On Fast Sample Preselection for Speeding up Convolutional Neural Network Training

Frédéric Rayar and Seiichi Uchida

Kyushu University, Fukuoka 819-0395, Japan
{rayar,uchida}@human.ait.kyushu-u.ac.jp

Abstract. We propose a fast hybrid statistical and graph-based sample preselection method for speeding up CNN training process. To do so, we process each class separately: some candidates are first extracted based on their distances to the class mean. Then, we structure all the candidates in a graph representation and use it to extract the final set of preselected samples. The proposed method is evaluated and discussed based on an image classification task, on three data sets that contain up to several hundred thousands of images.

Keywords: Convolutional neural network · Training data set preselection · Relative neighbourhood graph

1 Introduction

Recently, Convolutional Neural Networks (CNN) [7] have achieve the state-of-the-art performances in many pattern recognition tasks. One of the property of the CNN, that allows to achieve very good performance, is the multi-layered architecture (up to 152 layers for ResNet). Indeed, the additional hidden layers can allow to learn complex representation of the data, acting like an automatic feature extraction module. Another requirement to take advantage of CNN is to have at disposal large amounts of training data, that will be used to build a refined predictive model. By large amounts, we understand up to several millions labelled data, that will allow to avoid overfitting and enhance the generalisation performance of the model.

Nonetheless, the combination of deep neural networks and large amount of training data implies that substantial computing resources are required, for both training and evaluation steps. One of the solution that can be considered is the hardware specialization, such as the usage of graphic processing units (GPU), field programmable gate arrays (FPGA) and application-specific integrated circuits (ASIC) like Google's tensor processing units (TPU). Another solution is sample preselection in the training data set. Indeed, 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 computational requirement.

In a recent work [9], the relevance of a graph-based preselection technique has been studied and it has been experimentally shown that it allowed to reduce the

training data set up to 76% without degrading the CNN recognition accuracy. However, one limitation of the proposed method was that the graph computation time could still be considered as high for large data sets. Hence, in this paper, we aim at addressing this issue and propose a fast sample preselection technique to speed up CNN training when using large data sets.

The contributions of this paper are as follows:

1. We propose a hybrid statistical and graph-based approach for preselecting training data. To do so, for each class, some candidates are first extracted based on their distances to the class mean. Then, we structure the candidates in a graph and use it to gather the final set of preselected samples.
2. We discuss the proposed preselection technique, based on experimentation on three data sets, namely CIFAR-10, MNIST and HW_R-OID (50,000, 60,000 and 740,348 training images, respectively), in image classification tasks.

The rest of the paper is organised as follows: Section 2 presents the paradigms on sample preselection and briefly reminds the work that has been done previously in [9]. Section 3 presents the proposed hybrid statistical and graph-based preselection method. The experimentation details are given in Section 4 and the results that have been obtained are discussed in Section 5. Finally, we conclude this study in Section 6.

2 Related Work

2.1 Training sample selection

Several sample selection techniques have been proposed in the literature, to reduce the size of machine learning training data sets. They can be organised according to the following three paradigms:

1. *"editing"* techniques, that aim at eliminating erroneous instances and remove possible class overlapping. Hence, such algorithms behave as noise filters and retain class internal elements.
2. *"condensing"* techniques, that aim at finding instances that will allow to perform as well as a nearest neighbour classifier that uses the whole training set. However, as mentioned in [4], such techniques are *"very fragile in respect to noise and the order of presentation"*.
3. *"hybrid"* techniques (editing-condensing), that aim at removing noise and redundant instances at the same time.

These techniques exploit either: (i) random selection methods [8], (ii) clustering methods [15] or graph-based methods [12] to perform the sample selection. One can refer to thorough surveys that have been done recently: in 2012, Garcia et al. [2] focus on the sample selection for nearest neigbour based classification. Stratification technique is used to handle large data sets and no graph-based techniques has been evaluated. In 2014, Jung et al. [5] shed light on the sample preselection for Support Vector Machine (SVM) [1] based classification. However, they evaluated only post-pruning methods, to address issues of application

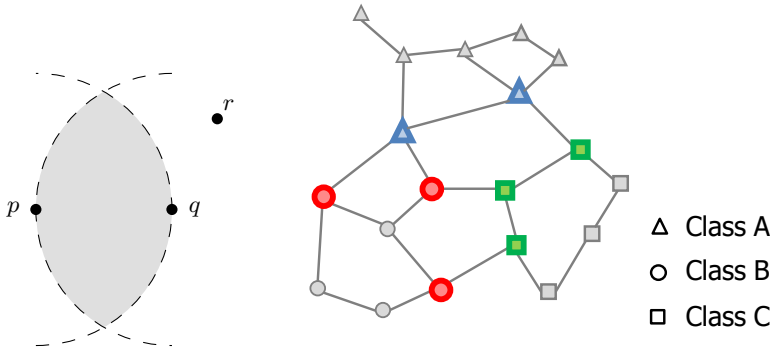

Fig. 1. (Left) Relative neighbourhood (grey area) of two points $p, q \in \mathbb{R}^2$. If no other point lays in this neighbourhood, then p and q are relative neighbours. (Right) Illustration of bridge vectors on a toy data set. The bridges vectors are highlighted with colours and thicker borders.

engineers. As confirmed by the existence of the two aforementioned surveys, sample selection has been widely studied for the nearest neighbour classifier and the SVMs However, to the best of our knowledge, no similar studies has been performed for CNN (or more generally neural networks). Conversely, the studies that use CNN usually focus on the acquirement of large training data sets, using crowdsourcing, synthetic data generation or data augmentation techniques.

2.2 Graph-based sample selection

G. Toussaint et al. [12] have been the first in 1985 to study the usage of a proximity graph [13] to perform sample selection for nearest neighbour classifiers using Voronoi diagrams. Following this study, several other proximity graphs have been used to perform training data reduction such as: the β-skeleton, the Gabriel Graph (GG), and the Relative Neighbourhood Graph (RNG). In this last study, the authors conclude that the GG seems to be the best fit for sample selection. More recently Toussaint et al. have used a graph-based selection technique and in a comparison study [14] against random selection, they conclude that *"proximity graph is useless for speeding up SVM because of the computation times"* and assert that *"a naive random selection seems to be better"*. However, they only evaluated their work with a data set of 1641 instances.

In [9], the efficiency of using a condensing graph-based approach to select samples for training CNN on large data sets has been experimentally shown. To do so, the RNG, that has been proven a good fit to preselect high-dimensional samples [14] in large training data sets [3], has been used. The method consisted in: (i) building the RNG of the whole training data set and (ii) extracting so-called *"bridge vectors"*, that correspond to nodes that are linked to another class node by an edge in the RNG. The bridge vectors are the final set of preselected training samples that are then fed to the CNN. Figure 1 illustrates the

RNG relative neighbourhood definition (left) and the notion of bridge vectors (right). This preselected set allowed to reduce the training data set up to 76% without degrading the recognition accuracy, and performed better than random approaches. However, the RNG computation of the whole training data sets can remain an issue when dealing with large data sets. Hence, in this study, we aim at addressing this issue by proposing a fast hybrid statistical and graph-based preselection method.

3 Fast hybrid statistical and graph-based sample preselection

Since the issue of the RNG computation is related to the number of data in the whole training data set, one first idea that comes to mind is to take advantage of the supervised property of the CNN-based classification, and build an RNG for each class. Then, the preselection boils down to gather the data that lie in each class border. However, both exact (*e.g.* cluster boundaries) and approximative (*e.g.* low betweenness centrality nodes) approaches still require high computation requirements (*e.g.* all-pair shortest path computation). To address this, we propose to first extract some candidates for each class using a statistical approach, and then use a graph-based approach on the candidates subset.

3.1 Frontier vectors

One of the goal of this study is to preselect samples that are similar to the bridge vectors presented in 2 (see Figure 1 (right)). Since these bridge vectors may lie in the frontiers of classes, we propose to perform a simple statistical-based candidates selection for each class. To do so, for each class C, we: (i) compute the mean, μ_C, (ii) compute the distances of each element $x \in C$ to the mean, $\delta(x, \mu_C)$, (iii) sort these distances by ascending order, (iv) select elements that are above a given distance D to the mean.

The elements that are gathered in this way are among the farthest to the mean, hence they have a better chance to lie in the boundary of the class. The extracted candidates at this step are later called *"frontier vectors"* (FV). Figure 2 presents the plots of the sorted distance distribution of the two first classes of the HW_R-OID data set.

3.2 Automatic threshold computation

The last step to gather the frontier vectors of a given class, is to select elements that are above a given distance D to the mean. Given the shapes of the curves presented in Figure 2, it corresponds to select the elements on the right part of the curve. The issue of the value of D arises: one naive solution could be to set a value regarding the number of elements of the class. However, this strategy has two drawbacks: (i) it introduces an empirical parameter that may have an impact on the results and (ii) it does not fit the observations made during the

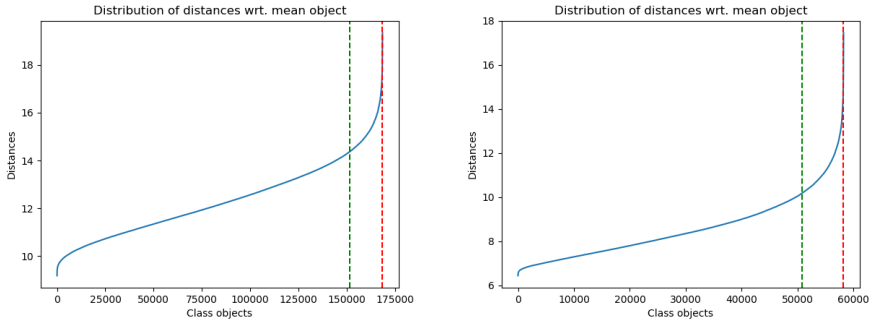

Fig. 2. Distribution of the sorted distances of a given class elements wrt. the class mean. We present here the distribution only for the two first classes of the largest data set (HW_R-OID), due to space allowance. The red vertical dotted line corresponds to the threshold that is obtained using a basic maximum curvature criterion strategy, and the green one corresponds to one obtained the sliding-window maximum curvature criterion strategy.

study of [9] on the bridge vectors. Indeed, no direct relation was found between the number of elements of a class and its number of bridge vectors.

To address the automatic computation of this parameter, we propose to use a maximum curvature criterion. For a given data set, let us consider a given class C. We denote n the number of elements of C, μ the mean of C, y the curve defined by the sorted distances $\delta(x, \mu)$ of each element $x \in C$ (in ascending order), and y', y'' the first and second derivative of y, respectively. Then, we define the curvature criterion γ as follows:

$$\gamma(x) = \frac{y''}{(1 + y'^2)^{3/2}}, \text{ where } x \in [\![1, n]\!].$$

A naive strategy consists in finding the index of the maximum curvature value of y; however, it may result in favouring indices associated to high values, and will gather only a few number of the class elements. This phenomenon could be seen in Figure 2: the red vertical dotted lines correspond to the thresholds computed using the naive strategy.

To circumvent this problem, we propose to use a sliding window maximum curvature criterion strategy. Such a strategy has already been used efficiently in a previous work [10]. Let us define the set of windows $W = \cup_{i \in [\![1, n-m]\!]} W_i$, where $W_i = \{w_1^i, ..., w_m^i\}$. $w_k^i \in [\![1, n]\!]$ are the indices of window W_i and m is the size of the windows. Hence, we have $|W| = n - m + 1$ windows defined on the interval $[\![1, n]\!]$. We then define the window's curvature γ_i:

$$\gamma_i = \gamma(W_i) = \frac{\frac{1}{m} \sum_{w \in W_i} \gamma(w)}{\max\limits_{w \in W_i} \gamma(w)}.$$

By selecting the maximum curvature over the set of windows, we have:

$i^* = \text{argmax}_{i \in \{1 \ldots |W|\}} \gamma_i$, and thus deduce $D = \delta(i^*, \mu)$.

Figure 2 illustrates the relevance of the sliding-window maximum curvature criterion strategy. We have set $m = \frac{n}{10}$ to have a trade-off between the global and local maximum curvature. For a given data set and a given class, the green dotted vertical line in the plot corresponds to the value of i^* that has been automatically computed.

3.3 Overall algorithm

Since the frontier vectors correspond to class boundaries, they may appear in a part of the feature space that do not correspond to classes frontiers. Hence, we use the bridge vectors extraction, proposed in the study of [9], but only on the frontier vector subset, addressing the high RNG computation time. Furthermore, this also allows to balance the fact that the proposed automatic threshold strategy does not extract only the farthest elements of a given class. The bridge vectors extracted at this step form the final preselected set of samples. We refer to these samples as *"frontier bridge vectors"* (FBV) in the rest of the paper. Algorithm 1 summarises the proposed hybrid statistical and graph-based sample preselection strategy.

4 Experimental setup

4.1 Data sets

To evaluate the proposed preselection method, we have used three data sets. First, the CIFAR-10 [6] data set is a subset of the Tiny Images [11] data set, that has been labelled. It consists of ten classes of objects with 6000 images in each class. The classes are: "airplane, automobile (but not truck or pickup truck), bird, cat, deer, dog, frog, horse, ship, and truck (but not pickup truck)", as per the definition of the data set's creator. We have used 50,000 images in the training data set and 10,000 for testing purpose. Second, the MNIST [7] data set, that corresponds to 28 $\times$ 28 binary 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. Last, the HW_R-OID data set is an original data set from [16]. It contains 822,714 images collected from forms written by multiple people. The images are 32 $\times$ 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 to gathered to build the training subset.

For the three aforementioned 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 two images.

Algorithm 1: Fast hybrid statistical and graph-based sample preselection algorithm

Input: $DATA$ // data features per class
Input: δ // distance function
Output: FBV // final preselected sample list

$FV = []$
for *each class* c **do**
  $n =$ number of elements in c
  $m = \frac{n}{10}$
  Compute class mean μ
  $list = []$
  for *each* $x \in c$ **do**
    Append $\delta(x, \mu)$ to $list$
  end
  Sort $list$ (by ascending order)
  Compute i^*
  Append elements of c at $[\![i^*, n]\!]$ to FV
end
$RNG =$ Build graph from FV
$FBV =$ Extract BV from RNG

4.2 Workflow

The goal are to evaluate the relevance of the proposed preselection technique, but also compare its performance to the bridge vectors of the study of [9]. 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,
- FV: only the extracted frontier vectors of WHOLE,
- FBV: only the extracted bridge vectors of the RNG build from FV,
- $\text{RANDOM}_{\text{FBV}}$: a random subset of WHOLE, with approximatively the same size as FBV.

4.3 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 of the study of [9] have been used. Regarding the CNN architecture, namely modified LeNet-5 is used: the main difference with the original LeNet-5 [7] is the usage of ReLU and max-pooling functions for the two CONV layers. As mentioned in [16], 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).

Table 1. BV and FBV preselection strategy computation times (in seconds).

	Data set	CIFAR-10	MNIST	HW_R-OID
BV	Data load	2	133	1,397
	RNG/BV computation	211	304	61,270
	Total	213	437	62,667
FBV	Data load	18	24	1,434
	Statistical pruning	3	4	147
	RNG/BV computation	9	5	622
	Total	40	32	2,203

No pre-initialisation of the weights is done, and the CNN is trained with an Adadelta optimiser on 10 epochs for the two handwritten digit data sets, and an Adam optimiser on 100 epochs for the CIFAR-10 data set. The Adam optimiser has been chosen for the CIFAR-10 data set to avoid the strong oscillating behaviour during the training observed when using the Adadelta optimiser. During our experimentation, 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.

5 Results

5.1 Preselection method computation times and data reduction

One of the goal of the present study is to address the high RNG computation requirement observed during the preselection phase in large training data sets. Table 1 presents the computation times of the previous preselection strategy, namely the bridge vectors, and the one proposed in this study, namely the frontier bridge vectors. For the three data sets, a major speed-up ratio is obtained: 5.33, 13.65 and 28.44 for CIFAR-10, MNIST and HW_R-OID, respectively. For the largest data set, it represents a reduction of the preselection computation time from $17h25m$ to $37m$.

Table 2 presents for each data sets, the size of the underlying training data sets in the first rows. Previously, using the bridge vectors as preselected samples, we have obtained a reduction of the training data set, up to 76%. By using the proposed hybrid preselection strategy, we achieve a data reduction that goes up to 96.57% (for the largest data set). Furthermore, we note that the hybrid approach, which extracts bridge vectors from the frontier vectors, allows its own data reduction. Indeed, this step allows to reduced the data, up to 69% between the FV and the FBV.

This reduction of the training data set has an expected impact on the CNN training time, with a speed-up ratio up to 15. The third rows of Table 2 present the average computation time per epoch.

Table 2. Classification results: (i) size of the training data set, (ii) average recognition accuracy and (iii) average training time per epoch (in seconds) are presented.

	Training data set	WHOLE	BV	FV	FBV	$\text{RANDOM}_{\text{FBV}}$
CIFAR-10	# training data	50,000	41,221	8,713	6,845	6,850
	accuracy (%)	76.65	75.17	59.05	58.63	61.45
	epoch time (s)	42	35	9	7	7
MNIST	# training data	60,000	22,257	6,637	2,876	2,880
	accuracy (%)	98.79	98.78	96.22	95.25	94.69
	epoch time (s)	24	10	3	2	2
HW_R-OID	# training data	740,438	173,808	80,477	25,395	25,397
	accuracy (%)	99.9343	99.9314	99.7460	99.7085	99.4307
	epoch time (s)	412	107	56	27	27

5.2 Preselection method efficiency

Table 2 also presents the average accuracies obtained for all the training data sets introduced in Section 4.2 for the three data sets. Several observations can be made from these results.

For the two handwritten isolated digit data sets, we have:

$$\text{WHOLE} \approx \text{BV} > \text{FV} > \text{FBV} > \text{RANDOM}_{\text{FBV}} \tag{1}$$

Furthermore, the average recognition rates obtained using only the FBV are in the same order of magnitude to the ones obtained when using the whole training data set: -3.54% and -0.2258% for MNIST and HW_R-OID, respectively. However, the same observation can be made for the $\text{RANDOM}_{\text{FBV}}$ training set, which may be interpreted as an indicator that either the data sets are lenient or that the FBV are not discriminative enough on their own in the training of the CNN.

For CIFAR-10, we observe a different behaviour that the one mentioned above. First, the relation described in Equation 1 does not stand. Indeed, the average accuracy obtained for $\text{RANDOM}_{\text{FBV}}$ is higher than both the ones of FV and FBV. Furthermore, the degradation in terms of average accuracy between $\{WHOLE, BV\}$ and $\{FV, FBV, \text{RANDOM}_{\text{FBV}}\}$ is no more negligible: around -16%. These results may be due to the strong dissimilarity between this data set class elements.

6 Conclusion

In this paper, we have proposed a fast sample preselection method for speeding up convolutional neural networks training and evaluation. The method uses a hybrid statistical and graph-based approach to reduce the high computational

requirement that was due to the graph computation. Hence, it allows to drastically reduce the training data set while having recognition rate of the same order of magnitude for two of the studied data sets.

Future works will be to perform experimentation on another data set, to evaluate the generalisation of the proposed method. We also aim at starting 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. Cortes, C., Vapnik, V.: Support-vector networks. In: Machine Learning. pp. 273–297 (1995)
2. Garcia, S., Derrac, J., Cano, J., Herrera, F.: Prototype selection for nearest neighbor classification: Taxonomy and empirical study. In: IEEE Transactions on Pattern Analysis and Machine Intelligence. pp. 417–435 (2012)
3. Goto, M., Ishida, R., Uchida, S.: Preselection of support vector candidates by relative neighborhood graph for large-scale character recognition. In: ICDAR. pp. 306–310 (2015)
4. Jankowski, N., Grochowski, M.: Comparison of instances seletion algorithms i. algorithms survey. In: ICAISC. pp. 598–603 (2004)
5. Jung, H.G., Kim, G.: Support vector number reduction: Survey and experimental evaluations. In: IEEE Transactions on ITS. pp. 463–476 (2014)
6. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. In: Computer Science Department, University of Toronto, Tech. Rep (2012)
7. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. In: Proceedings of the IEEE. pp. 2278–2324 (1998)
8. Lee, Y.J., Huang, S.Y.: Reduced support vector machines: A statistical theory. In: IEEE Transactions on Neural Networks. pp. 1–13 (2007)
9. Rayar, F., Goto, M., Uchida, S.: CNN training with graph-based sample preselection: application to handwritten character recognition. CoRR **abs/1712.02122** (2017)
10. Razafindramanana, O., Rayar, F., Venturini, G.: Alpha*-approximated delaunay triangulation based descriptors for handwritten character recognition. In: ICDAR. pp. 440–444 (2013)
11. Torralba, A., Fergus, R., Freeman, W.T.: 80 million tiny images: A large data set for nonparametric object and scene recognition. pp. 1958–1970 (2008)
12. Toussaint, G. T., B.K.B., Poulsen, R.S.: The application of voronoi diagrams to non-parametric decision rules. In: Computer Science and Statistics. pp. 97–108 (1985)
13. Toussaint, G.T.: Some unsolved problems on proximity graphs (1991)
14. Toussaint, G.T., Berzan, C.: Proximity-graph instance-based learning, support vector machines, and high dimensionality. In: MLDM. pp. 222–236 (2012)
15. Tran, Q.A., Zhang, Q.L., Li, X.: Reduce the number of support vectors by using clustering techniques. In: ICMLC. pp. 1245–1248 (2003)
16. Uchida, S., Ide, S., Iwana, B.K., Zhu, A.: A further step to perfect accuracy by training CNN with larger data. In: ICFHR. pp. 405–410 (2016)