CNN training with graph-based sample preselection: application to handwritten character recognition

Frédéric Rayar and Seiichi Uchida
Department of Advanced Information Technology
Kyushu University, Fukuoka, Japan
{*rayar, uchida*}*@human.ait.kyushu-u.ac.jp*

Masanori Goto
Research & Development Center
GLORY LTD., Hyogo, Japan
gotou.masanori@mail.glory.co.jp

***Abstract*—In this paper, we present a study on sample preselection in large training data set for CNN-based classification. To do so, we structure the input data set in a network representation, namely the Relative Neighbourhood Graph, and then extract some vectors of interest. The proposed preselection method is evaluated in the context of handwritten character recognition, by using two data sets, up to several hundred thousands of images. It is shown that the graph-based preselection can reduce the training data set without degrading the recognition accuracy of a non pre-trained CNN shallow model.**

***Keywords*-Convolutional neural network; Relative neighbourhood graph; Handwritten character recognition; Large data set; Training data set preselection**

I. Introduction

The advent of the so-called *Deep Learning* in the last decade has led to major advances in several research fields such as artificial intelligence, pattern recognition, computer vision, and natural language processing. The document image analysis and recognition community has also embraced these neural network approaches for various tasks such as binarisation, layout analysis, character recognition, script identification, and word spotting. The number of paper that use such approaches is increasing rapidly, as illustrated by the recent ICDAR 2017 event[1]: 40 out of 52 oral papers in the main conference used neural approaches. Indeed, results obtained using these approaches are often impressive and almost always outperform state-of-the-art methods.

Among the existing architectures, Convolutional Neural Networks (CNN) have become a subject undergoing intense study in the past few years. The usage of CNN falls into supervised machine learning, *i.e.* one needs to gather a training data set and use this set to train a model, model that will then be used to predict various outputs. One specificity of using a neural approach is that one needs to have a huge amount of training data to perform well. Recently, the authors of [1] showed that increasing the size of the training data set allows to achieve a near-perfect recognition performance on handwritten digits. They have achieved up to 99.99% of correct recognition using several hundred thousand training samples.

In this paper, we aim at studying the influence of a preselection step on the training data set, with regards to the recognition accuracy of a CNN-based classification. More precisely, we wonder if all the images of a large training data set are relevant in the training process. In what extent the underlying redundancy of such large training data set helps (or does not) during the training of a CNN model? Hence, we propose a graph-based preselection that removes samples lying around each class *"center"*, without any severe degradation of the recognition performance.

The contributions of this paper are as follows:

1) We propose a method for preselecting training images by analysing the data distribution. To do so, we structure the data in a network representation, namely the Relative Neighbourhood Graph (RNG) [2]. Candidates are then extracted from this graph in order to be used in the CNN training process.
2) We show through experimentations on two data sets, MNIST and HW_O-RID (respectively 60,000 and 740,438 training images), that the proposed preselection strategy reduce the training data set up to 76% without degrading the recognition accuracy.

The rest of the paper is organised as follows: Section II briefly presents the related works on data preselection in the literature. Section III details the proposed preselection strategy using the RNG. In Section IV, we present the performed experiments to evaluate the relevance of our work and discuss the results in Section V. Finally, we conclude our study in Section VI and outline directions for future research.

II. Related Work

Training a classification model is often performed on large training data sets, to avoid overfitting and enhance the generalisation performance of the model. However, as mentioned in [3], several reasons can support the need of reducing the training set: (i) reducing the noise, (ii) reducing storage and memory requirement and (iii) reducing the computation time of the training or the prediction phase.

Many sample (or prototype) selection solutions have been proposed in the past to address training set reduction. We can categorise these solutions in three families:

1) The *"editing"* paradigm, that aims at eliminating erroneous instances and remove possible overlapping between classes. Hence, such algorithms behave as

[1] http://u-pat.org/ICDAR2017/program_main.php

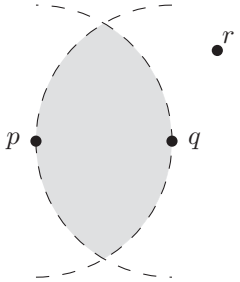

Figure 1. Relative neighbourhood (grey area) of two points $p, q \in \mathbb{R}^2$. If no other point of D lays in this neighbourhood, then p and q are relative neighbours.

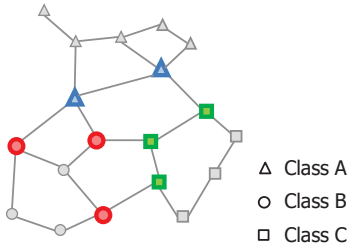

Figure 2. Illustration of bridge vectors on a toy data set. The bridge vectors are highlighted with colours and thicker borders.

noise filters. For instance, the edited nearest neighbour [4] algorithm removes an instance if its class is inconsistent with its neighbours' majority class.

2) The *"condensing"* paradigm, that aims at finding instances that will allow to perform as well as a Nearest Neighbour (NN) classifier that uses the whole training set. For instance, the condensed nearest neighbour rule [5], removes instances from the training set one by one if their absence do not degrade the classification accuracy. However, as mentioned in [3], such techniques are *"very fragile in respect to noise and the order of presentation"*.
3) The *"hybrid"* (editing-condensing) paradigm, that aims at removing noise and redundant instances at the same time.

Among the proposed techniques, that falls into the aforementioned categories, it is worth mentioning the usage of: (i) random selection techniques [6], (ii) clustering techniques [7] or graph-based techniques [8]. One can refer to thorough surveys that have been done recently by Garcia et al. [9] in 2012 (for NN based classification), and by Jung et al. [10] in 2014 for (Support Vector Machine (SVM) [11] based classification).

As one can deduce by the existence of the aforementioned surveys, prototype selection has been widely studied for the NN-based classification and SVMs, but to the best of our knowledge, no similar studies, has not been performed for CNN (or more generally neural networks). Indeed, most of the studies that use CNN are usually focused on the acquirement of large training data sets, using crowdsourcing (e.g. ImageNet [12]), synthetic data generation or data augmentation [13] techniques.

In this study, we have chosen a graph-based condensing sample preselection strategy. Indeed, an extensive series of work have been performed on the relevance of proximity graphs [14] to preselect samples in classification training set. More specifically, we have selected the RNG that has been recently proven a good fit to preselect high-dimensional samples [15] in large training data sets [16].

III. Preselection using the Relative Neighbourhood Graph

A. Relative Neighbourhood Graph

The relative neighbourhood graph has been introduced by G. Toussaint in the early 1980s [2]. The construction of this graph is based on the notion of *"relatively closeness"*, that defines two points as relative neighbours if "*they are at least as close to each other as they are to any other points*". From this definition, one can define $\mathrm{RNG} = (V, E)$ as the graph built from a given data set$D$, where distinct points p and q of D are connected by an edge $\overline{pq}$ if and only if they are relative neighbours. Thus,

$$E(\mathrm{RNG}) = \{\overline{pq} \mid p, q \in D, p \neq q,$$
$$\delta(p, q) \leq \max(\delta(p, r), \delta(q, r)), \forall r \in D \backslash \{p, q\}\}.$$

where $\delta : D \times D \rightarrow \mathbb{R}$ is a distance function. An illustration of the relative neighbourhood of two points $p, q \in \mathbb{R}^2$ is given in Figure 1.

The RNG has been used in several works in various fields such as computer vision, geographic analysis or pattern classification [17]. Its main benefit is that it is a connected graph, that highlights the topology of the data and embeds local information about vertex neighbourhood. The main drawback of the RNG is its construction complexity in $O(|D|^3)$. However, recent works such as [18] and [16] have proposed solutions to build RNG of large data sets, up to millions data points.

B. Bridge Vectors

Bridge vectors, as defined in [16], are points that lay in the outer frontiers of classes. Using the RNG representation of the data set, they correspond to points that have at least one relative neighbour from a different class. Figure 2 illustrates the notion of bridge vectors on a toy data set.

In our experiments, to get the bridge vectors of a given training data set, the following straightforward steps are performed: (i) build the RNG of the training data set using the algorithm proposed in [16] and (ii) if two points of different classes are connected by an edge in the computed graph, they are added in the bridge vectors' list.

C. Study workflow

Figure 3 illustrates the workflow of the proposed study. The main goal is to evaluate the relevance of the proposed

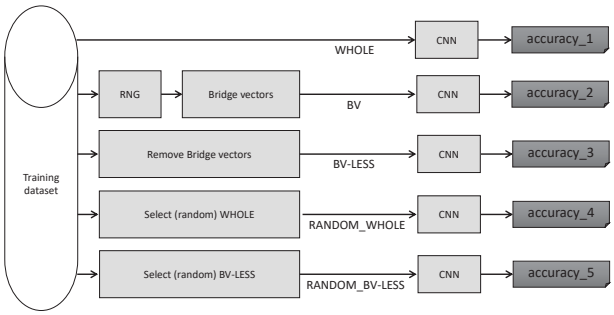

Figure 3. General workflow of the study. The CNN model is trained using either: (1) WHOLE, (2) BV, (3) BV-LESS, (4) RANDOM_WHOLE or (5) RANDOM_BV-LESS, as defined in Section III-C.

preselection technique, and highlight the importance of bridge vectors in the training of a CNN classifier.

To do so, five different training subsets have been used for a given data set:

- WHOLE: the whole training data set,
- BV: only the extracted bridge vectors of the RNG build from WHOLE,
- BV-LESS: the data set WHOLE, but without bridge vectors BV,
- RANDOM_WHOLE: a random subset of WHOLE, with approximatively the same size as BV,
- RANDOM_BV-LESS: a random subset of BV-LESS, with approximatively the same size as BV.

IV. Experimental setup

A. Data sets

We applied the proposed training data set preselection to two isolated handwritten digit data sets, namely MNIST and HW_R-OID. Table I shows the size of each subset, for the two data sets that have been used in this work. First, the MNIST [19] data set, that corresponds to binary 28×28 images of centered handwritten digits. Ground truth (*i.e.* correct class label (*"0"*, ..., *"9"*), is provided for each image. In our experiments, we have used 60,000 images in the training data set and 10,000 for testing purpose.

Second, the HW_R-OID data set is an original data set from [1]. It contains 822,714 images collected from forms written by multiple people. The images are 32×32 binary images of isolated digits and ground-truth is also available. In this data set, the number of the samples of each class is different but almost the same (between 65,000 and 85,000 samples per class, except the class *"0"* that has slightly more than 187,000 samples). In our experiments, we have split the data set in train/test subsets with a 90/10 ratio (740,438 training + 82,276 test images). To do so, 90% of each class samples have been gathered to build the training subset.

For the these two data sets, the intensities of the raw pixels have been used to described the images, and the Euclidean distance has been used to compute the similarity between pairs of images.

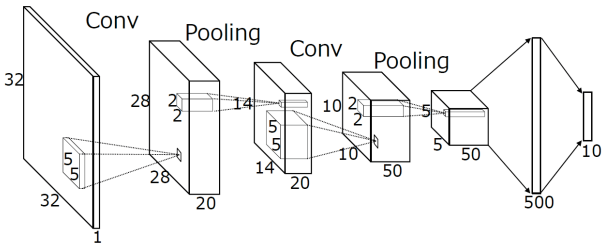

Figure 4. Modified LeNet-5 CNN architecture used in our experiments. No pretraining has been done. Inputs images are 28×28 and 32×32 for the MNIST and HW_R-OID data sets respectively.

B. CNN classification

Experiments were done on a computer with a i7-6850K CPU @3.60GHz, with 64.0GB of RAM (not all of it was used during runtime), and a NVIDIA GeForce GTX 1080 GPU. Our CNN classification implementation relies on the usage of Python (3.6.2) along with the Keras library (2.0.6) and a TensorFlow (1.3.0) backend.

The same CNN structure and parameters as [1] have been used. A simple CNN architecture, namely modified LeNet-5. The main difference with the original LeNet-5 [19] is the usage of ReLU and max-pooling functions for the CONV layers. Figure 4 illustrates the architecture that has been used. As mentioned in [1], it is *"a rather shallow CNN compared to the recent CNNs. However, it still performed with an almost perfect recognition accuracy"* (when trained with a large data set). No pre-initialisation of the weights is done, and the CNN is trained with classical back-propagation for 10 epochs.

During our experimentations, both computation times and recognition accuracies have been measured for further analysis. For each training data sets, experiments were run 5 times to compute an average value of the aforementioned metrics.

V. Results

A. Analysis of the preselection efficiency

Table I presents the average accuracies obtained for all the training data sets introduced in Section III-C for both MNIST and HW_R-OID. The major observation is that the average accuracy obtained by training the CNN with only the bridge vectors (BV) is almost the same as the one obtained by training the CNN with the whole available training data set (WHOLE). This supports the efficiency of the proposed preselection strategy.

Figure 6 (right) illustrates all the misclassified digits sorted, using either WHOLE or BV as training data set for HW_R-OID. One can easily spot some rather difficult instances of handwritten digits, even for a human being. However, the CNN still fails to classify some instances that are "easy" for a human (*e.g.* the last instances of *"9"* using only the bridge vectors).

By removing the bridge vectors from the training data set, the CNN recognition performance is lower, while the cardinality of the BV-LESS training data set remains

Table I
CNN-BASED CLASSIFICATION RESULTS: (I) SIZE OF THE TRAINING DATA SET, (II) AVERAGE RECOGNITION ACCURACY AND (III) AVERAGE TRAINING TIME (IN SECONDS) ARE PRESENTED.

	Training data set	WHOLE	BV	BV-LESS	RANDOM WHOLE	RANDOM BV-LESS
MNIST	# training data	60,000	22,257	37,743	22,260	22,258
	accuracy (%)	98.79	98.78	97.05	98.22	96.47
	training time (s)	252	104	164	104	103
HW_R-OID	# training data	740,438	173,808	566,630	174,002	174,012
	accuracy (%)	99.9343	99.9314	99.7372	99.8586	99.5631
	training time (s)	4171	1086	3252	1086	1089

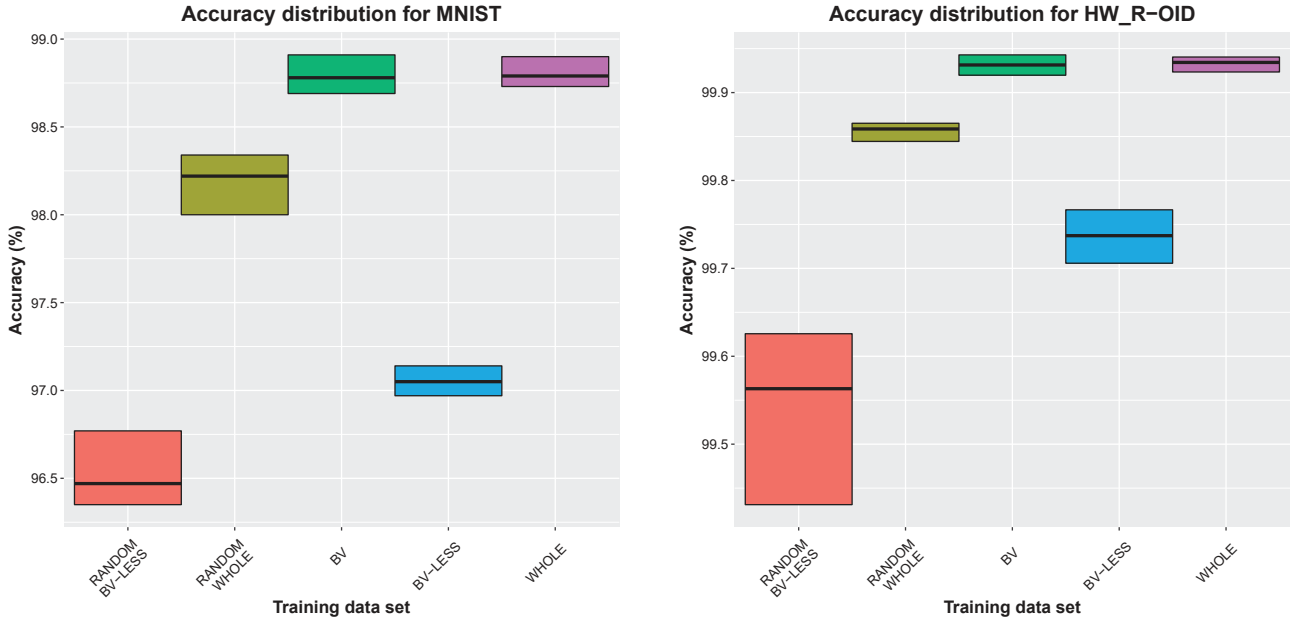

Figure 5. Accuracy distribution on MNIST (left) and HW_R-OID (right) data sets over the different training sets (presented in ascending cardinality).

greater than the bridge vector subset (BV). Hence, the size of the training data set is not the only criterion that should be considered while training a CNN.

To emphasize on the relevance of the proposed preselected candidates, we also have randomly selected samples from WHOLE and BV-LESS data sets (to produce RANDOM_WHOLE and RANDOM_BV-LESS respectively). The number of samples has been selected close to the the cardinality of the number of bridge vectors. The RANDOM_WHOLE training data set allows the CNN to perform better than both the BV-LESS and the RANDOM_BV-LESS. Hence, the sole presence of bridge vectors in the training data set seems to allow the CNN to achieves better recognition accuracy.

To avoid basing our conclusions only on average accuracies, we also present the accuracy distribution obtained for each training data sets (in ascending cardinality) over the 5 runs in Figure 5. It clearly helps us to assert the following statement: in terms of recognition accuracy, we have:

$$\text{WHOLE} \approx \text{BV} > \text{RANDOM\_WHOLE} > \ldots$$
$$\ldots \text{ BV-LESS} > \text{RANDOM\_BV-LESS}$$

Hence, we validate our hypothesis that not all the training images are mandatory to achieve high recognition accuracy. Some redundancy in the training data set can be pruned using the proposed preselection strategy, while allowing the CNN to achieve the same near-perfect recognition accuracies.

B. Analysis of the method overhead

Table II presents the computation times for the network representation of the training data set, using the RNG construction algorithm of [16]. Since the algorithm takes the images as input, we also provide the data loading time. For the smallest data set, only 5 minutes are needed to generate the RNG. For the larger data set, this computation time increases: about 17 hours. Hence, the computation of the RNG may be a limitation, even if this could be addressed by different solutions (*e.g.* the optimisation of the code or the usage of a GPU).

Nonetheless, since the computation of the RNG, for a given training data set, is performed only once, the ratio of this overhead can be minimised in a general CNN training framework, where one needs to make several number of trial-and-error iterations regarding the choice of the neural

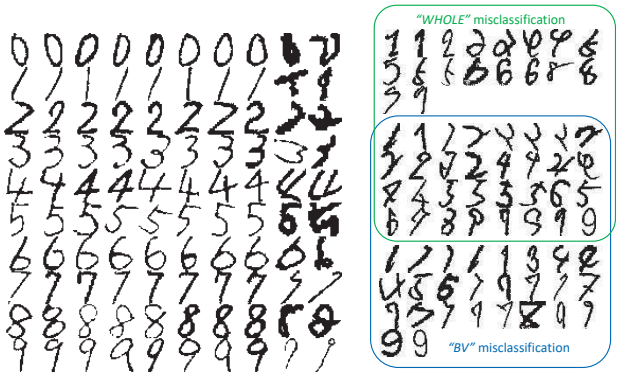

Figure 6. HW_R-OID bridge vector samples (left) and misclassified digits using either WHOLE or BV as training data set (right).

network structure and parameters.

Table I presents the training subsets' size and the average training times of our experiments. Regarding the CNN training, thanks to the proposed preselection technique, we can decrease the training data set by 76% and 63%, while having a training speed-up ratio of 3.8 and 2.4 for the MNIST and HW_R-OID data sets respectively.

C. Analysis of bridge vectors

Figure 6 (left) illustrates some bridge vectors, extracted from HW_R-OID whole training data set. One can see that while some ambiguous instances are present (last two columns, manually selected), the major part of the bridge vectors are *"standard"* patterns, that allow the CNN to recognize such patterns well during the recognition step.

Table III presents the quantitative results of the bridge vectors analysis for the two studied data sets. In the MNIST data set, the classes in the WHOLE training set are well balanced. One can note that class *"1"* is the class that had the most number of images in the training data set. However, it is the class that has the smallest number of bridge vectors (less than 4%). A similar observation can also be made for the HW_R-OID data set: indeed, class *"0"* has at least two times more samples than the other classes. However, only 8.15% of class *"0"* samples are bridge vectors, which represent only 7.90% of all the bridge vectors. Hence, one can deduce that for a given class, if most of the samples are very similar, they do not make a huge contribution during the CNN model training.

A second observation that can be made is that for both data sets, certain classes have more bridge vectors (more than 10% of the BV subset): {*"3"*,*"4"*,*"8"*,*"9"*} and {*"8"*,*"9"*} for MNIST and HW_R-OID respectively. This observation is consistent with the similarity of such handwritten digits observed in the data sets.

To visualise the separability of the bridge vectors, the t-distributed stochastic neighbour embedding (t-SNE) [20] algorithm has been used. Several values of perplexity have been used, within the interval $[5, 50]$ as suggested in [20], and we observed the same type of results.

Figure 7 presents the t-SNE visualisations, using $perplexity = 30.0$, of the bridge vectors for the two studied data sets. For MNIST, classes *"4"* (in turquoise) and *"9"* (in red) are clearly not separated. For the second data set, one can note that classes *"8"* (in orange) and *"9"* (in red) are well-separated but overlap with several other classes.

Table II
IMAGE LOADING TIME AND RNG COMPUTATION TIME (IN SECONDS).

	# training images	image load loading (s)	RNG computation computation (s)
MNIST	60,000	133	304
HW_R-OID	740,438	1397	61,270

VI. CONCLUSION

We have provided in this study some insights about the relevance of the training data that is fed to train a CNN model. Indeed, even if a huge quantity of training samples helps to achieve almost near-perfect recognition, its underlying redundancy does not have a huge impact on the model's training. The proposed graph-based preselection method allows to reduce the training data set considerably, and thus accelerates the training computation time without deteriorating the recognition accuracy.

Future works will be done towards the following directions: (i) address the limitation related to the RNG computation time, (ii) perform experimentations using data sets other than isolated handwritten characters and (iii) start a formal study on the existence of *"support vectors"* for CNN.

ACKNOWLEDGEMENT

This research was partially supported by MEXT-Japan (Grant No. 17H06100).

REFERENCES

[1] S. Uchida, S. Ide, B. K. Iwana, and A. Zhu, "A further step to perfect accuracy by training CNN with larger data," in *15th International Conference on Frontiers in Handwriting Recognition, ICFHR*, 2016, pp. 405–410.

[2] G. T. Toussaint, "The relative neighbourhood graph of a finite planar set," *Pattern Recognition*, vol. 12, pp. 261–268, 1980.

[3] N. Jankowski and M. Grochowski, *Comparison of Instances Seletion Algorithms I. Algorithms Survey*, 2004, pp. 598–603.

[4] D. L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-2, pp. 408–421, 1972.

[5] P. Hart, "The condensed nearest neighbor rule (corresp.)," *IEEE Transactions on Information Theory*, vol. 14, no. 3, pp. 515–516, 1968.

[6] Y. J. Lee and S. Y. Huang, "Reduced support vector machines: A statistical theory," *IEEE Transactions on Neural Networks*, vol. 18, no. 1, pp. 1–13, 2007.

[7] Q.-A. Tran, Q.-L. Zhang, and X. Li, "Reduce the number of support vectors by using clustering techniques," in *Proceedings of the 2003 International Conference on Machine Learning and Cybernetics*, vol. 2, 2003, pp. 1245–1248.

Table III

FOR A GIVEN (WHILE) TRAINING DATA SET: (I) THE FIRST TWO ROWS CORRESPOND TO THE NUMBER OF DATA PER DIGIT AND THE RATIO OVER THE TRAINING SET (IN); (II) THE NEXT ROWS CORRESPOND TO THE NUMBER OF BRIDGE VECTORS PER DIGIT, THE RATIO OVER THE CLASS ELEMENTS AND THE RATIO OVER THE WHOLE BRIDGE VECTOR SET. PERCENTAGE VALUES ARE ROUNDED OFF TO TWO DECIMAL PLACES.

	class	0	1	2	3	4	5	6	7	8	9
MNIST	# data	5923	6742	5958	6131	5842	5421	5918	6265	5851	5949
	class ()	9.87	11.24	9.93	10.22	9.74	9.03	9.87	10.44	9.75	9.91
	# BV	1518	828	2113	2917	2498	2794	1375	2021	2864	3329
	class (%)	25.6	12.3	35.46	47.58	42.76	51.54	23.23	32.26	48.95	55.96
	BV (%)	6.82	3.72	9.49	13.10	11.22	12.55	6.18	9.08	12.87	14.95
HW_R-OID	# data	168,521	58,202	76,987	65,848	60,030	60,075	58,172	56,723	73,176	62,704
	class (%)	22.76	7.86	10.39	8.89	8.10	8.11	7.85	7.66	9.88	8.47
	# BV	13,728	9,204	10,581	19,682	16,054	17,233	12,636	19,983	28,241	26,466
	class (%)	8.15	15.81	13.74	29.89	26.74	28.68	21.72	35.23	38.59	42.20
	BV (%)	7.90	5.29	6.08	11.32	9.24	9.91	7.27	11.49	16.25	15.23

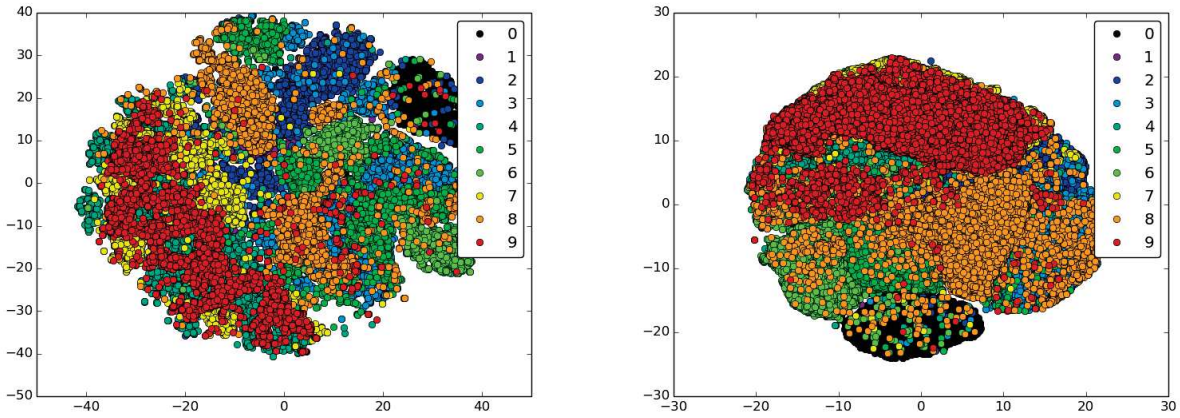

Figure 7. t-SNE visualisation of the bridge vectors of MNIST (left) and HW_R-OID (right) data sets (using perplexity $= 30.0$).

[8] B. K. B. Godfried T. Toussaint and R. S. Poulsen, "The application of voronoi diagrams to non-parametric decision rules," in *Computer Science and Statistics. North-Holland, Amsterdam*, 1985, pp. 97–108.

[9] S. Garcia, J. Derrac, J. Cano, and F. Herrera, "Prototype selection for nearest neighbor classification: Taxonomy and empirical study," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 417–435, 2012.

[10] H. G. Jung and G. Kim, "Support vector number reduction: Survey and experimental evaluations," *IEEE Trans. Intelligent Transportation Systems*, vol. 15, no. 2, pp. 463–476, 2014.

[11] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[12] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[13] S. C. Wong, A. Gatt, V. Stamatescu, and M. D. McDonnell, "Understanding data augmentation for classification: When to warp?" in *Int. Conf. on Digital Image Computing: Techniques and Applications (DICTA)*, 2016, pp. 1–6.

[14] G. T. Toussaint, "Some unsolved problems on proximity graphs," 1991.

[15] G. T. Toussaint and C. Berzan, *Proximity-Graph Instance-Based Learning, Support Vector Machines, and High Dimensionality: An Empirical Comparison*. Springer Berlin Heidelberg, 2012, pp. 222–236.

[16] M. Goto, R. Ishida, and S. Uchida, "Preselection of support vector candidates by relative neighborhood graph for large-scale character recognition," in *13th International Conference on Document Analysis and Recognition, ICDAR*, 2015, pp. 306–310.

[17] G. Toussaint, "Applications of the relative neighbourhood graph," *IJCSIA*, vol. 4, no. 2, pp. 77–85, 2014.

[18] F. Rayar, S. Barrat, F. Bouali, and G. Venturini, "An approximate proximity graph incremental construction for large image collections indexing," in *Foundations of Intelligent Systems: 22nd International Symposium, (ISMIS)*, 2015, pp. 59–68.

[19] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, no. 11, 1998, pp. 2278–2324.

[20] L. van der Maaten and G. Hinton, "Visualizing high-dimensional data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.