Pattern Classification with Rejection Using Cellular Automata-Based Filtering

Agnieszka Jastrzebska, Rafael Toro Sluzhenko

Pattern Classification with Rejection Using Cellular Automata-Based Filtering

Agnieszka Jastrzebska and Rafael Toro Sluzhenko

Faculty of Mathematics and Information Science, Warsaw University of Technology
ul. Koszykowa 75, 00-662 Warsaw, Poland
A.Jastrzebska@mini.pw.edu.pl,toror@student.mini.pw.edu.pl

Abstract. In this article we address the problem of contaminated data in pattern recognition tasks, where apart from native patterns we may have foreign ones that do not belong to any native class. We present a novel approach to image classification with foreign pattern rejection based on cellular automata. The method is based only on native patterns, so no knowledge about characteristics of foreign patterns is required at the stage of model construction. The proposed approach is evaluated in a study of handwritten digits recognition. As foreign patterns we use distorted digits. Experiments show that the proposed model classifies native patterns with a high success rate and rejects foreign patterns as well.

1 Introduction

Pattern recognition is an important branch of machine learning and data mining. In a typical pattern recognition scenario, we assume that there is a finite number of classes (categories) of patterns on which we want to form an automated mechanism that assigns appropriate class labels to patterns appearing on its input. The process of learning is based on an already labeled subset of patterns that we have at hand and we use to train the classifier. Ideally, we wish that the trained classifier will be able to recognize not only patterns used at the stage of training, but also new patterns that it had not seen before.

In this paper we address the problem of contaminated datasets, where apart from proper patterns, that we call *native patterns*, we have garbage patterns, so called *foreign patterns*. Processing of such dataset aims at:

- classifying native patterns and
- rejecting foreign patterns.

The problem of contaminated datasets is especially valid when we are collecting pattern samples in an out-of-lab environment. It may happen that preprocessing mechanisms (image binarization, gray-scale transformation, filtering, segmentation, etc.) do not perform well and we obtain incorrect patterns. Foreign patterns contaminate the dataset as they do not belong to any proper class. Moreover, we cannot assume that foreign patterns are known at the stage of classifier construction. Thus, the main idea behind this approach is to develop a generic

foreign pattern rejection mechanism that will allow us to reject a broad variety of samples of different kind.

In this paper we introduce a novel approach for classification with foreign pattern rejection based on the formalism of cellular automata. The method is suited for image recognition. The phase of model construction is based entirely on native data, so no knowledge about foreign patterns is needed to build the classifier. This makes the model applicable to real-world scenarios, where the origin of any foreign pattern is typically unknown. The proposed method is tested in an empirical study of handwritten digits recognition. As foreign patterns we use distorted digits. It shall be mentioned that patterns come in various forms: images, sound or voice recording, etc. and there are two approaches to process them: either by using the original pattern format or by employing feature vectors (features are numbers describing original patterns). In our method we operate on images (original pattern format).

The paper is structured as follows. Section 2 presents the background knowledge on foreign elements detection. Section 3 presents the concept of cellular automata in the context of pattern recognition. Section 4 covers the proposed method for native image classification with foreign pattern rejection. Section 5 is devoted to experimental evaluation of the proposed model. Section 6 concludes the paper.

2 Literature Review

Popular mechanisms for pattern recognition reinforced with foreign pattern rejection process data in form of feature vectors. In this branch of studies we find two groups:

- one in which the training phase is based on synthesized foreign patterns, and
- methods that are based on native patterns only.

The first stream of research assumes that foreign patterns form separate class(es) and we shall be able to describe them in a way analogical to the way how we describe native patterns. In the training process we learn how to distinguish between foreign class(es) and native class(es). Among studies following this line of thought we find, for instance [2] and [4].

The second group of methods does not assume that foreign patterns form class(es). Therefore, these rejection mechanisms are trained on native patterns only. As an example of such a method we may give a centroid-based approach, whose goal is to locate a determined number of centroids in the data and using the notion of distance determine areas around centroids that should be occupied by native patterns only. If a pattern resides outside of the area close to a centroid, it gets rejected. Particular implementations of this approach are documented in [11] and [12]. Another well-known method for foreign element rejection based on native patterns only is the one-class SVM, also called ν-SVM, or novelty detection SVM. For a more in-depth elaboration on this method one may consult [13] and [15]. The one-class SVM detects soft boundaries of a set that

we pass on to its input. As a consequence, we are able to asses which elements surely belong to this class and which of them do not.

In this paper we propose an approach to foreign pattern rejection that is not based on features vectors, but on images. Filters are the most similar approach to the method presented in this paper. A typical application of filters is to suppress, reconstruct or enhance certain characteristics of an image. Standard filters reduce noise in an image (like mean filter, median filter, smoothing filters, speckle removal filters), normalize content of an image, detect and/or enhance edges in an image. For a comprehensive reference to image processing one may consult [3]. In the presented approach we apply sets of automata rules that in fact act as a collection of filters and they alter images. On top of that, we provide a simple additional decision algorithm that assigns class labels and rejects foreign patterns. Importantly, rules are developed based on training set of images what distinguished this method from "classical" filters.

The literature presents a few cellular automata-based approaches to pattern recognition, including methods reported in [8, 10]. On the other hand, formalism of cellular automata as concept allows for substantial implementation flexibility and therefore at the current development stage we see greater similarity of our approach to filters rather than to the methods described in the cited papers.

3 Cellular Automata

An **Automaton** is a self-operating machine or software algorithm which follows a determined sequence of operations in order to process some input data. Analogously, a **Cellular automaton** can be understood as a software algorithm which follows a determined set of **rules** that prescribe changes to the input data. Another definition of this concept can be found in *Cellular Automata for Pattern Recognition* [14]:

"Cellular Automata (CA) are spatiotemporal discrete systems[9] that can model dynamic complex systems[14]."

Simulation A simulation of such algorithm would take place in a grid, i.e. a big rectangle with cells in it, like a matrix. In there, cells would be constantly evaluated and their colors (formally called **states**) modified, and such change would be determined by the rules that were previously defined before running the simulation. A rule tells the automaton where to look for and what to do with a given cell if such rule is satisfied. From the automaton perspective, a cell has two important properties:

1. a **state** that is subject to changes at any given time, and
2. a **neighborhood** of surrounding cells, whose states are considered as important as the state of the cell itself.

Rules A rule r_i is a statement applied to a single cell, which consists of a condition and an action to be taken when such condition is satisfied, where i denotes the index of such rule within its corresponding rule set. Usually, the condition takes into consideration the current state of a given cell and the states of its neighbors. If satisfied, the target cell might potentially change its current state, depending on what the matched rule dictates. In the application, 8 different rules were designed to work with a 3×3 neighborhood and their goal is to detect any type of edge an image could have. The general purpose of each rule is to find live cells and, if the rule is matched, switch their states to dead. At the end of such operation, the results will yield clearly defined edges of live cells for any image, which will allow the program to compute a series of statistics based on this result and store them for the label generation stage. The patterns shown in Figure 1 represent the rules used by this program. In order to find a match, it is enough for a rule to find the exact combination of states at the desired positions. This means that it is not a problem if there are more live cells than needed in a neighborhood, as long as the ones that are required to be in that particular state fulfill that requirement. This approach has been named *Partial Pattern Match*.

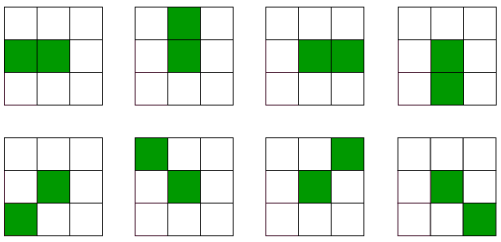

Fig. 1. Rules used by the program for finding edges in images.

4 Patterns Recognition Using Cellular Automata

4.1 Image Sets

The program works with different types of image sets, where each one plays its own role within the modeling and classification processes. These sets constitute the input to the cellular automaton algorithm.

Training Set A training set is a set of image samples that belong to C different classes and are used by the program for learning. Based on these samples, a classification model is created that is later used by the application for classifying different test images and verifying the quality of such model.

Validation Set A validation set is a set of image samples that belong to C different classes and are used for tuning up the parameters of a classifier. At the end of the training phase, a validation set is used in order to compare the performance of the algorithm and avoid over-fitting of its parameters. Such set of images is used for improving the classification model and assigning the appropriate scores to each class label. It is a middle-step between the actual training and the testing phases.

Test Set A test set (also called *hold-out set*) is a collection of sample images that are used only to assess the performance of a fully-trained classifier. A test set is used for evaluating the relationships that were discovered by the application and check if such relationships hold true.

4.2 The Algorithm

The goal for the cellular automaton is to take each subset of the input training set, analyze each image by applying the aforementioned rules and compute a series of statistics based on the cells that did not change with respect to the total number of dead cells that conform the actual shape in the image and were affected by a particular rule. This applies to every next image from the given collection. Each rule in the rule set is applied one time (if a given cell matches its criteria) and results of that single transition for each cell are recorded separately. Based on the collected information, decision rules are generated. These rules tell the system how to classify a test input image based on a range of the form:

$$min\{S_{C_i}\} \leq S_{r_i} \leq max\{S_{C_i}\}$$

where:

$min\{S_{C_i}\}$ - the lowest interval value for an image belonging to a class C_i, where i denotes the index of the given class within the set of all classes

$min\{S_{C_i}\}$ - the highest interval value for an image belonging to a class C_i, where i denotes the index of the given class within the set of all classes

S_{r_i} - the actual statistic computed for the given test image after applying a rule r_i

Such set of decision rules needs to be validated and eventually refined before it is used on test sets.
In order to avoid potentially imprecise image classifiers, a validation set is required for adjusting the already defined classification model and verifying its utility. The validation set consists of a collection of images that need to be classified by the automaton without knowing to which class they originally belong. The procedure for such classification is the same as in the **Automaton** subsection, with the difference that statistics for these unknown images are not stored anymore, instead, they are used for assigning labels to those images based on the already generated decision rules.
The steps for the classification model validation are as follows (for every image in the validation set):

Part 1:

1. Apply a rule to an image
2. Count how many cells remained in live state after applying the rule
3. Compute a statistic S_j of the number of cells that stayed alive and the type of rule that led to such result for a given image, where j corresponds to the index of the given statistic within the set of all statistics for a given class.
4. Use the statistic S_j to assign a class label L_C to the image based on the decision rules
5. Store information about the assignment in a table
6. Repeat the same procedure for every next image in the current subset

Part 2:

7. Check if the image was assigned the correct class label C_{L_i}, where i denotes the index of the given label from the set of all labels that were generated for the given class
8. Count how many classifications where correct for a given rule set R_i, where i denotes the index of the rule set from the set of all rule sets
9. Store the score S_{R_i} of the result in a pair $\{R_i, S_{R_i}\}$
10. Repeat the same procedure for every next image in the current subset

At the end, select a pair $\{R_i, S_{R_i}\}$ such that

$$S_{R_i} := \max\{S_{R_i}\}$$

Such rule set will be then defined as the best one for classifying images of a given class C_i. The system is now ready for analyzing test sets.

4.3 Testing the classification model

Once the classification model has been built, the next step is to test its efficiency by applying it to different test image sets. Classification of images is done following the steps below (for every image in the test set):

Part 1:

1. Apply a classifier to a test image
2. Count how many cells remained in live state after applying the classifier rule
3. Compare the number of cells that stayed alive and the type of rule that led to such result against the intervals that were defined by the label of the given classifier
4. If all the edge types match the given intervals, assign the classifier label
5. Store information about the assignment in a statistic S_j
6. Repeat the same procedure for every next image in the current subset

Part 2:

Once the classification procedure is over, use the statistics to build a confusion table and present it as a summary of all the tests performed on a given test set.

5 Experiments

5.1 Experimental Settings

Image Preprocessing Before running a training or testing routine, the program must ensure normalization of any image in a training/validation/test set. That is, each image must be properly scaled and converted to black and white. Preprocessing ensures comparability of different images in the imported sets, which in turn assures correctness of results. In general, it is assumed that every set will contain already normalized images, i.e. scaled, gray-scaled and cleared up. Nevertheless, a preventive local normalization shall be performed for any case. The aim of scaling is to resize each image to a standard, predefined size such that it has the same dimensions as the rest. Such resizing helps setting one common size to all the images in a set, i.e. the matrix size of each is the same. The standard dimensions used by the algorithm are 28×28 pixels, a size which does not implicate any quality loses in images of simple shape on a white background – such as handwritten digits – and helps the application to improve the image processing speed. Gray-scaling is applied in order to work with black and white images, whose shapes are better recognizable and their edges clearly defined.

Image Labeling Image labeling is the process of deciding to which class an unknown image belongs. In order to take such decision, the program needs to apply the classification model it created based on the training and validation sets that were provided by the user. The final results will depend on the quality and quantity of the input data. Each predefined rule used by the application has its own, unique type. The type attribute helps the program detecting edges in images much easier, thus preventing potentially erroneous label assignments.

5.2 Data Sets

The experiment was based on a native dataset of handwritten digits taken from the MNIST database [7]. Some samples are illustrated in Figure 2. The MNIST database holds approximately $1,000$ images of each digit, which makes a 10-class problem, leaving us with 10 unique class labels: $0, 1, \ldots, 9$. In order to train the classifier we took 200 samples of each native class for the training set, 200 samples of each native class for the validation set, and 300 samples of each class for the testing set. Naturally, training, validation, and testing sets of native patterns are disjoint.

For rejection mechanism testing purposes, we used datasets of distorted digits. Samples of foreign patterns are displayed in Figure 3. Distorted digits were formed based on original MNIST images, from which we randomly selected 300 samples of each digit and applied various modifications. In total, foreign datasets were made of $3,000$ samples (analogously to the test set of native patterns).

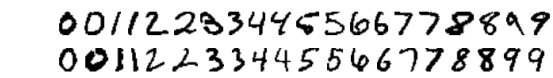

Fig. 2. Samples of native patterns (handwritten digits) from the MNIST database.

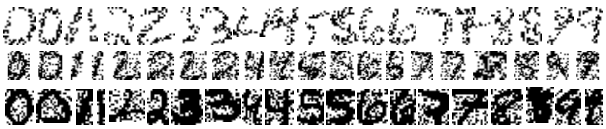

Fig. 3. Samples of foreign patterns – digits distorted with salt and pepper noise. From top to bottom: samples with few black pixels removed, samples with few black pixels removed and few black pixels added, samples with black pixels added.

5.3 Quality Evaluation

The case of pattern recognition with rejection requires dedicated quality measures. The following – very straightforward – notions are used:

- CC (Correctly Classified) – the number of correctly classified patterns, i.e. native patterns classified as native with correct class label,
- TP (True Positives) – the number of native patterns classified as native (no matter, into which native class),
- FN (False Negatives) – the number of native patterns incorrectly recognized as foreign,
- FP (False Positives) – the number of foreign patterns incorrectly recognized as native,
- TN (True Negatives) – the number of foreign patterns correctly recognized as foreign.

Quantities TP, FN, FP, and TN are widely used in different literature in the context of binary pattern recognition. This set of quantities is supplemented with another quantity: *Correctly Classified*, which allows a more versatile quality evaluation.

Based on the notions above: CC, TP, FN, FP and TN, we construct the following measures:

Table 1. Quality measures for classification with rejection.

$$\text{Native Precision} = \frac{\text{TP}}{\text{TP+FP}} \qquad \text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+FN+FP+TN}}$$

$$\text{Foreign Precision} = \frac{\text{TN}}{\text{TN+FN}} \qquad \text{Strict Accuracy} = \frac{\text{CC+TN}}{\text{TP+FN+FP+TN}}$$

$$\text{Native Sensitivity} = \frac{\text{TP}}{\text{TP+FN}} \qquad \text{Fine Accuracy} = \frac{\text{CC}}{\text{TP}}$$

$$\text{Foreign Sensitivity} = \frac{\text{TN}}{\text{TN+FP}} \qquad \text{Strict Native Sensitivity} = \frac{\text{CC}}{\text{TP+FN}}$$

$$\text{F–measure} = 2 \cdot \frac{\text{Precision} \cdot \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}}$$

5.4 Results

This subsection presents results of classification procedures performed on the different test sets mentioned above. Such results allow us comparing outcomes when the classification model is built, based on pattern (image) sets. We apply measures discussed in Section 5.3. Let us recall factors taken into account:

- Number of classes (Fixed: 10)
- Number of images in a test set (Fixed: 300 per class)
- Number of images in a training set (Fixed: 200 per class)
- Number of images in a validation set (Fixed: 200 per class)

Figure 4 illustrates quality measures computed from the classification with rejection procedures that were run against native handwritten digit datasets and foreign datasets of distorted digits.

In first place, let us look into the quality of native pattern acceptance. This could be evaluated by the Native Sensitivity, Native Precision, and Native F-measure. Native Sensitivity measures how many native patterns were accepted, and no knowledge about foreign pattern rejection is needed to compute it. Therefore, in Figure 4, bars for this measure stay exactly the same for all the three foreign sets we have tested. The proposed method accepts native patterns extremely well: Native Sensitivity is on the level of 99.53%. Native Precision expresses a balance between accepted native patterns and all accepted patterns. The poorest Native Precision was achieved on the foreign set of distorted digits with few black pixels randomly removed. In this case, the Native Precision is equal 83.74%. The other two sets were rejected with much higher success rate. Native Precision for distorted images with randomly added black pixels and with randomly added and removed black pixels was equal to 99.27% and 98.06% respectively. The Native F-measure expresses a proportion between Native Sensitivity and Native Precision, and it joins those to measures together into a single factor. Therefore, it is very high for the set with black pixels added and with black pixels removed and for the set with black pixels added. It is lower

for the set with black pixels removed, which is a consequence of a lower Native Sensitivity achieved for this dataset.

Strict Native Sensitivity measures the classification rate achieved by the classification/rejection model. It stays the same for all the foreign sets, as it only measures classification quality. The proposed mechanism works very well – it does not reject native patterns, and furthermore, it assigns correct class labels. The Strict Native Sensitivity is equal to 97.80%.

The Fine Accuracy parameter measures how – by adding a rejection mechanism – we can improve classification rates. Such rejection mechanism could be useful when rejecting native patterns that would otherwise be misclassified if such rejection procedure didn't exist. The Fine Accuracy in our experiment was indeed higher than the Strict Native Sensitivity, which confirms that the rejection model rejects "difficult" native patterns that are hard to classify correctly.

Accuracy and Strict Accuracy express a balance between native pattern acceptance and foreign pattern rejection. In the case of the foreign dataset of digits that were distorted by removing few black pixels, for which the rejection was less successful, those two measures are relatively low: oscillating around 90%, whereas for the other two foreign datasets they are much higher (around $98\% - 99\%$).

Foreign Precision, Foreign Sensitivity, and Foreign F–measure evaluate the rejection quality achieved for tested foreign datasets. The Foreign Sensitivity and Foreign F–measures revealed clearly that the proposed mechanism works better if new, unexpected black pixels are disturbing recognized images. Let us recall that the studied distortions come from three variants of salt and pepper noise. Removing black pixels turned out to be more confusing for the automata. The Foreign F–measure was equal to 90.95% for the set with removed black pixels, 98.03% for the set with white pixels added and black pixels removed, and 99.40% for the foreign set with black pixels added.

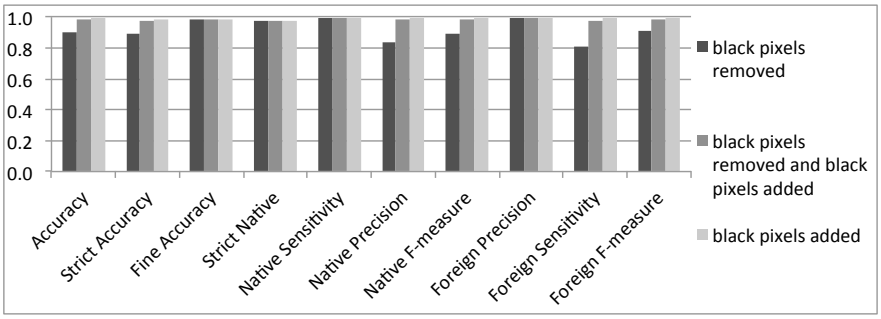

Fig. 4. Quality measures for classification with rejection for the native dataset of handwritten digits and foreign datasets of distorted handwritten digits.

6 Conclusion

In the paper we presented a novel approach to pattern recognition: classification of native patterns with foreign pattern rejection based on cellular automata. The proposed method was experimentally evaluated in a case study of handwritten digit recognition. As foreign patterns we used three datasets of distorted digits that were modified by adding different salt and pepper noise effects. Experimental results showed that the method, despite of its relative simplicity, has a very high success rate at native pattern classification and foreign pattern rejection.

If we compare our previous studies on feature-vector-based geometrical approaches to foreign pattern rejection reported in [1], we can say that the automata-based method introduced in this paper outperforms the geometrical model as it provides a better balance between native pattern acceptance and foreign pattern rejection. However, in order to popularize the proposed approach we need to extend the scope of experiments onto more challenging datasets, for instance for printed musical notation as reported in [5] to investigate how the proposed method works for imbalanced data.

In the future, we plan to take a full benefit of the cellular automaton formalism. The implemented and tested processing is so far very simple, because it is based on single automata transitions, making the implemented model more like a collection of filters than cellular automata itself. We believe that there is more to be achieved with the fully-developed cellular automata-based model, and that motivates us to continue further development of our project. Relative simplicity of the current implementation will become much more complicated in the process, but the model will get far more interesting from the theoretical point of view. An alternative path is to apply a different kind of automata instead, like for instance bipolar automata, in which processing is based on operations defined in [6].

Acknowledgements

The research is supported by the National Science Center, grant No 2012/07/B/ST6/01501, decision no DEC-2012/07/B/ST6/01501.

References

1. Ciecierski, J., Dybisz, B., Jastrzebska, A., and Pedrycz, W., *A Geometrical Approach to Rejecting Option in Pattern Recognition Problem*, in: Proc. of CISIM 2015, LNCS 9339, pp. 231-243, 2015.
2. Desir, C., Bernard, S., Petitjean, C., and Laurent, H., *One class random forests*, Pattern Recognition 46, pp. 3490-3506, 2013.
3. Gonzalez, R. C. and Woods, R. E., *Digital Image Processing*, Prentice Hall, 2008.
4. Hempstalk, K., Frank, E., and Witten, I., *One-class classification by combining density and class probability estimation*, Machine Learning and Knowl. Disc. in Databases, pp. 505-519, 2008.

5. Homenda, W., *Optical music recognition: The case study of pattern recognition*, Eds.: Kurzynski, M.; Puchala, E.; Wozniak, M.; et al., Proceedings of CORES 2005, Advances in Soft Computing, pp. 835-842, 2005.
6. Homenda, W. and Pedrycz, W., *Processing Uncertain Information in The Linear Space of Fuzzy Sets*, Fuzzy Sets And Systems, 44(2), pp. 187-198, 1991.
7. LeCun, Y., Cortes, C., and Burges, C., *The MNIST database of handwritten digits*, in: http://yann.lecun.com/exdb/mnist.
8. Maji, P., Ganguly, N., Saha, S., Roy, A. K., and Chaudhuri, P. P., *Cellular Automata Machine for Pattern Recognition*, in: Proc. of ACRI 2002, LNCS 2493, pp. 270-281, 2002.
9. Von Neumann, J., *Theory of Self-Reproducing Automata*. University of Illinois Press, 1966.
10. Raghavan, R., *Cellular automata in pattern recognition*, Information Sciences 70 (1-2), pp. 145-177, 1993.
11. Shin, K., Abraham, A., and Han, S., *Enhanced Centroid-Based Classification Technique by Filtering Outliers*, in: LNAI 4188, pp. 159-163, 2006.
12. Takci, H. and Gungor, T., *A high performance centroid-based classification approach for language identification*, in: Pattern Recognition Letters 33, pp. 2077-2084, 2012.
13. Vapnik, V., *The Nature of Statistical Learning Theory*. SpringerVerlag, New York, Inc., 1995.
14. Wongthanavasu, S. and Ponkaew, J., *Cellular Automata for Pattern Recognition, Emerging Applications of Cellular Automata*, Dr Alejandro Salcido (Ed.), InTech, 2013.
15. Ypma, A. and Duin, R., *Support objects for domain approximation*, in: Proc. of ICANN'98, 1998.