentation changes, even in the absence of ground truth data.

The rest of this paper is organized as follows: Section 2 gives the reader some background about the adopted image segmentation comparison measures and a brief description of the machine learning concepts used in this research work. We detail our test oracle approach in Section 3. We report its results from the cardiac left ventricle segmentation verification and validation study in Section 4, where we describe several classifiers and compare their performance. A discussion on related work is given in Section 5 and conclusions are drawn in Section 6.

2. Background

In this section some background is given on two main subjects that are essential to building the automated oracle. Section 2.1 will give the reader some background about the similarity measures which are used for training the classifiers. Section 2.2 describes the machine learning algorithms that use the similarity measures in Section 2.1 to build the classifiers. The techniques used to evaluate the performance of the classifiers in order to obtain a valid classifier for use in the oracle (as explained in Section 3) are also described in Section 2.2.

2.1. Similarity measures

As explained in Section 1, we rely on several measures to quantify the similarity between segmentations with respect to different criteria. Apart from a few measures that we have defined ourselves, the rest of the measures are obtained from the image processing literature and adapted to our needs. We divide the measures into three types: volume difference, overlap, and geometrical measures. An overview of the different categories is given here. For a detailed description of the measures please refer to [12]. Table 1 summarizes the measure names, types (VD = Volume Difference, O = Overlap, G = Geometrical), and the acronym descriptions.

Volume difference measures calculate either in absolute or relative terms the difference in the number of voxels[2] labeled in the two segmentations multiplied by the volume associated with each voxel in the image. These measures are relevant because a common purpose for medical image segmentation is the measurement of volume. In the application of our case study, the most important output of the segmentation system was the left ventricle volume, which is used for diagnosis purposes.

Overlap measures [9,28] calculate some measure of overlap between the two segmentations. The intersecting and non-intersecting regions of the two segmentations are identified and different fractions are defined, each measure placing more emphasis on the extent of agreement of some regions of interest.

Geometrical measures [1,5,11,15,23,24,27] compare the segmentations in terms of their shape differences capturing variations such as the *distance* between the boundary voxels of the two segmentations. These measures may help in finding such cases, where for example a segmentation has a high percentage of overlap with the correct segmentation but incorrectly labels some voxels that make the segmentation incorrect in the view of a medical expert.

The similarity measures in Table 1 are not dependant on the application domain and only compare segmentations using general properties of a segmentation, thus they can be potentially used in any image analysis application.

2.2. Machine learning

The goal of machine learning is to teach a computer system to *learn*. A learning system can determine various relationships between different variables representing a concept (association algorithms), or learning how to divide a piece of data into different *natural* groups (clustering). The system may also learn how to assign a *class* to examples of a domain (classification algorithms). In this paper we are interested in classification algorithms to learn whether two segmentations can be classified as consistent or inconsistent based on the similarity measurements between segmentations. The input of classification algorithms is a set of instances that are each characterized by the values of a number of *attributes* and assigned a *class* (a distinct category). In machine learning terminology the input is referred to as *training instances*. The algorithm constructs a *classifier* that shows in some format (for examples rules) what ranges of the attributes and relationships between them lead to each class. The constructed classifier can predict the class of unknown instances for which we do not know their class [30]. Better classifiers are constructed when more training instances with similar proportions of instances from different classes are available. In our case, an instance is a pair of two segmentations for the same patient but generated by different segmentation algorithm versions, attributes are similarity measures, and classes describe whether a medical expert evaluated the two segmentations to be consistent.

We have used the WEKA-implemented machine learning algorithms J48, JRIP and PART [30] in our case study. The Waikato Environment for Knowledge Analysis (WEKA) is a tool widely used by machine learning researchers. It provides readily available implementations for popular machine learning algorithms and an environment for conducting standard tests to determine how well the machine learning algorithms perform with respect to different training and test sets. J48 implements the very well-established C4.5 algorithm that is a standard algorithm to create decision trees. C4.5 uses a divide and conquer approach, choosing different

[2] A pixel in a 3D image is referred to as a voxel.

Please cite this article in press as: K. Frounchi et al., Automating image segmentation verification and validation by learning test oracles, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.06.009

attributes in the order of least entropy[3] to divide the instances into different branches, growing the tree recursively and stopping the growth of a branch when the class of the instances in that branch can be determined or, in other words, a leaf node has been reached. PART uses partial decision trees to construct rules from the branches that lead to a leaf node covering the most instances. This algorithm attempts to avoid building a full decision tree for each rule by growing the resultant attribute split subsets with the lower entropies first, which leads to small sub-trees and more generic rules [30]. JRIP implements the Repeated Incremental Pruning to Produce Error Reduction (RIPPER) algorithm [8] that is a rule-induction technique. For each class, JRIP starts by finding a rule that covers most of the training instances and has the best success rate (least number of misclassified instances). This procedure is repeated recursively until all instances are covered for that class and then repeated for the other classes. A procedure known as *incremental reduced-error pruning* refines each rule immediately after construction and a number of *global optimization* stages are applied after the construction of all the rules for further refinement [8]. In the case of decision trees or rule-based classifiers, pruning removes extra branches or rules, with the aim to prevent the *over-fitting* problem, where the classifier is over-fitted to the training data and cannot perform well when classifying unknown instances. In reduced-error pruning the training set is split into a growing set to construct the rules and a pruning set for pruning which means fewer instances are used for training. We select C4.5 pruning for J48 and PART. Please refer to [30] for further discussion on pruning techniques.

The use of decision branches and rules in these machine learning algorithms allows technical and medical experts to easily interpret the classifiers and gain more confidence in the decisions made by the classifier and the overall approach. Although more complicated machine learning algorithms exist, such as SVM and Neural Networks, they tend to produce outputs that are more difficult to interpret and are left to future work. Our main focus in this paper is to demonstrate our solution to the oracle problem in regards to image segmentation evaluation and, as presented in Section 4.4, the adopted classifiers perform very well in terms of classification accuracy.

To improve the performance of our machine learning algorithms, we have taken advantage of attribute selection techniques such as filters and wrappers. These techniques weed out attributes that do not add any significant improvement in building a better classifier. This is done before constructing the classifiers. Correlation-based Feature Selection (CFS) chooses a subset of attributes from the original attribute set that have a high correlation with the class and a low correlation with each other. Wrappers select attributes by first training a classifier from different subsets of the original attribute set and choosing the subset of attributes that trains the best performing classifier. Both filters and wrappers require searching the attribute space. We chose an exhaustive search method for CFS, where all the possible attribute sets are considered, and we chose a greedy method for the J48 wrapper as an exhaustive search proves to be very time consuming in this case because of the requirement to build a classifier for each attribute set. In a greedy search method such as forward hill climbing, we start from an empty subset adding attributes to the set until the addition of no attribute will result in a performance improvement at which point the current subset is the selected subset of attributes. The main motivation for choosing a wrapper and CFS filter instead of using other attribute selection techniques were the benchmarking results of different attribute selection methods reported in [14].

ISAO starts using automated test oracles when an accurate classifier for the purposes of the application has been built. A standard technique to test classifier performance is *stratified 10-fold cross-validation*. This technique splits the training set into 10-folds, each time training the classifier with 9-folds and testing it with the remaining fold. A procedure known as *stratification* randomizes the instances in each fold such that each one contains a similar proportion of the different classes. In order to prevent bias, the procedure is repeated 10 times in our experiments. Metrics such as accuracy and the area under the Receiver Operating Characteristic (ROC) curve [30] are used as indicators of performance in cross-validation. Accuracy refers to the success rate of the classifier (percentage of correct predictions). The ROC curve is a plot of the true positive (i.e. instances that are correctly classified as positive, in our case consistent) rate versus the false positive rate. The larger the area under this curve the better the classifier performs, reaching perfect performance when the area is 1.

3. Semi-automated verification and validation approach

In this section we depict our solution towards the oracle problem in the context of image segmentation evaluation. Fig. 3 shows an activity diagram depicting the flow of activities in ISAO. Two series of activities take place concurrently, as illustrated by two swimlanes: *Segmentation evaluations* and *Learning classifier*. The segmentations are evaluated manually in the segmentation evaluations swimlane (Section 3.1) until enough data is available to build an accurate classifier in the learning classifier swimlane (Section 3.2), thus allowing the automated evaluation of segmentations.

3.1. Segmentation evaluation swimlane

This swimlane has four activities: activities A–D; and is similar to the activity diagram of Fig. 2. The first time the *Devise/Update segmentation algorithm* activity (activity A) is performed, an initial version of the image segmentation algorithm is devised. Each time this activity is repeated, i.e., when the segmentations produced by the image segmentation algorithm are deemed inadequate, the image segmentation algorithm is revised. Further versions of the image segmentation algorithm are made in subsequent iterations until a satisfactory set of segmentations are produced.

During the *Segment images* activity (activity B), the segmentation algorithm produced in activity A is used to segment a set of sample images (*Test suite*) used as a benchmark. This results in a set of segmentations, each segmentation corresponding to one sample image. We refer to the segmentations obtained from version i of the segmentation algorithm (i.e., during the ith iteration of this swimlane) as segmentation set set_i.

The segmented images are verified either manually or automatically in the next two activities (C and D), depending on whether an accurate classifier has been learnt. If this is not the case (Section 3.2), the segmentations have to be manually evaluated (activity C—*Manual evaluation*). When an accurate classifier is available (Section 3.2), the evaluation is done automatically: activity D—*Automated evaluation*. In activity D, the classifier predicts, based on the similarity measurements (activity E) between the segmentations produced by the current version i of the segmentation algorithm and the segmentations produced by previous versions ($j < i$) of the segmentation algorithm, whether or not a segmentation pair is consistent.

Table 2 shows how the correctness of segmentation n produced by version i of the segmentation algorithm ($S_{n,i}$) can be obtained

[3] Entropy is a term acquired from information theory that in simple terms conveys the extent of non-uniformity in a group of instances: a group of instances that show equal proportions across classes have an entropy of 1 (entropy ranges from 0 to 1), depicting a uniform distribution.

Please cite this article in press as: K. Frounchi et al., Automating image segmentation verification and validation by learning test oracles, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.06.009

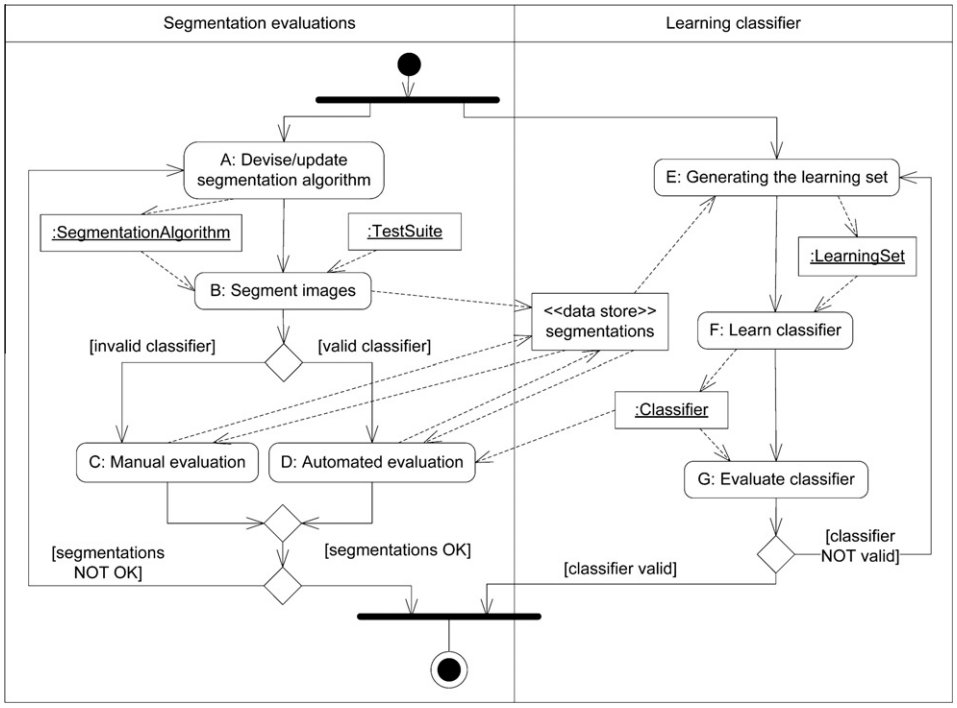

Fig. 3. Image Segmentation Automated Oracle (ISAO).

Table 2
Mapping between classifier results and the correctness evaluation of the test image segmentation.

Evaluation of $S_{n,j}$ $(j < i)$	Predicted consistency of segmentation pair ($S_{n,j}$–$S_{n,i}$)	Evaluation of $S_{n,i}$
Correct	Consistent	Correct
Correct	Inconsistent	Incorrect
Incorrect	Consistent	Incorrect
Incorrect	Inconsistent	Requires manual evaluation

from the consistency classifications of the learnt classifier. If the classifier predicts segmentation $S_{n,i}$ to be consistent (based on similarity measures) with a correct segmentation $S_{n,j}$ $(j < i)$ then segmentation $S_{n,i}$ is predicted to be correct. If segmentation $S_{n,i}$ is predicted to be inconsistent with a correct segmentation or consistent with an incorrect segmentation then it is predicted to be incorrect. In the case, where segmentation $S_{n,i}$ is predicted to be inconsistent with an incorrect segmentation, no conclusion can be drawn and the correctness of segmentation $S_{n,i}$ has to be manually evaluated by an expert. Depending on the adopted machine learning algorithms, the accuracy of each prediction can be determined by the automated oracle such that the segmentation algorithm designer can manually evaluate the questionable predictions and understand how much emphasis he should put on some results compared to others that have been made with higher accuracies when creating a new version of the segmentation algorithm.

As the number of iterations increases, more segmentations can be used to compare new segmentations with and evaluate their correctness. For example, if a segmentation $S_{n,i}$ is inconsistent with the incorrect segmentation $S_{n,i-1}$, but consistent with the correct segmentation $S_{n,i-2}$, it would be considered to be correct. In determining the correctness of a new segmentation in the presence of several prior segmentations of the same image, different policies can be adopted. The most straightforward option is to use a simple majority vote, where the segmentation is deemed (in)correct if it is predicted to be consistent with an (in)correct segmentation in the majority of comparisons. If one believes that the latest version of the segmentation algorithm is superior to all previous versions, then more weight could be given to the most recent version of the segmentation algorithm. The best policy should be empirically determined.

After either activities C or D, if more than an acceptable percentage threshold[1] of image segmentations are evaluated to be incorrect, we go back to activity A, where the image segmentation algorithm is revised. Otherwise, the testing process ends and the current version of the image segmentation algorithm is deemed to be satisfactory.

3.2. Learning classifier swimlane

This swimlane has three activities: activities E–G. In activity E (*Generating the learning set*), pairs of segmentations obtained from multiple versions of the image segmentation algorithm (current version i, and version j, $j < i$) are compared using a set of similarity measures (Section 2.1). At least the first two sets of segmentations generated by the first two versions of the segmentation algorithm are required to get the first set of similarity measurements. In other words, at least two iterations of the *Segmentation evaluations* swimlane (with manual evaluation in activity C) are necessary. More iterations may add to the accuracy of the classifier at the expense of more expert intervention.

Pairing segmentations of the same images/patients across two segmentation sets set_i and set_j results in three distinct subsets of paired segmentations. The first set is composed of the pairs of segmented images that were both deemed correct by an expert, denoted by set_{yy} (i.e., '*y*' for "yes" for the two versions). The second set is composed of the pairs of segmented images, where either the first or second segmented image was deemed incorrect and the other deemed correct, denoted by set_{yn} (one is correct: '*y*', and one is incorrect: '*n*'). The third set is the set of all the pairs of

Please cite this article in press as: K. Frounchi et al., Automating image segmentation verification and validation by learning test oracles, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.06.009

segmented images that were both considered incorrect, denoted by set_{nn}.

The machine learning algorithm (activity F) does not use set_{nn} as the information obtained from comparing two incorrectly segmented images would not help the learning algorithm construct a classifier to recognize diagnostically equivalent (consistent) segmentations. Though diagnostically equivalent segmentations refers to two segmentations that both lead to the same diagnostic by the medical expert, two segmentations may be incorrect for two completely different reasons and cannot be categorized as consistent with each other. Table 3 summarizes how we categorize a pair of segmentations to be (in)consistent, where $S_{n,i}$ and $S_{n,j}$ are two segmentations obtained from image n using versions i and j of the segmentation algorithm, respectively.

Table 3 assumes that the segmentation evaluations are binary, i.e. either correct or incorrect. This was certainly the case in our case study. In the event, where segmentations are graded with more granularity, for example each segmentation is graded from 1 to n, where grade 1 represents the lowest correctness grade and grade n the highest, a mapping to a binary scale can be defined, e.g., segmentations with grades 1–$[n/2]$ can be considered incorrect while segmentations with grades $([n/2] + 1)$ to n be considered to be correct. This obviously leads to a loss of information but it can only be resolved if a degree of correctness can be decided for every possible combination of grades for a pair of segmentations. This means if one of n grades are given to a segmentation, then $C(n, 2)$ correctness classes can be assigned. For example if we have four grades (grades 1–4) a class will represent the case, where one segmentation has grade 1 and the other has grade 3 and another one represents the case, where both segmentations are graded 4. A first challenge in doing so is that since we have many more classes, having enough examples for each class is a concern when training the classifier. The reliability and subjectivity of the grading in such a fine scale is also a concern. The assignment of a correctness class also tends not to be as intuitive with multiple grades. A binary correctness scale is therefore, at this stage of the research, more practical. More discussion on this subject is left to future work.

The application of the similarity measures to image pairs in set_{yy} and set_{yn} generates a set of tuples in the form of $(sm_{i1}, \ldots, sm_{ik}$, consistency), where sm_{ij} denotes similarity measure j on image pair i and "Consistency" can take two values: yy (consistent) or yn (inconsistent). Index k is the number of similarity measures. This set of tuples is the training set that is fed into the machine learning algorithm: activity F (*Learn Classifier*). The machine learning algorithm learns which combination of measures and which ranges of their values depict consistent or inconsistent segmentation pairs. Before using the machine learning algorithms to train the classifiers, experiments are done to find the optimal set of attributes. This is done through attribute selection techniques described in Section 2.2 (practical results are shown in Section 4.4.2). Any machine learning algorithm can be plugged into ISAO. Experiments are reported in Section 4.4.3 to find the best segmentation algorithm among the three chosen for demonstrating and assessing ISAO.

Table 3
Consistency of two segmentations.

Manual evaluation of $S_{n,i}$	Manual evaluation of $S_{n,j}$	Class
Correct	Correct	Consistent
Correct	Incorrect	Inconsistent
Incorrect	Correct	Inconsistent
Incorrect	Incorrect	Unusable (not a class)

The learned classifier (activity F) is validated using techniques such as 10-fold cross-validation (Section 2.2) in activity G (*Evaluate classifier*). A classifier is deemed *valid* if its accuracy is deemed to be sufficiently high to be used in practice. This means that, given the needs and constraints of the application domain, the classifier correctly classified, using the segmentation measurements, a *reasonable* number of segmentation pairs to be consistent or inconsistent. Once the valid classifier is learnt, the evaluation in the *Segmentation evaluation* swimlane can be performed automatically (activity D). To do so, the similarity measures depicted in Table 1 are collected for the unevaluated segmentations of the new version of the segmentation algorithm and the already evaluated segmentations of a previous version, to obtain tuples of the form $(sm_{i1}, \ldots, sm_{ik})$, using the same notation as the previous paragraph. These tuples are then used as inputs to the classifier which then assigns a "consistent" or "inconsistent" class to each tuple, at which point Table 2 is used to assess the correctness of each new segmentation.

In order for the automated evaluation activity (activity D) to kick in, at least two iterations of the segmentation evaluations swimlane are necessary. This is to provide the training data necessary to train the classifier. If an accurate classifier is not obtained after the first two iterations, additional iterations of manual evaluations must be performed to provide more training data until a valid classifier is learnt. With k iterations of manual evaluations and n test cases, $n \times C(k, 2)$ training instances are available. Even in the case, where we do not succeed in learning a classifier with satisfactory *average* accuracy, we can expect that there will always be some parts of the classifier predictions with very low error rates that can be trusted with high confidence. For example, particular branches in decision trees or rules in rule induction techniques may exhibit much higher accuracy than other branches or rules. If the classifier can provide accurate predictions for a practically significant number of cases, then the classifier is a good candidate for partially automating the segmentation evaluations.

When testing industry strength image segmentation algorithms, the accuracy of these algorithms is paramount as they are usually applied for diagnostic purposes (such as our left ventricle segmentation case study). As a result, image benchmarks are usually large enough to provide enough data to feed the machine learning algorithm. Consequently, the need for finding an automatic approach to testing such algorithms is acute as a large number of segmentations have to be verified by medical experts through many iterations. With ISAO though, medical experts are still required for the first few iterations and human errors are therefore possible. An interesting study on the trade-off between the expense of finding more accurate training data versus the use of training data that may have some noise shows that if the size of the training data is substantial, the effect on the performance of a classifier is minimal [17]. ISAO trains classifiers that determine consistency between old and new segmentations and then derives the correctness of each new segmentation based on prior correctness information about the old segmentations. It does not train a classifier that can assess correctness directly from a segmentation. Training such classifiers would require measures that can help assess the correctness of a standalone segmentation without using any medical expert knowledge.

4. Case study

This section describes the application of ISAO to the verification and validation of a left ventricle segmentation algorithm. As defined by IEEE, *validation* is "the process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements".

Please cite this article in press as: K. Frounchi et al., Automating image segmentation verification and validation by learning test oracles, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.06.009

IEEE also defines *verification* as "the process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase". We present here an automated oracle for the assessment of segmentation algorithms based on a benchmark of images (*validation*); the requirement here is the usage of the segmentation algorithm for diagnostic purposes. Our approach also includes *verification* since the correctness of the segmentation algorithm, which is a condition imposed at the start of each revision, is checked after every revision of the segmentation algorithm. The layout of this section is as follows: The study objectives and design are explained in Section 4.1. We then describe the different attribute sets (similarity measures) that we have used in order to train the best performing segmentation consistency classifiers and thus oracle (Section 4.2), identify the best performing classifiers and provide insights on how to interpret them (Section 4.3), and compare the performance obtained with various attribute sets, machine learning algorithms, and the filtering methods (Section 4.4).

4.1. Study objectives and design

In this case study we intend to reach the following objectives:

- Analyze the performance of ISAO on a cardiac left ventricle segmentation algorithm devised by Siemens Corporate Research. Our goal is to determine if ISAO can lead to practical benefits.
- Investigate which similarity measures have the highest impact in determining the consistency of two segmentations and understand the tradeoff between using more expensive measures (in terms of run-time and complexity) and achieving better classifier performance.
- Analyze and compare the performance of several classification algorithms combined with attribute selection techniques to determine whether they are appropriate for our application and produce acceptable results.

The segmentation algorithm we use in this case study identifies the left ventricle of the heart[4] from a Computed Tomography (CT) Scan. CT-Scan images of the heart of 50 patients have been taken at different times during the cardiac cycle, resulting in 181 CT-Scans. Each CT-Scan is a 3D image that comprises a set of 2D images (slices). All the 2D slices, together, form the 3D image. The segmentation algorithm and similarity measures work on 3D images. Due to the variety in acquisition protocol and equipment across clinical centers, each CT-Scan has between 53 and 460 2D images, and each 2D image depicts either a 256 by 256 or 512 by 512 pixel slice of the heart. Each CT-Scan, representing a patient's heart at a given instant, constitutes a test case in the test suite. The segmentations are obtained using a form of the general segmentation algorithm presented in [13] that is customized for cardiac segmentation.

Two successive versions of the segmentation algorithm were used to obtain the segmentations of the 181 CT-Scans, involving a medical expert for the evaluation of each segmentation. In other words, activities A–C in Fig. 3 were executed twice. Pairs of segmentations from the same CT-Scan, output from the two segmentation algorithm versions, are compared using the 18 similarity measures of Table 1 (activity E in Fig. 3). 181 tuples or training instances (Section 3.2) are thus generated combining similarity measures and the consistency class of each pair: 104 tuples are found to be consistent (positive instances) while 77 tuples are Inconsistent (negative instances).

[4] The volume of the left ventricle can then be computed at different times during the cardiac cycle for diagnostic purposes.

Table 4
Study description summary.

Segmentation target	Left ventricle of the heart
Number of patients	50
Number of test cases in test suite	181
Number of attributes (measures)	18
Number of positive instances	104
Similarity measure types	Overlap, volume difference, geometrical
Machine learning algorithms	J48, JRIP, PART
Classifier evaluation technique	Stratified 10-fold cross-validation
Performance metrics	Accuracy, area under ROC curve
Statistical test	*t*-test (95% confidence level)

As explained in Section 2.2, the J48, JRIP, and PART machine learning algorithms were chosen to train classifiers (activity F in Fig. 3). Though activity F does not require the use of several classification algorithms, in our research work, we are interested in evaluating alternative algorithms to determine the best option for our application. The performance of these algorithms is evaluated using 10 stratified 10-fold cross validations by measuring their classification accuracy and the area under the ROC curve (activity G in Fig. 3). Any *significant* differences with respect to the choice of attribute set type (Section 4.4.1), attribute selection technique (Section 4.4.2) or machine learning algorithm (Section 4.4.3) is measured using the statistical *t*-test with a confidence level of 95%. All this information is summarized in Table 4.

4.2. Attribute sets

Recall from Section 2.1 that we consider three types of measures: volume difference, overlap, and geometrical measures. The former two types of measures do not consider the shape differences between the two segmentations, thus being less complex in terms of computation, and also take less run-time compared to the geometrical measures. The overlap and volume difference measures have therefore been combined together, and we will refer to these measures as *simple* measures. We are interested in studying the changes in performance of the learning process when using only the two least expensive types of measures (i.e., simple measures), the most expensive type of measures alone (i.e., geometrical measures), or when using all measures. In order to investigate how the classifier performance would change when built using a filtered set of attributes we have applied the Correlation-based Feature Selection (CFS) filter and a J48 based wrapper. Applying (or not) the two filtering mechanisms to the three groupings of measures, we end up considering nine different attribute/measure sets (Table 6) to train the classifiers.

Some of these measures show high correlation with each other as shown by Spearman's rank correlation values in Table 5. Highly correlated attributes will not add any significant discriminating power in terms of learning the class of instances and large numbers of attributes are known to possibly affect the resulting accuracy of the classifier. Since in this work the number of attributes is small, we let the machine learning algorithms decide which ones to use while also taking advantage of effective techniques to select attributes, such as filters and wrappers (Section 2.2).

Table 5
Spearman rank correlations between similarity measures.

TC	DSC	1.000
SODI	ODI	0.995
SUDI	UDI	0.992
HD	BD	0.961
PMME	PMMSE	0.955

Please cite this article in press as: K. Frounchi et al., Automating image segmentation verification and validation by learning test oracles, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.06.009

Table 6
Attribute sets.

	Attribute set	Selection criteria	Label
1	TC–DSC–TPVF–FPVF–AVD–ANVD	Overlap and volume difference measures	OV
2	TC–DSC–AVD–ANVD	Overlap and volume difference measures chosen by the CFS filter using exhaustive search	OVC
3	TC–FPVF	Overlap and volume difference measures chosen by the J48 decision tree wrapper using forward selection greedy search	OVW
4	BD–HD–PFOM–RMSSD–ADBD–SODI–ODI–SUDI–UDI–PAD–PMME–PMMSE	Geometrical measures	G
5	HD–BD–PFOM–RMSSD–ODI–UDI–PAD	Geometrical measures chosen by the CFS filter using exhaustive search	GC
6	PFOM–SODI–SUDI	Geometrical measures chosen by the J48 decision tree wrapper using forward selection greedy search	GW
7	TC–DSC–TPVF–FPVF–AVD–ANVD–BD–HD–PFOM–RMSSD–ADBD–SODI–ODI–SUDI–UDI–PAD–PMME–PMMSE	All measures	A
8	TC–DSC–AVD–ANVD–HD–RMSSD–ODI–UDI	All measures chosen by the CFS filter using exhaustive search	AC
9	TC–FPVF–SODI	All measures chosen by the J48 decision tree wrapper using forward selection greedy search	AW

4.3. Classifiers

Before we delve into the details of the classifiers, let us summarize the configuration parameters of the machine learning algorithms (Table 7). The parameters for J48 and PART are similar as PART uses partial decision trees in its construction process. The number of instances per leaf node parameter restricts the minimum number of instances in a leaf node to four. Higher values for this parameter take away the freedom of C4.5 to form leaf nodes but very low values also may cause over-fitting to the data. Initial experiments were conducted with WEKA in order to fine tune these parameters. For J48/PART, the number of instances per leaf node parameter was swept from 2 to 10, each time training with one of the attribute sets in Table 6 The chosen value (4) was either significantly better or equivalent to the other values of this parameter in the majority of cases. Statistical significance was determined using the *t*-test (with a significance level of 0.05) when comparing the accuracies obtained from performing 10 times a stratified 10-fold cross-validation using each value of this parameter. Similarly, the number of folds for pruning and the number of global optimizations parameters were optimized by sweeping each parameter from 2 to 10. C4.5 pruning also proves to be either significantly better or equivalent to reduced error pruning, which is the motivation behind its selection for J48/PART.

Fig. 4 shows the best performing decision tree trained with the wrapper selected attributes (attribute set AW in Table 6) and produced by J48. Based on performing 10 times a 10-fold cross validation, this classifier's accuracy is 94.92% and its ROC area is 0.95. The numbers shown in the parentheses in Fig. 4 (from left to right) represent the number of training instances that reach each leaf node and the number of these instances that are incorrectly classified by that leaf node, respectively.

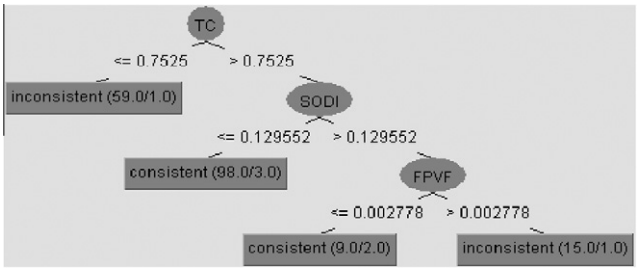

Fig. 4. Best decision tree classifier.

Reading the decision tree shows that if the overlap (using measure TC) between the two segmentations is less than approximately 75% then the segmentations are categorized as inconsistent (as confirmed by 59 of the 74 inconsistent pairs in the training set) with only 1 misclassified instance, otherwise a distance of less than 0.13 determined by the geometrical measure SODI will lead to a consistent pair (confirmed by 98 of the 107 consistent pairs) with only 3 misclassifications. The overlap measure FPVP helps classify the remaining pairs of segmentations when the TC overlap is greater than 75% and the SODI distance is greater than 0.13. If the FPVP overlap between the two segmentations is less than 0.28% then we have a consistent pair again, otherwise the segmentations are inconsistent. The branch predictions of consistent pairs can be considered quite accurate with a minimum error rate of 3% (TC–SODI-consistent branch) and a maximum error rate of 22% (TC–SODI–FPVF-consistent branch), which only applies to a small number of the training instances (two errors out of only 9 instances). In practice, the maximum acceptable error rate should be determined for each application, a threshold above which the manual verification of a segmentation should be required. Assuming a maximum error rate threshold of say 5% is acceptable, then we see that most classifications would be acceptable (157 out of 181).

Trained with the same attribute set (set AW), PART constructs four rules:

Rule 1: if ((TC > 0.7525) && (SODI ⩽ 0.129552)) then class = consistent (98/3).
Rule 2: else if (TC ⩽ 0.76311) then class = inconsistent (59/1).
Rule 3: else if (FPVF > 0.002778) then class = inconsistent (15/1).
Rule 4: else class = consistent (9/2).

Interpreting each of the branches in the decision tree of Fig. 4 as a rule, we see that the PART rules are essentially the same as the J48 classifier in Fig. 4 with a small difference in the threshold used for TC for identifying inconsistent pairs (0.76311 in the second rule versus 0.7525 in the decision tree). This stems from the fact that PART constructs a different decision tree to generate the second rule, which means that it may not necessarily come up with the same threshold as the first rule when trying to cover the remaining instances that are not covered by rule 1.

On the same attribute set (set AW), JRIP generates three rules:

Rule 1: if (TC ⩽ 0.737932) then class = inconsistent (58/1).
Rule 2: else if ((SODI ⩾ 0.131275) && (FPVF ⩾ 0.002838)) then class = inconsistent (16/1).
Rule 3: else class = consistent (107/5).

Table 7
Classifier configuration parameters.

Parameter	Value	Algorithm
Number of instances per leaf node	4	J48/PART
Pruning method	C4.5	J48/PART
Number of folds for pruning	3	JRIP
Number of global optimizations	4	JRIP

Please cite this article in press as: K. Frounchi et al., Automating image segmentation verification and validation by learning test oracles, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.06.009

JRIP covers all the instances that belong to one class before proceeding to the next class, whereas PART uses partial decision trees for rule induction and depending on the chosen leaf at each stage, instances belonging to a different class may be covered. Again, the rules produced by JRIP are very similar to the ones produced by J48 and PART, with small differences in thresholds used for TC, SODI and FPVP.

We see from all three classifiers that TC is the best discriminator for negative instances (inconsistent pairs) while the combination of TC and SODI is the best discriminator for positive instances (consistent pairs).

Unfortunately, there is no existing oracle for this problem to which we could compare the learned classifiers—this is the oracle problem and it is precisely why we try to learn the expert's opinion.

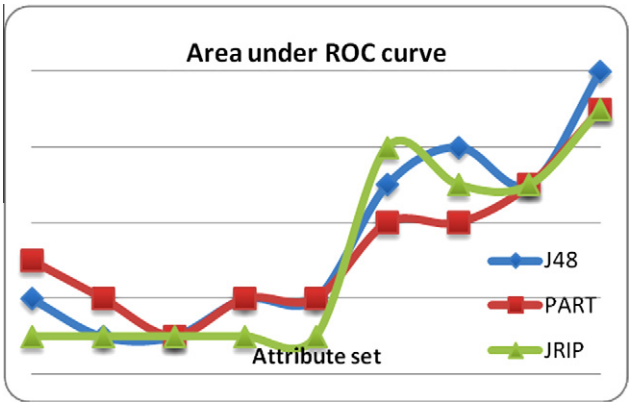

Fig. 6. Average area under the ROC curve for all classifiers.

4.4. Classifier performance comparisons

In this section we investigate the performance of the classifiers with respect to the attribute set (Section 4.4.1), the application of filters/wrappers (Section 4.4.2) and the machine learning algorithm (Section 4.4.3). We refer to Figs. 5 and 6 throughout the discussion. Each point in these figures is the average across ten 10-fold cross validations. We use the *t*-test to find any *significant* differences in the results. The paired *t*-test is a standard statistical test used to evaluate whether the null hypothesis specifying that two random samples have the same mean can be rejected or not (with a specified degree of confidence). If rejected, this would indicate that the means are significantly different. The *t*-test assumes that the mean follows a normal distribution regardless of the distribution of the samples themselves, which is a valid assumption if we have large enough samples. In our case, the *t*-test has been performed on the mean of the accuracies that have been obtained after performing 10 times stratified 10-fold cross validation creating a sample size of 100 (a reasonable sample size). We have used a confidence level of 95% to report our results.

4.4.1. Effect of attribute set type

In this section, we compare the performance of the classifiers trained with the geometrical measures to the performance of the classifiers trained with the simple measures (overlap and volume difference) and then investigate how using all the measures will affect the classifier performance using the accuracy and ROC area metrics.

As Figs. 5 and 6 show, the geometrical measures (set G) do not show any noticeable performance improvement compared to just using the simple measures (set OV). This is also confirmed with a *t*-test showing there is no *significant* performance difference between these two sets. This result indicates that using solely geometrical measures may not be required in this case study and thus may not be required in activity F of Fig. 3 to learn a classifier. Considering that the geometrical measures result in no performance improvement and also are more complex to implement and test, this increases the applicability of our approach. However, combining the geometrical and simple measures (set A) seems to be very promising. We see in Table 8 significant accuracy improvements (statistically significant with a confidence level of 95%) for all three classifiers compared to just using either of the simple (from 4.82% to 6.47%) or geometrical measures (from 4.91% to 7.35%). A similar trend is visible for the ROC areas (Fig. 6). This shows that the classifiers achieve very high discriminating power when taking advantage of both the simple and geometrical measures.

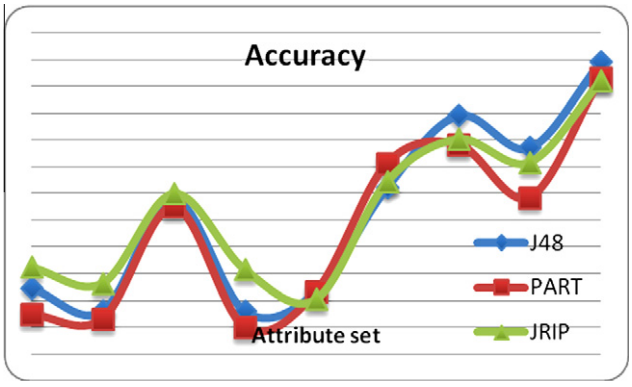

Fig. 5. Average accuracy for all Classifiers.

4.4.2. Effects of wrappers and filters

The effect of filtering and using wrappers is investigated in this section. Table 9a–c shows respectively the classifier accuracies achieved when trained by the simple, geometrical and all measures sets before and after applying the CFS filter and J48 wrapper. In Table 9, "Not Sig." means that there is no significant difference (as indicated by performing the *t*-test with 0.05 significance level) between the two trained classifiers to be reported; all the reported numbers indicate statistically significant differences. We see that using the training instances with only the attributes selected by the wrapper always achieves a significant improvement in the accuracy of the trained classifier compared to using all the attributes: the maximum accuracy improvement is 7.66% for the J48 classifier when using the geometrical measures. Similar trends were observed with the ROC area. On the contrary the CFS filter in some cases significantly degrades the classifier's accuracy. Using a PART wrapper or a JRIP wrapper also improves classifier accuracy, but the improvement is not significantly better than the J48 wrapper using any combination of classifier/attribute set and thus, we do not report these results here.

Table 8
Comparison of classifier accuracies with respect to the attribute set category.

	Sets				
	OV	G	A	Δ (A–OV)	Δ (A–G)
J48	86.46	85.58	92.93	6.47	7.35
PART	85.46	84.98	91.83	6.37	6.85
JRIP	87.23	87.14	92.05	4.82	4.91

Please cite this article in press as: K. Frounchi et al., Automating image segmentation verification and validation by learning test oracles, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.06.009

Table 9
Comparison of classifier accuracies with respect to the use of filters and wrappers.

(a) Using only simple measures

	OV	OVC	OVW	Δ (OVC–OV)	Δ (OVW–OV)
J48	86.46	85.59	89.78	−0.87	4.19
PART	85.46	85.30	89.50	Not sig.	4.04
JRIP	87.23	86.63	90.00	Not sig.	2.77

(b) Using only geometrical measures

	G	GC	GW	Δ (GC–G)	Δ (GW–G)
J48	85.58	86.31	93.24	Not sig.	7.66
PART	84.98	86.31	90.13	1.33	5.15
JRIP	87.14	86.03	90.46	Not sig.	3.32

(c) Using all measures

	A	AC	AW	Δ (AC–A)	Δ (AW–A)
J48	92.93	91.71	94.92	−1.22	1.99
PART	91.83	89.84	94.32	−1.99	2.49
JRIP	92.05	91.17	94.21	Not sig.	2.16

Table 10
Comparison of classifier accuracies with respect to the machine learning algorithm.

	J48	PART	JRIP	Δ (J48–PART)	Δ (J48–JRIP)	Δ (PART–JRIP)
OV	86.46	85.46	87.23	1	Not sig.	−1.77
OVW	89.78	89.50	90.00	Not sig.	Not sig.	Not sig.
G	85.58	84.98	87.14	Not sig.	−1.56	−2.16
GW	90.24	91.13	90.46	−0.89	Not sig.	Not sig.
A	92.93	91.83	92.05	1.1	Not sig.	Not sig.
AW	94.92	94.32	94.21	0.6	0.71	Not sig.

4.4.3. Effects of different machine learning algorithms

In this section we investigate whether the different machine learning algorithms yield significant performance differences. Table 10 shows that there is not a noticeable difference in the accuracy of the classifiers trained with the three machine learning algorithms. Similar trends were observed with the ROC area. JRIP trains a better classifier when only accounting for geometrical measures, performing 2.16% better than PART and 1.56% better than J48. When using all the measures, whether applying the wrapper or not, J48 generally performs slightly better. We have not considered the results for the classifiers that are trained with the CFS filtered attribute sets (OVC, OG, AC in Table 6) as these classifiers do not achieve any significant accuracy improvement over just using the original attribute sets (please refer to Section 4.4.2). From Table 10, one can conclude that using any of the three machine learning algorithms would provide equivalent performance, however one may consider interpreting decision trees to be easier than rules or vice versa.

5. Related work

There is a large body of work in the image processing field on studying and improving image segmentation algorithms but this is not the focus of our work. Rather we try to find a solution to the oracle problem when testing (medical) image segmentation algorithms that are iteratively improved based on large image benchmarks. Recall that the oracle problem in software testing is the problem of finding a procedure to assess the correctness of test results (in our case image segmentations) [19].

Defining an oracle is not a trivial task and may not be feasible at all times. In [10,29] different solutions to the oracle problem are proposed: design redundancy, data redundancy, consistency checks, and the use of simplified data. Design redundancy attempts to check if the output of the software under test corresponds to one (or many) extra implementations of the specification of the software under test. It may be that the output of all the software versions is incorrect which is less likely if it is assumed that the software versions are independent. The independence assumption has been empirically debated in [16], showing that it may not hold at all times. Design redundancy is similar to multi-version programming in regression testing [3]. ISAO uses an approach which is similar to design redundancy since we use different implementations of the segmentation algorithm however we do not add redundant versions of the software under test rather the different versions of the software under test are the product of each iteration of the segmentation evaluation swimlane in ISAO. In [7], a methodology is proposed that uses design redundancy to identify failures in mesh simplification programs, i.e., programs which attempt to reduce the complexity of graphical objects to improve the processing performance of these objects. In this methodology, other already existing programs (referred to as reference models) and their mutations are used to train a classifier capable of finding faulty outputs from the program under test. The experiments in [7] show that it is advisable to have reference models that resemble the program under test. The best classification accuracies in [7] range from 60% to 70%.

In data redundancy, similar to N-copy programming in the fault tolerance domain [4], the input is re-expressed in different forms and the output of the original input is checked for agreement with the re-expressed inputs. In [21] the authors propose an approach to help automate metamorphic testing (a data redundancy technique). Of course the inputs to the application require the definition of metamorphic properties for this kind of testing. These are properties of the application domain, which can be used to produce the same output from different inputs.

Consistency checking simplifies the oracle to just verifying whether certain plausible conditions are met by the software under test, and testing with simplified inputs ignores testing with more complex inputs and only considers simpler inputs for which an oracle can be devised.

None of these approaches offer any quantification of the reliability of the oracle. In the case, where formal specifications exist for the software under test, researchers have proposed methods to derive oracles from formal specifications. This has been investigated for real-time systems in [26], where the test case consists of some test executions or sequences of stimuli. Test oracles in the form of assertions are obtained using symbolic interpretation of the specifications for each control point. A discussion on oracles that are composed of assertions based on properties specified either in first order logic or computation tree logic is given in [18]. Assertions under the form of pre and post-conditions can also be used as test oracles [6].

In [31], the authors use machine learning for evaluating the best of two segmentations, produced by two different segmentation algorithms (or two different settings of one segmentation algorithm). The authors have a fundamentally different goal. Specifically, they attempt to find the better of two segmentations for an image while we try to find whether the unknown segmentation is (in)consistent with an already evaluated segmentation in order to detect (in)correct segmentations (which does not correspond to a better/worse segmentation). Also, the authors do not use any similarity measures to compare the two segmentations as we do for finding (in)consistent segmentation pairs. Instead, they evaluate the goodness (e.g., color uniformity) of each segmentation by applying measures that target each segmentation separately. Each segmentation measurement leads to one decision tree, and the comparison of segmentations is obtained by combining the results of all the decision trees using meta-learning. It is also noteworthy that their measures require the processing of image properties

Please cite this article in press as: K. Frounchi et al., Automating image segmentation verification and validation by learning test oracles, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.06.009

such as color and texture while our measures only rely on the data from the segmentation labeling. In [33], the authors attempt to fix segmentation outputs generated by unknown segmentation algorithms. To do so, they apply measures to the test segmentations that include spatial and intensity information. Adaboost (an algorithm that builds stronger classifiers from multiple weak classifiers) is used to find the combination of these measures that best classify the mislabeled voxels. Manual segmentations created by experts are required to train the classifier. This is in contrast to our work, which tries to find a solution for helping image segmentation developers to systematically test the revisions of their algorithm during its evolution. Additionally, our approach only relies on expert judgment, which is less expensive than asking experts to create segmentations (so called ground truth). In [20], an application is designed for comparing ground truth segmentations with segmentations output by the segmentation algorithm under test. The purpose of the application is to provide better visualization of the output of the segmentation algorithms.

6. Conclusions

Image segmentation is widely used in medical imaging and many other application domains. The production of a commercial system requires the iterative development of specific algorithms that must be repeatedly verified based on large benchmarks of images.

This paper addresses the automation of the verification and validation process of an image segmentation algorithm, through the (semi)automated construction of a test oracle. In this paper we propose an approach (referred to as ISAO – Image Segmentation Automated Oracle) to replace the expensive, manual verification of segmentations by medical experts with an automated oracle. Its construction relies on machine learning to construct a classifier from similarity measures that can predict the consistency between two segmentations. This is then used to predict the correctness of new segmentations for a benchmark of images as the algorithm evolves. ISAO is generic in the sense that it can be applied to any image segmentation software that goes through such an iterative development and verification and validation procedure. Substantial time and effort required by (medical) experts can therefore be saved. Guidance as to how the segmentation algorithm can be improved is provided by the learned classifier, thus leading to faster development time to delivery and hopefully increased testing.

In this paper, we have only introduced the application of ISAO in a two-class segmentation problem (where the object of interest is one class and the background is the other class). ISAO can also be used in multi-class segmentation algorithms (i.e., where multiple objects are each identified with one class in the segmentation) since it can be applied for every class independently. The application of ISAO to other forms of image processing is potentially interesting and should be investigated in future work. Any image processing algorithm that can be graded as correct/in-correct by a human evaluator and, where metrics can be found for comparing the output images is a plausible candidate for ISAO. Imaging problems such as inpainting or colorization are potential candidates to investigate.

We investigated the performance of ISAO on the evaluation of industry-strength cardiac left ventricle segmentation algorithms based on real 3-D medical images (CT scans). Although there is room for improvement, the results are very promising and fairly simple classifiers show very good classification accuracies, thus suggesting that the correctness of most segmentations can be determined automatically and with high accuracy. For example, using C4.5 decision trees with all available similarity measures, and relying on a wrapper for filtering candidate measures, we achieve an average accuracy of approximately 95% and an average area of 0.95 under the ROC curve when only using the training data obtained from the first two versions of the segmentation algorithm. Using machine learning also helps to understand which similarity measures are relevant in determining what differences between segmentations are important from a medical standpoint. Results also show that the choice of a particular learning algorithm is not significant, among the ones we considered here for generating logical rules. Using a wrapper to pre-select similarity measures is however effective. For maximum accuracy, all types of similarity measures should be combined to construct classifiers.

Future work will be directed towards testing the performance of ISAO on various other segmentation applications. Most of the measures defined in this report can be re-used in other segmentation contexts as the measures are not dependant on the left ventricle segmentation problem.

Even though the results in this study are promising, more experiments need to be conducted in order to investigate (1) how many segmentation algorithm iterations are required for the machine learning algorithm to be able to converge towards an accurate classifier in other applications, (2) whether the set of measures is complete enough (other segmentation algorithms may require other measures), (3) the impact of the expert during manual evaluation (possible mistakes, possible disagreements between experts), (4) the impact of other evaluation frameworks (e.g., the mapping between expert evaluation under the form of a scale and the correct/incorrect evaluation), and (5) the use and improvement of the ISAO tool support itself.

References

[1] E. Abdou, W.R. Pratt, Quantitave design and evaluation of enhancement/thresholding edge detectors, in: Proceedings of the IEEE, vol. 67 (5), 1979, pp. 753–763.
[2] T. Acharya, A.K. Ray, Image Processing: Principles Applications, Wiley-Interscience, 2005.
[3] P. Amman, J. Offut, Introduction to Software Testing, Cambridge University Press, 2008.
[4] P.E. Ammann, J.C. Knight, Data diversity: an approach to software fault tolerance, IEEE Transactions on Computers 37 (4) (1988) 418–425.
[5] A.J. Baddeley, An error metric for binary images, in: Proceedings of the International Workshop on Robust Computer Vision, 1992.
[6] L.C. Briand, Y. Labiche, H. Sun, Investigating the use of analysis contracts to improve the testability of object oriented code, Software Practice and Experience 33 (7) (2003) 637–672.
[7] W.K. Chan, S.C. Cheung, J.C.F. Ho, T.H. Tse, PAT: a pattern classification approach to automatic reference oracles for the testing of mesh simplification programs, Journal of Systems and Software 82 (3) (2009) 422–434.
[8] W.W. Cohen, Fast effective rule induction, in: Proceedings of the International Conference on Machine Learning, 1995, pp. 115–123.
[9] W.R. Crum, P. Camara, D.L.G. Hill, Generalized overlap measures for evaluation and validation in medical image analysis, IEEE Transactions on Medical Imaging 25 (11) (2006) 1451–1461.
[10] M.D. Davis, E.I. Weyuker, Pseudo-oracles for non-testable programs, in: Proceedings of the ACM Annual Conference, 1981, pp. 254–257.
[11] X. Deng, L. Zhu, Y. Sun, C. Xu, L. Song, J. Chen, R.D. Merges, M.P. Jolly, M. Suehling, X. Xu, On simulating subjective evaluation using combined objective metrics for validation of 3D tumor segmentation, in: Proceedings of the Medical Image Computing and Computer-Assisted Intervention, 2007, pp. 977–984.
[12] K. Frounchi, L.C. Briand, L. Grady, Y. Labiche, R. Subramanyan, Automating Image Segmentation Verification and Validation by Learning Test Oracles, Carleton University, Technical Report SCE-09-06, 2009. <squall.sce.carleton.ca/pubs/tech_report/TR_SCE_09-06.pdf>.
[13] L. Grady, Random walks for image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (11) (2006) 1768–1783.
[14] M.A. Hall, G. Holmes, Benchmarking attribute selection techniques for discrete class data mining, IEEE Transactions on Knowledge and Data Engineering 15 (6) (2003) 1437–1447.
[15] R. Klette, A. Rosenfeld, Digital Geometry: Geometric Methods for Digital Picture Analysis, Morgan Kaufmann, 2004.
[16] J.C. Knight, N.G. Leveson, An experimental evaluation of the assumption of independence in multi-version programming, IEEE Transactions on Software Engineering 12 (1) (1986) 96–109.

Please cite this article in press as: K. Frounchi et al., Automating image segmentation verification and validation by learning test oracles, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.06.009

[17] C.P. Lam, D.G. Stork, Evaluating classifiers by means of test data with noisy labels, in: Proceedings of the International Joint Conference on Artificial Intelligence, 1995, pp. 513–518.
[18] P.D.L. Machado, W.L. Andrade, The oracle problem for testing against quantified properties, in: Proceedings of the IEEE International Conference on Quality Software, 2007, pp. 415–418.
[19] A.P. Mathur, Foundations of Software Testing, Addison Wesley Professional, 2007.
[20] K. McGuinness, G. Keenan, T. Adamek, N.E. O'Conner, Image segmentation evaluation using an integrated framework, in: Proceedings of the IET International Conference on Visual Information Engineering, 2007.
[21] C. Murphy, K. Shen, G. Kaiser G. Automatic system testing of programs without test oracles, in: Proceedings of the ACM International Symposium on Software Testing and Analysis, 2009, pp. 189–200.
[22] E. Neri, D. Caramella, C. Bartolozzi, Image Processing in Radiology: Current Applications, Springer, 2008.
[23] C. Odet, B. Belaroussi, H. Benoit-Cattin, Scalable discrepancy measures for segmentation evaluation, in: Proceedings of the IEEE International Conference on Image Processing, vol. 1, 2002, pp. 785–788.
[24] T. Peli, D. Malah, A study of edge detection algorithms, Computer Graphics and Image Processing 20 (1) (1982) 1–21.
[25] D.L. Pham, C. Xu, J.L. Prince, A survey of current methods in medical image segmentation, Annual Review of Biomedical Engineering 2 (2000) 315–338.
[26] D.J. Richardson, S.L. Aha, T.O. O'Malley, Specification based test oracles for reactive systems, in: Proceedings of the ACM International Conference on Software Engineering, 1992, pp. 105–118.
[27] C. Rosenberger, S. Chabrier, H. Laurant, B. Emile, Unsupervised and supervised image segmentation evaluation, in: Y. Zhang (Ed.), Advances in Image and Video Segmentation, IGI Global, 2006, pp. 365–393.
[28] J.K. Udupa, V.R. Leblanc, Y. Zhuge, C. Imielinska, H. Schmidt, L.M. Currie, B.E. Hirsch, J. Woodburn, A framework for evaluating image segmentation algorithms, Computerized Medical Imaging and Graphics 30 (2) (2006) 75–87.
[29] E.J. Weyuker, On testing non-testable programs, The Computer Journal 25 (4) (1982) 465–470.
[30] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, second ed., Elsevier, 2005.
[31] H. Zhang, S. Cholleti, S.A. Goldman, J. Fritts, Meta-evaluation of image segmentation using machine learning, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2006, pp. 1138–1145.
[32] Y. Zhang, Advances in Image and Video Segmentation, IRM Press, 2006.
[33] H. Wang, S. Das, J.W. Suh, M. Altinay, J. Pluta, C. Craige, B. Avants, P. Yushkevich, A learning-based wrapper method to correct systematic errors in automatic image segmentation: consistently improved performance in hippocampus, cortex and brain segmentation, Neuroimage 55 (3) (2011) 968–985.

Please cite this article in press as: K. Frounchi et al., Automating image segmentation verification and validation by learning test oracles, Inform. Softw. Technol. (2011), doi:10.1016/j.infsof.2011.06.009