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

The raw pixels are standardized. For each example i in the training set, and each feature j, we replace it with its standardized value

$$x'_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}.$$

These values for μ_j and σ_j are used to apply the same transformation to the test set examples.

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

Principal component analysis (PCA), developed by Pearson in the early 1900s[12], converts a set of possibly correlated features into a linearly uncorrelated representation, with the resulting features ordered by variance.

We use the implementation of PCA provided by the Scikit-learn library[13], which uses singular value decomposition of the input data matrix.

As removing the least useful features made the results of our baseline classifier worse, we keep full dimensionality. However, we expect PCA features to produce better results given that they are linearly uncorrelated.

3.3 Gabor

Gabor filters are linear filters used for edge detection and are thought to be similar to the early stages of visual processing in humans. Indeed, their frequency and orientations correspond roughly to simple cells in the visual cortex of humans and other mammals; these cells can be modelled with Gabor functions.[8][11] As these filters are well-suited to image processing, we attempt to translate them into features.

Previous research has used the energy of the convolution between a Gabor filters and image (a measure of how strongly the filter responds to that image[6]) as a feature by summing its magnitudes in the image.[1] This is equivalent to using the Frobenius norm of the convolved image as a feature.

Using the Scikit-learn library[13], we generate 16 Gabor filters with 4 equidistant θ values, $\sigma = \{1, 3\}$, and frequencies $= \{0.05, 0.25\}$. Figure 1 illustrates the effects of θ, σ, and frequency with a sample of our filters. We form the feature vector of an image in the dataset by collecting the Frobenius norms of the image convolved with each filter.

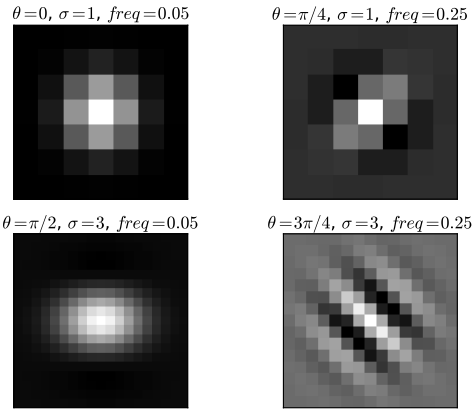

Figure 1: Some Gabor filters

4. ALGORITHM SELECTION

4.1 Perceptron

We implement a multiclass perceptron as a baseline classifier. Given n examples with m features each, the perceptron

4.2 Neural Network

4.3 Linear SVM

Use Scikit-learn implementation [13]

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. [10]

Problem of neural networks = overfitting. Regularization... (L1 + L2 norms). Dropout [7] Dropout in fully-connected layers of a convolutional net [9] 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, PCA features were the most performant, with a validation accuracy of 26.282%.

Features	Pixels	PCA	Gabor
Accuracy	25.794	26.282	10.398

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 2 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 25 training iterations, yielding a mean validation accuracy of 26.598%.[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 27.420%. As an approximation of the test confusion matrix, we provide the confusion matrix for the combined validation sets in Figure 3. Note the perceptron's greater ability to identify 0s and 1s compared to other digits.

7.2 Neural Network

Try a baseline net with the three feature sets, compare (5-fold cross-validation).

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

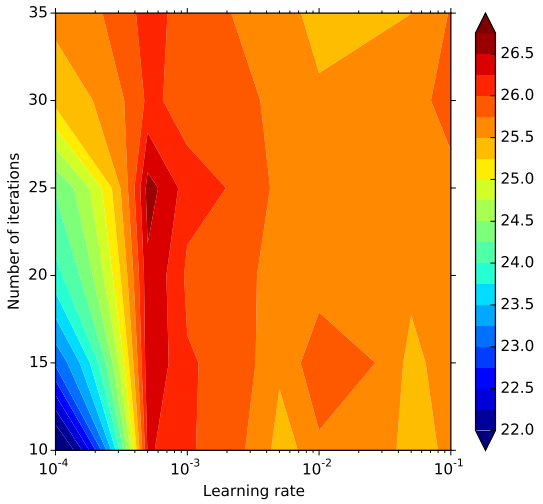

Figure 2: Mean cross-validation accuracy as a function of parameters α and number of iterations

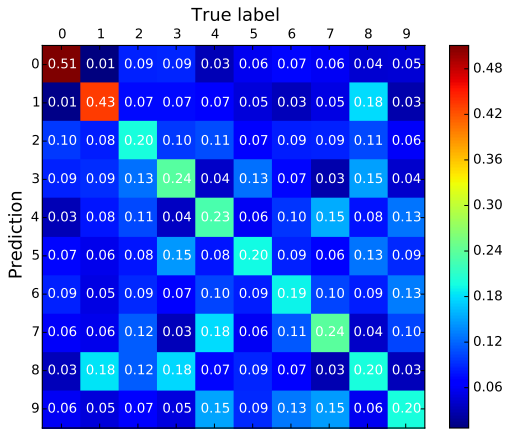

Figure 3: Validation confusion matrix for perceptron

Use the best feature set to find best hyperparameters with cross-validation. You should probably try random search rather than gridsearch (faster and just as good or better), for some set number of configurations. Use some graph to show the hyperparameter space results. Plot validation confusion matrix. Say the test set (kaggle) result using best model found during crossval.

7.3 Linear SVM

7.4 Convolutional Neural Network

Only appropriate features = raw pixels. Applied contrast normalization on top of standardization.

Due to limited time, we can only sample a small sample of the hyperparameter space.

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 [15]

Other version of dropout [17]

Pretraining [4]

Others: [14], [16]

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] S. E. Grigorescu, N. Petkov, and P. Kruizinga. Comparison of texture features based on gabor filters. *Image Processing, IEEE Transactions on*, 11(10):1160–1167, 2002.

[7] 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.

[8] J. P. Jones and L. A. Palmer. An evaluation of the two-dimensional gabor filter model of simple receptive fields in cat striate cortex. *Journal of neurophysiology*, 58(6):1233–1258, 1987.

[9] 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.

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

[11] S. Marčelja. Mathematical description of the responses of simple cortical cells*. *JOSA*, 70(11):1297–1300, 1980.

[12] K. Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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

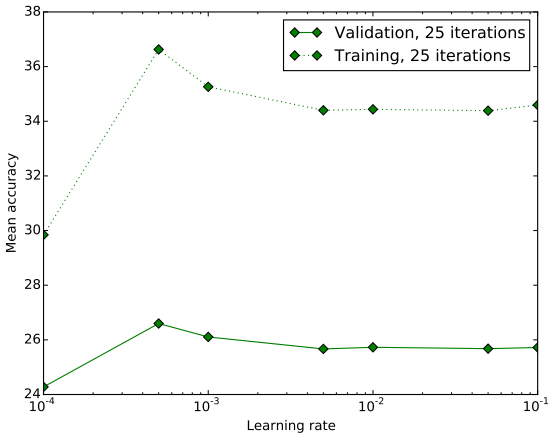

Figure 4: Cross-validation over α with perceptron, keeping # iterations optimal

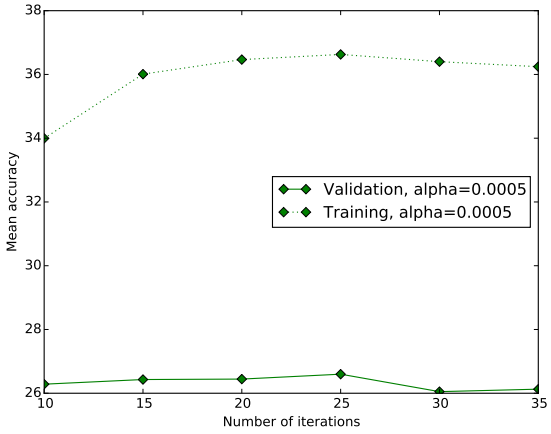

Figure 5: Cross-validation over # of iterations with perceptron, keeping α optimal