SVMNET: NON-PARAMETRIC IMAGE CLASSIFICATION BASED ON CONVOLUTIONAL SVM ENSEMBLES FOR SMALL TRAINING SETS

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep convolutional neural networks (DCNNs) have demonstrated superior power in their ability to classify image data. However, one of the downsides of DCNNs for supervised learning of image data is that their training normally requires large sets of labeled "ground truth" images. Since in many real-world problems large sets of pre-labeled images are not always available, DCNNs might not perform in an optimal manner in all real-world cases. Here we propose SVMnet – a method based on a layered structure of Support Vector Machine (SVM) ensembles for non-parametric image classification. By utilizing the quick learning of SVMs compared to neural networks, the proposed method can reach higher accuracy than DCNNs when the training set is small. Experimental results show that while "conventional" DCNN architectures such as ResNet-50 outperform SVMnet when the size of the training set is large, SVMnet provides a much higher accuracy when the number of "ground truth" training samples is small.

1 INTRODUCTION

Deep convolutional neural networks (DCNNs) are powerful tools for multiple tasks of automatic image analysis, demonstrating paramount success and consequently gaining substantial popularity over the past decade. By analyzing the pixels directly, CNNs can be applied to various types of image content without the need to develop task-specific algorithms, and can easily be applied to a broad range of domains with excellent performance (Khan et al., 2020).

One of the major weaknesses of modern DCNNs is their dependence on a large set of examples for training. Cutting-edge DCNNs can have hundreds of layers, each with thousands of trainable parameters. For instance, the common ResNet-50 (He et al., 2016) contains over $2{\cdot}10^6$ artificial neurons. Therefore, to achieve meaningful performance and avoid underfitting, DCNNs normally rely on relatively large training sets.

Training DCNNs normally requires large datasets of labeled ground truth images. Commonly used datasets include benchmarks such as the Modified National Institute of Standards and Technology (MNIST) or ImageNet. These benchmark datasets provide tens of thousands of images with high-quality annotations for training deep CNNs, and are commonly used for testing the performance of CNNs. However, in many cases of real-world image classification problems, large datasets of clean, labeled ground truth are not available. For instance, in the biomedical domain the time required to annotate radiographs by a committee of expert pathologists can be over one hour for a single image. This process is time consuming and can be prohibitively expensive, making such datasets far smaller than MNIST or ImageNet.

A common solution to increasing the size of the training set is data augmentation, in which different modifications of the images in the original dataset can lead to more training samples and increase the size of the training set. However, that strategy can also lead to biases by overusing the same samples. In some cases transfer learning can be used to better train neural networks using pre-trained models. However, the need for a large number of training samples is a practical downside of CNNs, making it difficult to use in an optimal manner in many real-world cases.

This paper explores a new form of image classification based on an ensemble composition of support vector machines (SVMs). Inspired by CNN architecture, SVMnet utilizes a large number of SVMs, and uses the classification power of SVMs organized in an architecture that provides layers equivalent to the convolutional layers often used in CNN architecture. An SVM (Cortes & Vapnik, 1995) is less sensitive to high-dimensionality feature spaces (Chen et al., 2005; Mukkamala et al., 2002), and has the advantage of learning from a relatively small number of training samples (Shin et al., 2005; Byvatov et al., 2003; Paiva et al., 2018) compared to other supervised machine learning approaches. That makes SVMs suitable in the common real-world cases where the number of clean labeled training samples is limited.

2 ARCHITECTURE OF SVMNET

The proposed SVMnet architecture is designed as a hierarchical ensemble of numerous simple SVM classifiers organized into one or more layers. Each layer is an array of SVMs which functions similarly to a convolutional layer in a CNN. Each SVM in a layer is independent and all are assigned an equal-sized rectangular portion of the layer's input, referred to as a window. Variable stride length and padding, as described in Chapter 2 of Dumoulin & Visin (2018), are specified as hyperparameters. The output of the layer can be a 2-dimensional array of class labels or a 3-dimensional array of per-class probabilities, specified at instantiation. Each class label or probability vector is produced by a single SVM.

When a layer is fitted, each SVM in the layer is fitted against ground truth labels. The input to the SVM is the flattened portion of each input image that is within the SVM's window. Each pixel channel within the window is essentially treated as one input feature. For instance, a 5×5 window would create a 25-feature SVM for grayscale input and a 75-feature SVM for 3-channel RGB input. During this step, the SVMs may be given weights based on the accuracy of the fit on the training set. These weights are accessible by the following layer in the model. Each SVM then predicts a class label or a vector of class probabilities for its window of each input, creating the input matrix for the next layer.

Figure 1 shows a simple layer in SVMnet. Each node in the layer is one SVM, trained using the ground truth labels for the input samples. The weights are determined based on the classification accuracy of the SVM compared to the ground truth. That is, during training each SVM is tested for its accuracy compared to the ground truth of the training set, and the classification accuracy of the SVM is used to determine the weight using a weight function that will be described later in this Section.

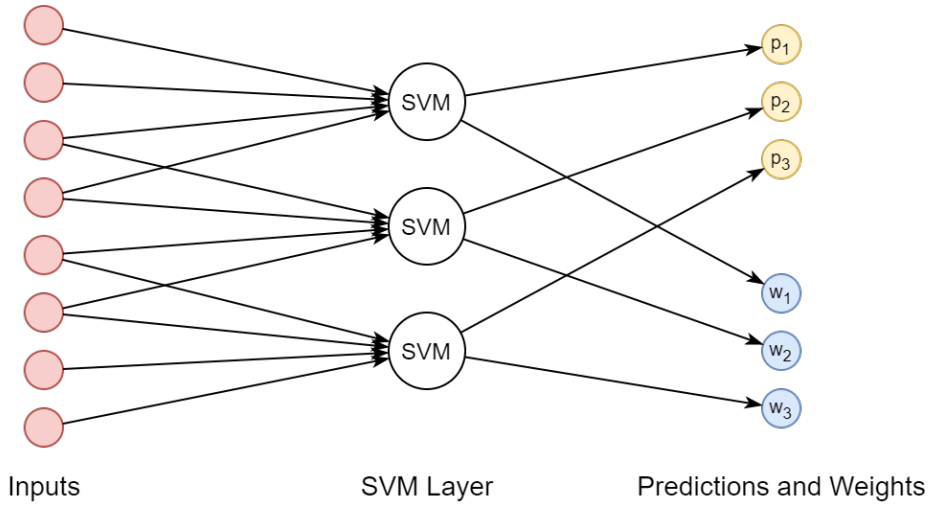

Figure 1: Example of a simple weighted layer of SVMnet. Each node in the layer is an SVM, trained with a subset of the inputs (pixels).

To produce a single class label for each input, the model performs a weighted vote after the final layer. This vote combines the results of the final layer by treating each value as a vote for that class label. If the final layer is weighted, these are used to weigh the votes in favor of SVMs with higher accuracy. By combining the outputs of the final layer, the vote can be conceptualized as analogous to the final fully connected layers in a standard CNN.

$$S_c = \Sigma_i \eta(A_i)[P_i = c] \tag{1}$$

The total voting score S_c of each class c is calculated by Equation 1, where A_i is the accuracy score of SVM i in the final layer, η is the weight function, and P_i is the class label predicted by SVM i. That is, if the predicted label P_i of SVM i is class c, the weighted score $\eta(A_i)$ is added to the vote for that class. The weight function emphasizes the predictions of the SVMs with higher accuracy during training. The class that has the highest score S_c is chosen as the predicted label by the model for the given sample. The weight function η in this experiment is defined as $\eta(x) = x^2$, where x is the classification accuracy of the SVM, determined during training.

While the layers support arbitrary estimators, in SVMnet we use only support vector machines (SVM), hence the name SVMnet. The SVMs are trained with a Radial Basis Function (RBF) kernel (Chang & Lin, 2011) and scaling gamma value, and they continue to iterate until convergence with a 0.001 tolerance. The ability to choose different estimators in each layer can be compared to the ability to use different activation functions in the layers of neural networks.

Figure 2 illustrates one possible two-layer SVMnet architecture. Each SVM in the first layer analyzes a specific neighborhood of pixels in each image and is fitted independently against ground truth labels. These SVMs then produce a vector of class probabilities for the same pixel region which forms the input matrix for the following layer. The SVMs in the second layer are fitted on a region of these probabilities and predict a class label for the image. These labels are then tallied in a final vote to produce one label for the input.

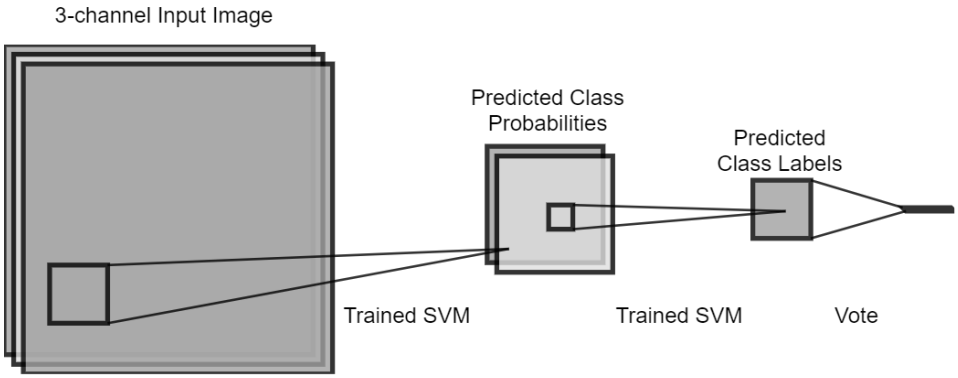

Figure 2: Example SVMnet architecture. The first layer trains SVMs on pixel regions from the input image (left) and outputs a matrix of class probabilities (in this case, a binary classification). The second layer trains SVMs on these probabilities and outputs a matrix of class labels, which are then tallied to produce one label (right).

2.1 DROPOUT

Not every window is expected to produce a well-informed SVM. Some regions of the images, particularly towards the edge, often lack the details necessary to distinguish samples from each other. This can cause the outputs of these SVMs to act as noise in a vote tally. Even with the expected low accuracy score of the SVM depressing the weight of its vote, if the low-information regions are large then enough inaccurate votes may overwhelm the more accurate votes. To help prevent this, a dropout system is implemented for the vote tally.

When using dropout, the number of SVMs to drop is calculated when fitting the model. The final layer of the model must be weighted. First, the SVMs are ordered from the highest weight (by

default, the square of the SVM's accuracy score) to the lowest. Votes are then cumulatively tallied one SVM at a time with the accuracy of the votes measured between each tally. The SVMnet then finds the global maximum accuracy of the cumulative tally (if the value occurs multiple times, the tally that includes the most SVMs is used). This marks the point where including the votes of the less-accurate SVMs lowers the overall accuracy of the tally, so those SVMs are marked for dropout. When the model is used to make predictions, the vote will only include the outputs of the SVMs that contributed to the most accurate tally.

In most cases during testing, automatic dropout resulted in equal or better performance than without dropout, as the least informative regions of the image were ignored. However, as with all hyperparameters, performance sometimes decreased and required fine-tuning. In each of the experiments described in Section 3, the SVMnet model presented is the one with the highest-performing hyperparameters among the combinations tested.

3 EXPERIMENTAL RESULTS

To test the efficacy of SVMnet compared to a "conventional" CNN, several experiments were performed using common, relatively small datasets. The purpose of SVMnet is not to outperform CNNs in the general case, but to achieve higher accuracy when the number of labeled training images is limited. Therefore, the experiments were made with different sizes of training sets to compare the classification accuracy as the training set increases.

The performance of the SVMnet was compared to the performance of residual network, or ResNet, models with 18, 34, and 50 layers (He et al., 2016). ResNet is a powerful architecture that was designed to reduce the number of required training samples for deep learning tasks and has demonstrated excellent efficacy in image classification. All ResNet models were trained from scratch rather than starting with pre-trained weights derived from ImageNet. Following the practice in He et al. (2016), the final convolutional layer is followed by a global average pooling layer, then by a single fully-connected layer with softmax activation and as many units as class labels in the respective task. Models are trained using stochastic gradient descent (SGD) optimization. We differ by adjusting the learning rate for each experiment and training the ResNet models for 50 epochs.

While the height and width of inputs can be adjusted for ResNet, the architecture always expects 3-channel RGB color images. Grayscale images were modified for use by ResNet by duplicating the pixel values into three equal channels. This approach was used in Section 3.2 and Section 3.3. Before training and classification by ResNet, images were also passed through a preprocessing filter provided by the Keras library to prepare the data for ResNet models. All inputs were normalized by dividing by the mean and subtracting the variance before being used to train SVMnet. For RGB color inputs, the images were normalized per-channel.

All experiments and analysis presented in this section used the same hardware environment. SVMnet was parallelized across 16 cores of Intel Xeon Gold 6130 CPUs, and ResNet models were trained on an nVidia GeForce GTX 2080 GPU.

3.1 COIL-100 OBJECT RECOGNITION

Columbia Object Image Library (COIL-100) is a common dataset used for basic object recognition (Nene et al., 1996). It contains RGB color images of 100 different objects, each photographed 72 times at $5\,\mathrm{deg}$ increments about the vertical axis. Background details were removed in all images and the objects are centered and enlarged to fill the frame. Some objects contained in this dataset include coffee mugs, small toy cars, and various fruits and vegetables.

The SVMnet in this experiment used one layer with a 25×25 window (giving each SVM 1875 input features) and a stride length of 7, followed by a weighted vote with dropout. The SVMnet and ResNet models were fitted with 100-500 training images in increments of 100, each controlled to have an equal number of samples from each object.

Figure 3 shows the accuracy of the models for each training set when evaluated on the remaining images not shown to the model during training, as well as the time taken to train each model. When fitted on the smallest training set, containing only one example image per object, SVMnet correctly predicted labels for over 60% of the remaining COIL-100 images. With the same training set, all

three ResNet models performed only marginally better than random chance. SVMnet achieved higher accuracy with less training time compared to all three ResNet models with the two smallest training sets.

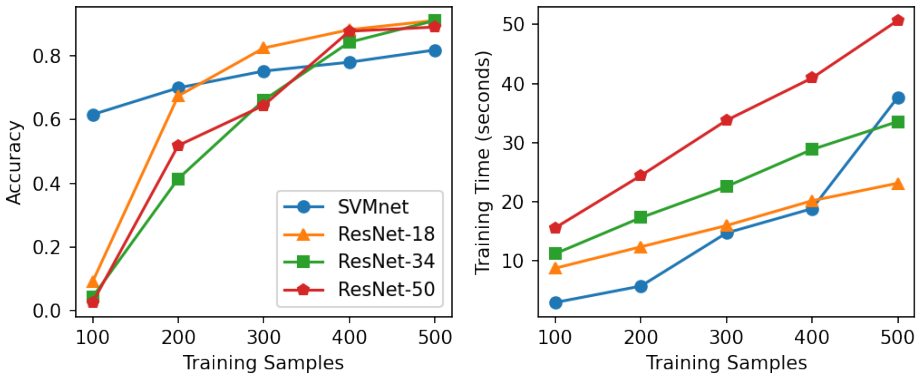

Figure 3: Test-set accuracy and training time of SVMnet and ResNet on COIL-100 images when fitted with different training set sizes.

3.2 COVID-19 RADIOGRAPHY

During the COVID-19 pandemic, machine learning techniques have been applied to various kinds of data to assist the medical community in making accurate diagnoses (El-Din Hemdan et al., 2020; Gangloff et al., 2021; Li et al., 2020). During the early stages of a disease outbreak, diagnostic data is expected to be limited or sparse, making it difficult to train most kinds of machine learning models. A type of model capable of learning from a small number of samples would be the most effective in this time frame.

Here we apply SVMnet to a database of chest x-ray images from healthy patients and patients diagnosed with COVID-19 (Chowdhury et al., 2020; Rahman et al., 2021). In this experiment, only the images labeled as "Normal" and "COVID" are used. Images were downscaled to 128×128 pixels (approx. 43% of the original size). An equal number of images were selected from each class, totaling 7232 samples. SVMnet and ResNet models were fitted with 10, 20, 50, 100, and 200 training samples. The SVMnet uses two layers: the first with window size 19, stride 7, and class probability outputs; the second with window size 5 and stride 5, followed by an unweighted vote.

Figure 4 shows the accuracy of the models for each training set when evaluated on the remaining images not shown to the model during training, as well as the time taken to train each model. SVMnet was able to correctly label between 64% and 78% of unseen x-rays depending on the number of training samples, but the ResNet models were not able to surpass 50% accuracy (as they assigned the same label to all samples). Additionally, SVMnet was several times faster to train than ResNet.

3.3 ASTRONOMICAL IMAGE DATA

To test the performance of SVMnet on a current real-world image classification problem, a dataset of galaxy images from the Panoramic Survey Telescope and Rapid Response System (Pan-STARRS) was used. The dataset is made of galaxies separated into elliptical and spiral morphology. The galaxy images were taken from the catalog of Pan-STARRS galaxies classified by their morphological type (Goddard & Shamir, 2020).

An equal number of images were selected of each morphological type, totaling 26,732 samples. Each image is grayscale and has a dimension of 120×120 px. SVMnet and ResNet models were fitted with 10, 20, 40, 80, 160, 320, and 640 training samples. The SVMnet uses one layer with a window size of 22 and stride 5, followed by a weighted vote with dropout.

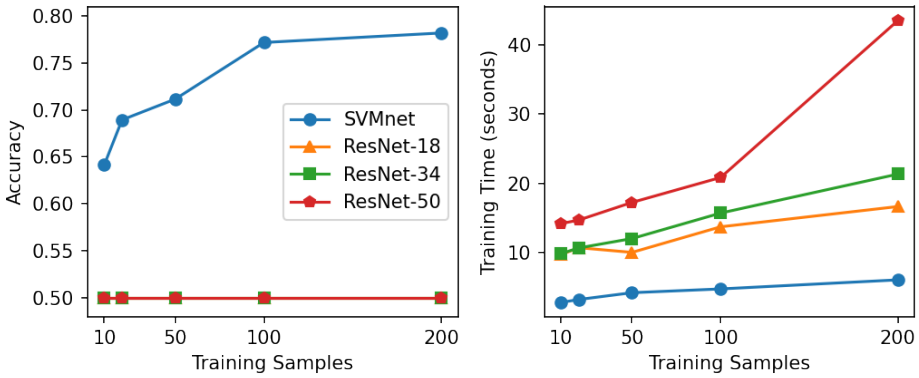

Figure 4: Test-set accuracy and training time of SVMnet and ResNet on COVID-19 chest x-ray images when fitted with different training set sizes. The accuracy of each ResNet model (left) is 50% in all cases, causing overlap in the figure.

Figure 5 shows the accuracy of the models for each training set when evaluated on the remaining images not shown to the model during training, as well as the time taken to train each model. As the graph shows, when the number of training samples is relatively small, SVMnet provides a significantly higher classification accuracy compared to ResNet models, which were unable to distinguish classes until over 100 training samples were used. When the training set gets larger the performance of the ResNet models increases significantly, slightly overtaking SVMnet with the largest training set used. With all training sets, SVMnet finished training faster than all ResNet models.

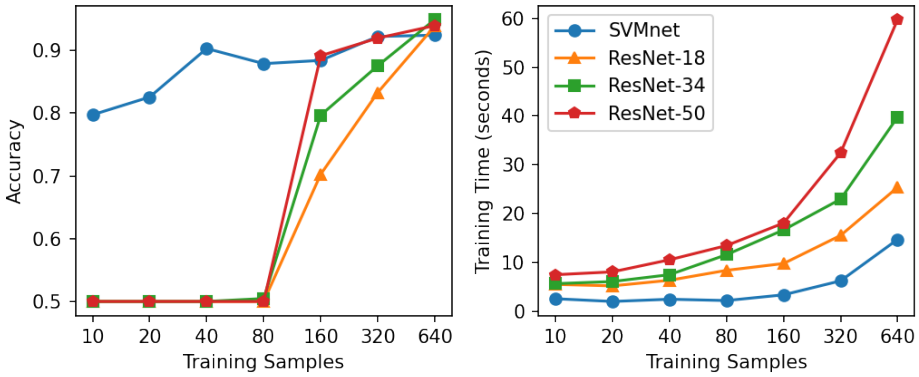

Figure 5: Test-set accuracy and training time of SVMnet and ResNet on Pan-STARRS galaxy images when fitted with different training set sizes.

3.4 Computational complexity

The complexity of fitting an SVM is asymptotic and polynomial. For a training set containing n samples, the algorithm is dominated by either an n^2 term or an n^3 term based on the formulation of the problem (Bottou & Lin, 2007). Therefore, training a large number of SVMs can be a computationally demanding task, and can lead to substantial computational complexity during training.

The number of SVMs N in a layer receiving rectangular input with width I_x and height I_y is given by Equation 2. The window size W (equivalent to the kernel size in other CNN literature), stride length S, and padding amount P in their respective dimensions follow from standard convolutional

arithmetic. When using a square window on square input, the formula can be simplified to Equation 3.

$$N = \left(\frac{I_x + 2P_x - W_x}{S_x} + 1\right) \cdot \left(\frac{I_y + 2P_y - W_y}{S_y} + 1\right) \tag{2}$$

$$N = \left(\frac{I + 2P - W}{S} + 1\right)^2 \tag{3}$$

Fitting a layer in SVMnet requires fitting N SVMs - a polynomial time operation. If the layer includes weights, then the SVMs must predict a class label for each input during the fit step, which scales linearly with the number of samples n. When using dropout as described in Section 2.1, SVMnet performs an additional step during training that scales linearly with n. Thus, fitting SVMnet is dominated by the polynomial fit time of the SVMs.

CNNs can theoretically be trained infinitely, but there is a definitive point at which the SVMs within SVMnet converge. This places a soft upper bound on the training time of SVMnet based on the tolerance parameter of the SVMs. Additionally, a firm upper bound may be placed on the number of iterations of the SVM algorithm, allowing for a shortened training time at the expense of some accuracy.

SVMnet trains multiple SVMs simultaneously using process-based parallelism and shared memory, greatly increasing its speed on typical multicore computers with minimal overhead. While this allows SVMnet to be set up quite easily on relatively inexpensive systems, the potential performance gain from extra hardware is minimal compared to the extreme optimization of CNNs for GPU devices.

While the training of SVMnet is slower than CNNs when the size of the training set becomes relatively large, SVMnet is designed for situations in which the size of the training set is small. Therefore, the computational complexity of the training is not expected to introduce a major obstacle in many real-world cases where the size of the training set is limited, and the time required for training does not necessarily explode to an unmanageable response time in the situations where SVMnet is most effective.

3.4.1 Response time of image classification

Predicting a single class label of an image using SVMnet typically requires a large number of individual SVMs to predict a label followed by a vote tally. Despite its affinity for parallelization, this process is expected to take longer than the highly optimized matrix operations of a CNN. Table 1 compares the response time of SVMnet and ResNet to make predictions using the COIL-100 dataset.

	1	10	100	1000
SVMnet	2.36	2.66	3.81	24.2
ResNet-18	0.054	0.056	0.082	0.296
ResNet-34	0.060	0.060	0.098	0.410
ResNet-50	0.061	0.064	0.106	0.515

Table 1: Comparison of the response time (in seconds) of SVMnet and ResNet to predict class labels for 1, 10, 100, and 1000 samples of the COIL-100 dataset.

The comparison shows that SVMnet is significantly slower than ResNet for classifying samples, but the speed of classification is still practical for many real-world systems. The parallelization of SVMnet greatly reduces the time needed to make predictions, but the overhead of shared memory operations is significant in the case of few samples.

3.5 Architecture Comparison

As with CNNs, SVMnet can be configured into a wide variety of architectures which are expected to differ in performance depending on the classification task. Due to the high number of possible

models, determining which is the most effective for a single task is non-trivial. In this section we show how a variety of SVMnet configurations were tested on the COIL-100 dataset to inform the choice of model used in Section 3.1. Similar methods were used to select the models for other datasets.

Figure 6 shows how the performance of a one-layer SVMnet changes with the window size, stride length, voting method, and number of training samples when fitted to COIL-100. Prediction accuracy improves in all cases as the window size increases but with diminishing returns. Increasing the stride length tends to lower accuracy when the window is small but incurs little to no penalty when the window is large. When the vote of an SVM is weighed with the square of its accuracy on the training set, model accuracy improves in all cases compared to an unweighted vote; performance increases further when inaccurate SVMs are ignored entirely with automatic dropout. This effect is more significant when the window size is small.

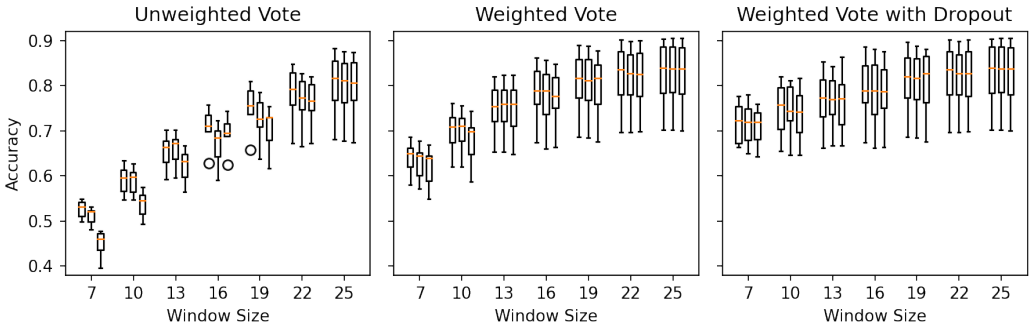

Figure 6: Prediction accuracy of one-layer SVMnet architectures fitted to COIL-100. Each group of three box plots represents the same window size with stride length 3, 5, and 7, respectively. Each box plot shows the distribution in model accuracy when fitted with 200-1000 training images (in increments of 200). The final vote is either unweighted (left), weighted without dropout (center), or weighted with dropout (right).

Figure 7 shows how the time required to fit SVMnet on COIL-100 changes with the number of SVMs (see Equation 3) and the number of features for each SVM (in this case equal to $3W^2$). Since increasing the stride length significantly reduces the number of SVMs in the model, an SVMnet with large windows can still be trained quickly with only a minor increase in stride without sacrificing accuracy. Each SVMnet in this experiment was trained in parallel using 16 CPUs.

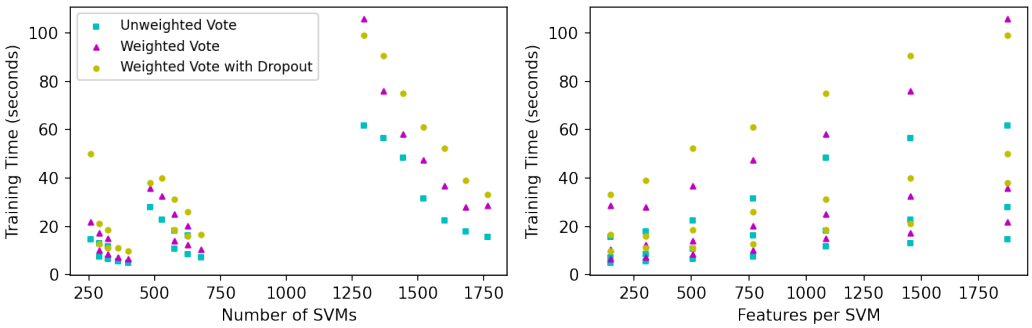

Figure 7: Training times for one-layer SVMnet architectures fitted to COIL-100.

4 CONCLUSION

Deep convolutional neural networks provide excellent performance in automatic classification of image data while eliminating the need to develop and tailor algorithms for specific image classification problems. With the availability of libraries, DCNNs have become the de facto first solution to image classification.

Here we explore one of the primary weaknesses of DCNNs, which is the need of a relatively high number of labeled "ground truth" samples for effective training of the network. While DCNNs are often tested on relatively large datasets such as MNIST or ImageNet, in many real-world problems a very large number of clean labeled samples that can be used for training is not available.

For instance, medical datasets such as image-based diagnostics are difficult to prepare due to the long time required to assign a sample with a label, consequently leading to a high cost. Additionally, acquiring a radiograph can also require substantial resources, as medical image acquisition systems such as Magnetic Resonance Imaging (MRI) require expensive instrumentation and staff. Additionally, the consent of the patient is required for each sample. These limitations make large datasets of biomedical images substantially more expensive and more difficult to prepare.

In many other cases labeled training samples are not available. For instance, when analyzing archaeological artifacts, the number of training samples are limited by the number of available artifacts, which is a hard limit that cannot be easily changed. Using computer vision to analyze art (Khan et al., 2014) is limited by the number of paintings each artists created, which can be a firm limit, especially when the painter is no longer alive.

SVMnet aims at providing an effective solution for the numerous real-world situations in which the number of labeled image samples that can be used for training is limited. SVMnet utilizes the ability of an SVM to learn from a smaller number of samples compared to other machine learning approaches. The flexible structure of SVMnet allows it to learn directly from the pixel values, and to utilize different layers that correspond to the convolutional and fully connected layers in "conventional" deep neural networks. Like DCNNs, SVMnet does not require the design of specific algorithm for a specific image classification problem.

Complexity analysis shows that the training time for SVMnet scales more quickly with the number of input samples than DCNNs, suggesting that SVMnet might take substantial computational resources when trained using large datasets. However, SVMnet is designed for situations in which the labeled training set is relatively small, and therefore the training time might not be a practical obstacle in many real-world situations in which SVMnet can be used.

The underlying structure used to create SVMnet is very flexible, allowing other kinds of machine learning algorithms to be used rather than solely SVMs. Constructing the layers with classifiers such as random forests or logistic regression may result in better performance for some datasets. These layers can be mixed in the same model as well, i.e. using one layer of SVMs followed by a layer of random forests. These possibilities present a promising avenue for future related work.

SVMnet is not designed to become a general solution that can outperform deep convolutional neural networks such as ResNet-50, but experimental results show that it is an effective solution for cases in which the number of labeled training samples is small. Since such cases are not rare, SVMnet can complement "conventional" deep neural networks by providing image classification in the cases where not many labeled training samples are available.

ACKNOWLEDGEMENTS

The research was funded by NSF grant number XXXXXX.

REFERENCES

Léon Bottou and Chih-Jen Lin. Support vector machine solvers. *Large scale kernel machines*, 3(1): 301–320, 2007.

Evgeny Byvatov, Uli Fechner, Jens Sadowski, and Gisbert Schneider. Comparison of support vector machine and artificial neural network systems for drug/nondrug classification. *Journal of chemical information and computer sciences*, 43(6):1882–1889, 2003.

Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

Wun-Hwa Chen, Sheng-Hsun Hsu, and Hwang-Pin Shen. Application of svm and ann for intrusion detection. *Computers & Operations Research*, 32(10):2617–2634, 2005.

Muhammad E. H. Chowdhury, Tawsifur Rahman, Amith Khandakar, Rashid Mazhar, Muhammad Abdul Kadir, Zaid Bin Mahbub, Khandakar Reajul Islam, Muhammad Salman Khan, Atif Iqbal, Nasser Al Emadi, Mamun Bin Ibne Reaz, and Mohammad Tariqul Islam. Can ai help in screening viral and covid-19 pneumonia? *IEEE Access*, 8:132665–132676, 2020. doi: 10.1109/ACCESS.2020.3010287.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning, 2018.

Ezz El-Din Hemdan, Marwa A. Shouman, and Mohamed Esmail Karar. COVIDX-Net: A Framework of Deep Learning Classifiers to Diagnose COVID-19 in X-Ray Images. *arXiv e-prints*, art. arXiv:2003.11055, March 2020.

Cedric Gangloff, Sonia Rafi, Guillaume Bouzillé, Louis Soulat, and Marc Cuggia. Machine learning is the key to diagnose covid-19: a proof-of-concept study. *Scientific Reports*, 11(1):7166, Mar 2021. ISSN 2045-2322. doi: 10.1038/s41598-021-86735-9. URL https://doi.org/10.1038/s41598-021-86735-9.

Hunter Goddard and Lior Shamir. A catalog of broad morphology of pan-starrs galaxies based on deep learning. *The Astrophysical Journal Supplement Series*, 251(2):28, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Asifullah Khan, Anabia Sohail, Umme Zahoora, and Aqsa Saeed Qureshi. A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, 53(8):5455–5516, 2020.

Fahad Shahbaz Khan, Shida Beigpour, Joost Van de Weijer, and Michael Felsberg. Painting-91: a large scale database for computational painting categorization. *Machine Vision and Applications*, 25(6):1385–1397, 2014.

Wei Tse Li, Jiayan Ma, Neil Shende, Grant Castaneda, Jaideep Chakladar, Joseph C. Tsai, Lauren Apostol, Christine O. Honda, Jingyue Xu, Lindsay M. Wong, Tianyi Zhang, Abby Lee, Aditi Gnanasekar, Thomas K. Honda, Selena Z. Kuo, Michael Andrew Yu, Eric Y. Chang, Mahadevan " Raj" Rajasekaran, and Weg M. Ongkeko. Using machine learning of clinical data to diagnose covid-19: a systematic review and meta-analysis. *BMC Medical Informatics and Decision Making*, 20(1):247, Sep 2020. ISSN 1472-6947. doi: 10.1186/s12911-020-01266-z. URL https://doi.org/10.1186/s12911-020-01266-z.

Srinivas Mukkamala, Guadalupe Janoski, and Andrew Sung. Intrusion detection using neural networks and support vector machines. In *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290)*, volume 2, pp. 1702–1707. IEEE, 2002.

Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia object image library (coil-100). 1996.

Joana S Paiva, João Cardoso, and Tânia Pereira. Supervised learning methods for pathological arterial pulse wave differentiation: a svm and neural networks approach. *International Journal of Medical Informatics*, 109:30–38, 2018.

Tawsifur Rahman, Amith Khandakar, Yazan Qiblawey, Anas Tahir, Serkan Kiranyaz, Saad Bin Abul Kashem, Mohammad Tariqul Islam, Somaya Al Maadeed, Susu M. Zughaier, Muhammad Salman Khan, and Muhammad E.H. Chowdhury. Exploring the effect of image enhancement techniques on covid-19 detection using chest x-ray images. *Computers in Biology and Medicine*, 132:104319, 2021. ISSN 0010-4825. doi: https://doi.org/10.1016/j.compbiomed.2021.104319. URL https://www.sciencedirect.com/science/article/pii/S001048252100113X.

Kyung-Shik Shin, Taik Soo Lee, and Hyun-jung Kim. An application of support vector machines in bankruptcy prediction model. *Expert Systems with Applications*, 28(1):127–135, 2005.