Difficult Handwritten Digit Classification

Code available at: `https://github.com/slflmm/Miniproject-3`

COMP 598 – Miniproject 3 – Team X

Narges Aghakazemjourabbaf
?

Stephanie Laflamme
260376691

Benjamin La Schiazza
?

ABSTRACT

1. INTRODUCTION

2. PREPROCESSING

Standardization (x-mean/standard deviation) Contrast normalization

3. FEATURE DESIGN AND SELECTION

When appropriate for the learner, we consider three feature sets; raw pixels, PCA, and Gabor filter-based features.

3.1 Pixels

We use the post-standardization pixel information as a baseline feature set.This produces feature vectors of length 2304.

3.2 PCA

(Describe PCA)

As removing the least useful features made the results of our baseline classifier worse, we keep the same initial number of dimensions.

3.3 Gabor

[1]

4. ALGORITHM SELECTION

4.1 Perceptron

4.2 Neural Network

4.3 Linear SVM

Use Scikit-learn implementation [9]

4.4 Convolutional Neural Network

Origin of convnets [5]. We can see it has good invariance to rotation and resistance to noise with the MNIST dataset. [8]

Problem of neural networks = overfitting. Regularization... (L1 + L2 norms). Dropout [6] Dropout in fully-connected layers of a convolutional net [7] SGD with minibatches. Momentum.

5. OPTIMIZATION

GPU for convolutional network – Theano [3]

6. PARAMETER SELECTION METHOD

Preliminary manual search followed by either:

Gridsearch

Random search [2]

7. TESTING AND VALIDATION

7.1 Perceptron

In a primary step, we performed 5-fold cross-validation to compare raw pixels, PCA, and Gabor features using a fixed learning rate ($\alpha = 0.01$) and number of iterations (15). As shown in Table 1, the most performant feature set was ???, with a validation accuracy of ????.

Features	Pixels	PCA	Gabor
Accuracy	25.794	26.268	10

Table 1: Mean validation accuracy of perceptron using different features

Using (the best feature set), we performed 5-fold cross-validation over the learning rate α and the number of training iterations of the perceptron model. Figure 1 shows the results of gridsearch with both parameters. The precise values of the best parameters found by the gridsearch cross-validation procedure were $\alpha = 0.0005$ with 20 training iterations, yielding a mean validation accuracy of 26.594%.[1]

After training our perceptron on the complete training set using these parameters, we submitted our results to Kaggle and obtained a test accuracy of ???. As an approximation of the test confusion matrix, we provide the confusion matrix for the combined validation sets in Figure 2. Note the perceptron's greater ability to identify 0s and 1s compared to other digits.

[1] Additional results showing training error vs validation error are shown in Appendix 1

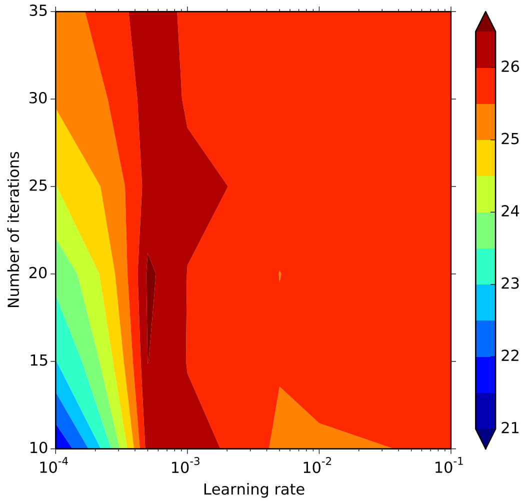

Figure 1: Mean cross-validation accuracy as a function of parameters α and number of iterations

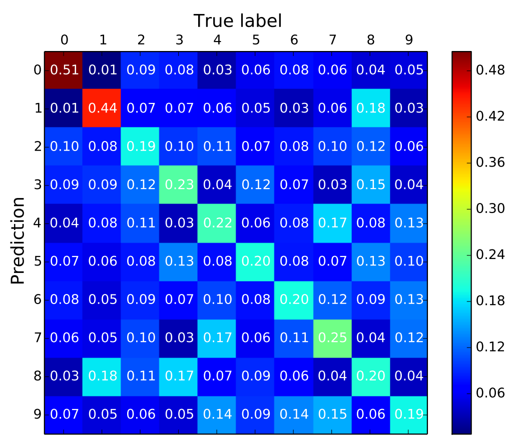

Figure 2: Validation confusion matrix for perceptron

8. DISCUSSION

Why is it better at classifying 0s and 1s? (Check number of examples in classes–could be Benford's law?

Using Gabor filters as a kernel rather than feature [11]

Other version of dropout [13]

Pretraining [4]

Others: [10], [12]

We hereby state that all the work presented in this report is that of the authors.

9. REFERENCES

[1] T. C. Bau. *Using Two-Dimensional Gabor Filters for Handwritten Digit Recognition*. PhD thesis, M. Sc. thesis, University of California, Irvine.

[2] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, 13(1):281–305, 2012.

[3] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010. Oral Presentation.

[4] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *The Journal of Machine Learning Research*, 11:625–660, 2010.

[5] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

[6] 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.

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

[8] Y. LeCun. Lenet-5, convolutional neural networks. `http://yann.lecun.com/exdb/lenet/index.html`, 2004. Accessed: 2014-10-22.

[9] 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 E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[10] H. A. Rowley, S. Baluja, and T. Kanade. Rotation invariant neural network-based face detection. In *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, pages 38–44. IEEE, 1998.

[11] M. Sabri and P. Fieguth. A new gabor filter based kernel for texture classification with svm. In *Image Analysis and Recognition*, pages 314–322. Springer, 2004.

[12] P. Y. Simard, Y. A. LeCun, J. S. Denker, and B. Victorri. Transformation invariance in pattern recognition–tangent distance and tangent propagation. In *Neural networks: tricks of the trade*, pages 235–269. Springer, 2012.

[13] L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus. Regularization of neural networks using dropconnect. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1058–1066, 2013.

APPENDIX

A. ADDITIONAL RESULTS

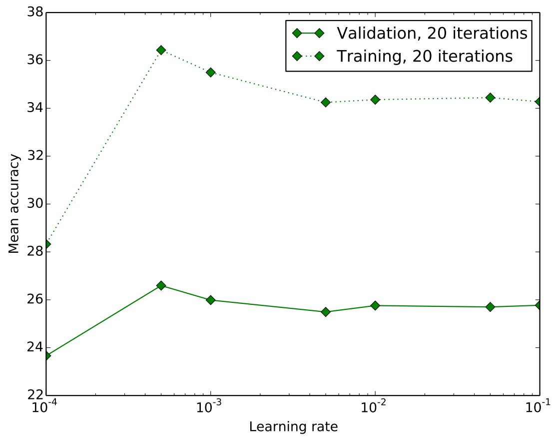

Figure 3: Cross-validation over α with perceptron, keeping # iterations optimal

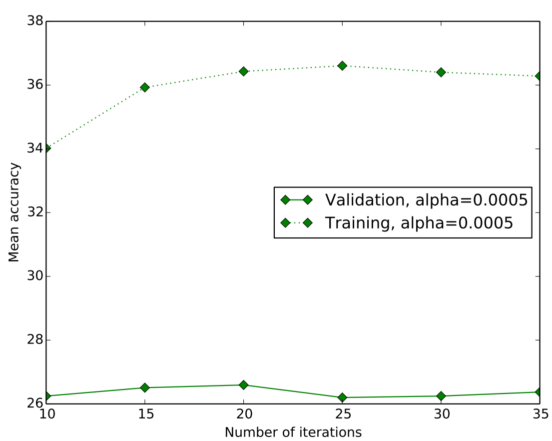

Figure 4: Cross-validation over # of iterations with perceptron, keeping α optimal