CIFAR-10: KNN-based Ensemble of Classifiers

Conference Paper · December 2016

DOI: 10.1109/CSCI.2016.0225

4 authors, including:

Yehya Abouelnaga
The American University in Cairo
3 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Hager Radi
The American University in Cairo
1 PUBLICATION **0** CITATIONS
SEE PROFILE

Mohamed N Moustafa
The American University in Cairo
46 PUBLICATIONS **186** CITATIONS
SEE PROFILE

Accepted in the 2016 International Conference on Computational Science and Computational Intelligence (Las Vegas, USA)

CIFAR-10: KNN-based Ensemble of Classifiers

Yehya Abouelnaga, Ola S. Ali, Hager Rady, and Mohamed Moustafa
Department of Computer Science and Engineering, School of Sciences and Engineering
The American University in Cairo, New Cairo 11835, Egypt
{ devyhia , olasalem1 , hagerradi , m.moustafa }@aucegypt.edu

***Abstract*—In this paper, we study the performance of different classifiers on the CIFAR-10 dataset, and build an ensemble of classifiers to reach a better performance. We show that, on CIFAR-10, K-Nearest Neighbors (KNN) and Convolutional Neural Network (CNN), on some classes, are mutually exclusive, thus yield in higher accuracy when combined. We reduce KNN overfitting using Principal Component Analysis (PCA), and ensemble it with a CNN to increase its accuracy. Our approach improves our best CNN model from 93.33% to 94.03%.**

***Keywords*—Ensemble of Classifiers; K-Nearest Neighbors; Convolutional Neural Networks; Principal Component Analysis**

I. Introduction

CIFAR-10 is a multi-class dataset consisting of 60,000 32×32 colour images in 10 classes, with 6,000 images per class. There are 50,000 training images and 10,000 test images [1]. In this paper, we explore different learning classifiers for the image-based multi-class problem. We will begin with training simple classifiers (like Logisitc Regression and Bayesian), and incrementy move towards more complex alternatives: Support Vector Machines, Decision Trees, Random Forrests, Gradient Boosting and K-Nearest Neighbours. Eventually, we will train a Deep Convolutional Neural Network (CNN). We will also explore various feature engineering appraoches like Principal Component Analysis (PCA). In an attempt to improve the state-of-the-art accuracy, we will devise an ensemble of classifiers. We will compare the performance of the previously mentioned classifiers, feature extractors, and their effect on the final ensemble.

CIFAR-10 presents a challenging classification problem. 32×32 images don't contain enough information for most classifiers to draw clear decision boundaries. A clear example of this is the confusion between "cat" and "dog" classes. The images objects are different in scale, rotation, position, and background. Some of the images are very unclear and hard to classify (even for human beings [2]).

II. Literature Review

Neural networks are widely used in solving image recognition problems. There is a wide variety of architectures that serve different purposes. Some publications aim at simple architectures to achieve decent results. [3] presents a shallow neural network that is, unlike multi-layered (i.e. deep) architectures, fast to train and more suitable for realtime applications. Their network achieved 75.86% test accuracy.

[4], [5] and [6] optimize their networks to learn better representation of features. [4] presents a simple architecture (PCANet) where layers of PCA is used to learn features (rather than using convolutional layers). PCANet achieves 78.67% test accuracy. [5] suggests using K-means (unsupervised learning) to learn better feature representations to transform the image space into a linearly separable feature space to be used with a standard linear classification algorithm (e.g. SVM). [6] aims at learning features by training a convolutional neural network using only unlabelled data.

[7], [8], [9] and [10] experiment with network pooling. [7], [8] and [9] suggest regulazing existing pooling functions. [7] proposes a flexible parameterization of the spatial pooling step and learn the pooling regions together with the classifier. [8] replaces the conventional deterministic pooling operations with a stochastic procedure (randomly picking the activation within each pooling region). [9] has formulated a fractional (stochastic) version of maxpooling (where non-integer multiplicative factors are allowed). This helps reduce overfitting, and achives state-of-the-art accuracy (with 96.53% test accuracy). [10] learns new pooling functions by combination of max and average pooling functions, or tree-structured fusion of pooling filters.

[11], [12], [13], [14], [15] and [16] optimize network activation functions. [11] presents a new randomized leaky rectified linear units (RReLU). [12] designed a novel form of piecewise linear activation function that is learned independently for each neuron using gradient descent (and outperforms rectified linear units). [13] introduce an exponential linear unit (ELU) which speeds up learning in deep neural networks and leads to higher classification accuracies. [14] defines a simple new model called *maxout* to improve the accuracy of *dropout* (see [17]) fast approximate model averaging technique. [15] present a probabilistic variant (*probout*) of the recently introduced *maxout* unit to improve its invariance properties. [16] shows that replacing the softmax layer with a linear support vector machine (SVM) consistently improves accuracies.

[18] introduces a novel deep network structure called "Network In Network" (NIN) to enhance model discriminability for local patches within the receptive field. [19] combines Network in Network architecture (NIN) (see [18]) and the maxout units (see [14]) to enhance model discriminability and facilitate the process of information abstraction within the receptive field.

[20] proposes a deep neural network architecture for object recognition based on recurrent neural networks (ReNet). The proposed network replaces the ubiquitous convolution+pooling layer of the deep convolutional neural network with four recurrent neural networks that sweep horizontally and vertically in both directions across the image. [21] proposes a recurrent CNN (RCNN) for object recognition by incorporating recurrent connections into each convolutional layer to enhance the ability of the model to integrate the context information (which is important for object recognition).

[22] introduces a novel architecture that decreases depth and increases width of residual networks. Wide residual networks (WRNs) are superior to their deep residuals couterparts. [23] proposes a simple method for weight initialization for deep net learning, called Layer-sequential unit-variance (LSUV) initialization to improve test accuracies.

The previous publications aim at improving neural networks as general learning tools. They research new activation functions, pooling functions, weight initialization methods, learning algorithms, and variations of existing models. Ensemble of classifiers are proven to achieve better results in the literature (see [24], [25], [26]). Despite their robustness, variations of CNN-based ensembles are not thoroughly researched and analyzed. In an attempt to explore more variations of ensembles, we analyze the impact of ensembling a simple learning tool like K-Nearest Neighbours (KNN) with Convolutional Neural Networks (CNN).

III. Proposed Method

We propose an ensemble-based approach to improve the CIFAR-10 test accuracy. We experiment with possible learning models, and try to find the best combination of classifiers to reduce classifier confusion and improve accuracy. We found that K-Nearest Neighbours (KNN) consistently improves the accuracy of Convolutional Neural Networks (CNN).

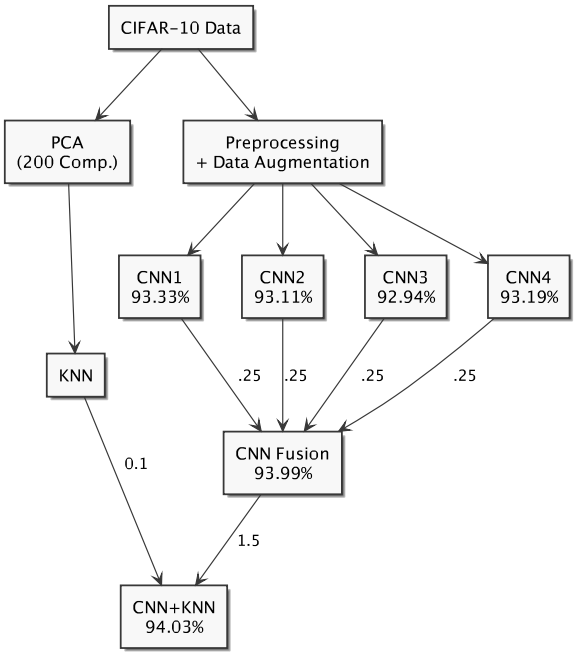

Fig. 1. Architecture of ensembling 4 ConvNets and KNN. PCA improves the accuracy of KNN as it reduces overfitting. Ensembling the 4 ConvNets improves their accuracy to 93.99%. KNN improves the accuracy of the ensemble to 94.03%.

A. Principal Component Analysis (PCA)

PCA is a dimensionality reduction tool used to remove noise from images. This behaviour is very clear when using K-Nearest Neighbours (KNN) for classification. The lower the number of components, the more accurate the distance

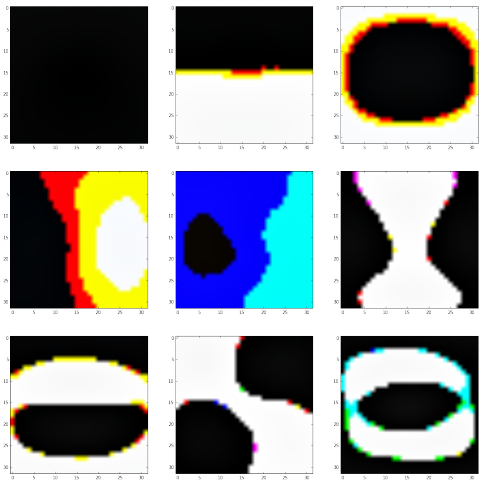

Fig. 2. The first 9 components in the PCA (of 200 components) preserve 65.5% of the data.

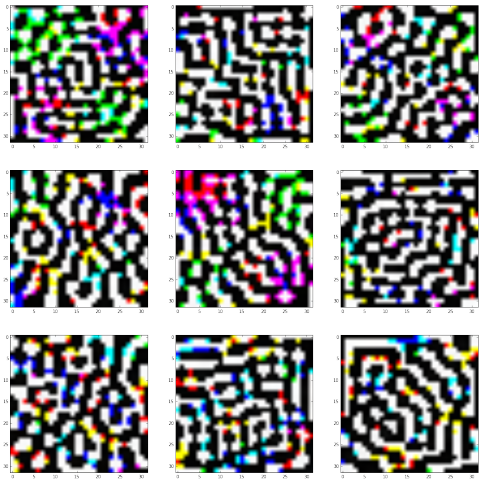

Fig. 3. The last 9 components in the PCA (of 200 components) preserve 0.28% of the data.

function performs. Throughout this paper, PCA will provide performance gains in almost all classifiers.

B. K-Nearest Neighbours

K-Nearest Neighbours (KNN) is a simple non-parametric classification method that depends on the between-sample geometric distance. It finds the best k means, $M_1, ..., M_k$, of the training sample representing the entire sample, and classifies every new point X_i based on the closest distance, D, to the means.

$$C(X_i) = \underset{k}{\operatorname{argmin}}\ D(X_i, M_k)$$

The simplicity of the KNN model rectifies major confusion between similar classes (i.e. cats, dogs and horses).

C. Convolutional Neural Network (CNN)

We used the best available model (see [27]). Models from other publications with higher accuracies were either not publicly available or not training properly. The architecture of

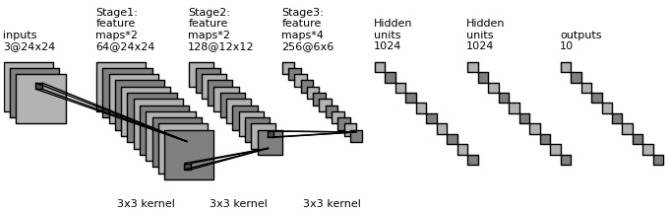

Fig. 4. Convolutional Neural Network Architecture.

TABLE I. CONVOLUTIONAL NEURAL NETWORK ARCHITECTURE

Input
2 x (Conv. + ReLU) kernel: 3x3, channel: 64, padding: 1
Max Pooling (kernel: 2x2, stride: 2) Dropout (rate: 0.25)
2 x (Conv. + ReLU) kernel: 3x3, channel: 128, padding: 1
Max Pooling (kernel: 2x2, stride: 2) Dropout (rate: 0.25)
4 x (Conv. + ReLU) kernel: 3x3, channel: 256, padding: 1
Max Pooling (kernel: 2x2, stride: 2) Dropout (rate: 0.25)
Linear (channel: 1024) + ReLU
Dropout (rate: 0.5)
Linear (channel: 1024) + ReLU
Dropout (rate: 0.5)
Linear (channel: 10)
Softmax

our deep neural network consists of 8 convolutional layers in addition to 3 linear layers (see Table I). It achieves, on average, a test accuracy of 93.13%. We trained the model 4 times with different initial seeds and averaged them to produce 93.99% test accuracy.

Data augmentation has been used to artificially enlarge the size of the dataset and reduce the effect of over fitting. We used cropping, horizontal reflection (similar to [28]) and scaling. All three methods allow the transformed image to be generated from the original image with little computation. As for preprocessing, we used Global Contrast Normalization (GCN) and ZCA whitening.

D. Ensemble Weight Estimation

[24], [29], [25], [26] propose multiple approaches for ensemble parameter estimation in a weighted voting system. However, we opted out for a simple exhaustive search. Assume, $C_1, C_2, ..., C_n$, are experts. We define a sequence of possible weights, $W_{n+1} = W_n + S$ and $W_0 = 0$, where S is a step between two consecutive weights. Let $R = \{W_0, W_1, ..., W_k\}$, where k is the number of possible weights.

$$E(C_1, C_2) = \underset{w_i, w_j \in R}{\text{argmax}} \; w_i \times C_1 + w_j \times C_2$$

We estimate all parameters in a chain rule style (i.e. $E(E(C_1, C_2), C_3)$).

IV. EXPERIMENTS

A. Feature Extractors

We experimented with Principal Component Analysis (PCA), Scale-Invariant Feature Transform (SIFT), Speeded Up Robust Features (SURF), Oriented Fast and Rotated Brief (ORB), and Histogram of Oriented Gradients (HOG). All, except PCA, overfitted the model. They increased the training accuracy but lowered the test accuracy. For most classifiers (i.e. Logistic Regression, Decision Trees, Random Forests, and Gradient Boosting), we found that 200 PCA components (out of 3,072) reduce overfitting and increase test accuracy. For K-Nearest Neighbours, the best test accuracy was acheived by a lower number of components (30 components).

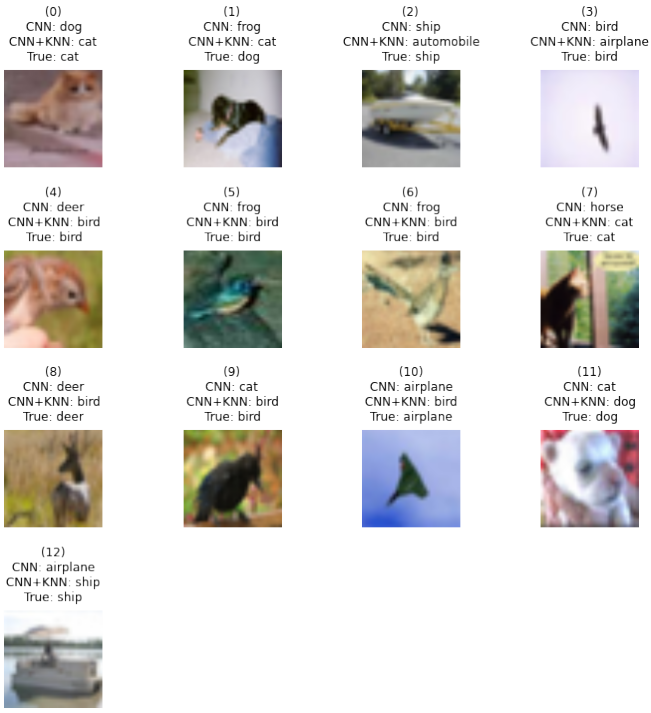

Fig. 5. Effect of K-Nearest Neighbours (KNN) on our Convolutional Neural Network (CNN) model.

B. Results Analysis

In Fig. 5, we study the difference between the CNN and KNN. In images (0, 4, 5, 6, 7, 9, 11, 12), we find that KNN's vote in the classification decision improves CNN's insensitivity to shape. It improved the confusion between cat, dog, and horse classes. In image (1), it also predicted a dog as a cat (which is closer than a frog). In image (2), it didn't generalize very well. CNN voted for a ship. CNN+KNN voted for an automobile. However, the image contains a ship on top of a vehicle (with two tires). The shape sensitivity of the KNN vote helped reduce confusion among cat, horse, dog, and bird classes. However, it confused the CNN on images (3, 8, 10), where airplane was confused for a bird (and vice versa), and a deer confused for a bird (due to its posture).

V. CONCLUSION

K-Nearest Neighbors (KNN) is a simple classification algorithm based on geometric distance. On CIFAR-10 dataset, we showed that it stabilizes the decision of other Convolutional Neural Networks (CNN) due to its shape sensitivity. We showed that an ensemble of CNNs and KNN improves the accuracy of the model. More research is to be done on the

TABLE II. EXPERIMENTAL RESULTS

Classifier	Accuracy (%)
Log. Reg. + 3,072 Features	37.5
Log. Reg. + 50 PCA Comp.	37.69
Log. Reg. + 100 PCA Comp.	40.13
Log. Reg. + 150 PCA Comp.	40.18
Log. Reg. + 200 PCA Comp.	**41.04**
Log. Reg. + 225 PCA Comp.	40.56
Log. Reg. + 250 PCA Comp.	40.87
KNN + 3,072 Features	33.86
KNN + 200 PCA Comp.	36.54
KNN + 75 PCA Comp.	39.77
KNN + 50 PCA Comp.	40.12
KNN + 40 PCA Comp.	40.93
KNN + 30 PCA Comp.	**41.78**
KNN + 25 PCA Comp.	41.57
KNN + 15 PCA Comp.	38.75
KNN + 10 PCA Comp.	34.93
RFC/512	49.26
RFC/1024	48.97
RFC/512 + 200 PCA Comp.	48.59
RFC/1024 + 200 PCA Comp.	**49.52**
GRB + 3,072 Features	**47.78**
SVM + 3,072 Features	**49.88**
CNN1 + Data Augm.	93.33
CNN2 + Data Augm.	93.11
CNN3 + Data Augm.	92.94
CNN4 + Data Augm.	93.19
CNN Fusion	93.99
CNN Fusion + KNN	**94.03**

TABLE III. ENSEMBLING RESULTS

	Base Line	KNN	GRB	RFC
CNN1	93.33	**93.46**	93.38	93.40
CNN2	93.11	93.15	93.15	**93.16**
CNN3	92.94	92.97	**92.98**	**92.98**
CNN4	93.19	**93.25**	93.19	93.21
CNN Fusion	93.99	**94.03**	94.00	93.99

effect of KNN on CNN. A KNN needs to be checked against other datasets to generalize a statement on the effect of KNN. Other weight estimation methods should be evaluated and compared to the simple exhaustive search we presented in this paper.

ACKNOWLEDGMENT

The authors relied on the implementation of Scikit Learn Python Library [30] and Torch in most of the experiments carried out in this paper.

REFERENCES

[1] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images." 2009.

[2] A. Karpathy, "Lessons learned from manually classifying CIFAR-10," 2011. [Online]. Available: http://karpathy.github.io/2011/04/27/manually-classifying-cifar10/

[3] M. D. McDonnell and T. Vladusich, "Enhanced image classification with a fast-learning shallow convolutional neural network," in *2015 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2015, pp. 1–7.

[4] T.-H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma, "PCANet: A simple deep learning baseline for image classification?" *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5017–5032, 2015.

[5] A. Coates and A. Y. Ng, "Learning feature representations with k-means," in *Neural Networks: Tricks of the Trade*. Springer, 2012, pp. 561–580.

[6] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2014, pp. 766–774.

[7] M. Malinowski and M. Fritz, "Learning smooth pooling regions for visual recognition," in *24th British Machine Vision Conference*. BMVA Press, 2013, pp. 1–11.

[8] M. D. Zeiler and R. Fergus, "Stochastic pooling for regularization of deep convolutional neural networks," *arXiv preprint arXiv:1301.3557*, 2013.

[9] B. Graham, "Fractional max-pooling," *arXiv preprint arXiv:1412.6071*, 2014.

[10] C.-Y. Lee, P. W. Gallagher, and Z. Tu, "Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree," in *International Conference on Artificial Intelligence and Statistics*, 2016.

[11] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.

[12] F. Agostinelli, M. Hoffman, P. Sadowski, and P. Baldi, "Learning activation functions to improve deep neural networks," *arXiv preprint arXiv:1412.6830*, 2014.

[13] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[14] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. C. Courville, and Y. Bengio, "Maxout Networks," *ICML (3)*, vol. 28, pp. 1319–1327, 2013.

[15] J. T. Springenberg and M. Riedmiller, "Improving deep neural networks with probabilistic maxout units," *arXiv preprint arXiv:1312.6116*, 2013.

[16] Y. Tang, "Deep learning using linear support vector machines," *arXiv preprint arXiv:1306.0239*, 2013.

[17] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[18] M. Lin, Q. Chen, and S. Yan, "Network in Network," *arXiv preprint arXiv:1312.4400*, 2013.

[19] J.-R. Chang and Y.-S. Chen, "Batch-normalized maxout network in network," *arXiv preprint arXiv:1511.02583*, 2015.

[20] F. Visin, K. Kastner, K. Cho, M. Matteucci, A. Courville, and Y. Bengio, "ReNet: A recurrent neural network based alternative to convolutional networks," *arXiv preprint arXiv:1505.00393*, 2015.

[21] M. Liang and X. Hu, "Recurrent convolutional neural network for object recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3367–3375.

[22] S. Zagoruyko and N. Komodakis, "Wide Residual Networks," *arXiv preprint arXiv:1605.07146*, 2016.

[23] D. Mishkin and J. Matas, "All you need is a good init," *ICLR*, 2015.

[24] D. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," *Journal of Artificial Intelligence Research*, vol. 11, pp. 169–198, 1999.

[25] A. Rahman and S. Tasnim, "Ensemble Classifiers and Their Applications: A Review," *International Journal of Computer Trends and Technology (IJCTT)*, vol. 10, no. 1, 2014.

[26] T. G. Dietterich, "Ensemble methods in machine learning," *International workshop on multiple classifier systems*, p. Springer.

[27] Nagadomi, "Kaggle CIFAR-10," 2014. [Online]. Available: https://github.com/nagadomi/kaggle-cifar10-torch7

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[29] H. Kim, H. Kim, H. Moon, and H. Ahn, "A weight-adjusted voting algorithm for ensembles of classifiers," *Journal of the Korean Statistical Society*, vol. 40, no. 4, pp. 437—-449, 2011.

[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825—-2830, 2011.