Neural pairwise classification models created by ignoring irrelevant alternatives

Ondrej Šuch[1,2], Martin Kontšek[2], and Andrea Tinajová[1]

[1] Mathematical Institue, Slovak Academy of Sciences
955 01 Banská Bystrica, Slovakia
ondrejs@savbb.sk,
[2] Žilinská Univerzita v Žiline, Univerzitná 8215/1,
010 26 Žilina

Abstract: It is possible to construct multiclass classification models from binary classifiers trained in pairwise (one-on-one) manner. Important examples of models created in this way are support vector machines applied to multiclass problems. In this work we examine feasibility of this approach for convolutional neural networks. We examine multiple ways to train pairwise classification networks for MNIST dataset, and multiple ways to combine them into a multiclass classifier for MNIST classification problem. Our experimental results show definite promise of this approach, especially in reducing complexity of deep neural networks. An important unresolved question of our approach is how to choose the best pairwise network to include into a full multi-class model.

Keywords: MNIST, convolutional network, pairwise coupling, one-on-one classification, binary classification, dropout

1 Introduction

Deep neural networks are currently the most powerful type of classifiers applicable for a multitude of machine learning problems [1]. Perhaps their biggest drawback is their complexity, which manifests in multiple ways. First, they require preparation and use of large datasets to attain the best precision [2]. They need a lot of specialized computing power for training [3] and the training process may take a long time. Finally, the classification process is obscured by their complexity, which makes it harder to understand their weaknesses and guarantee performance on unseen instances. In this article we consider the question, whether the classification process using deep neural networks could be made more modular, alleviating the drawbacks resulting from the complexity.

The approach is inspired by research on support vector machines. Support vector machines were proposed by Vapnik as a general purpose classifier [4]. They are still popular to this date for a variety of classification tasks. Since SVM work by dividing feature space (or its embedding into a higher-dimensional space [5]) into two parts by a hyperplane, they are naturally suited for two-class problems. Multi-class classification with SVM is accomplished by training SVM for pairs of classes, then fitting sigmoid to obtain pairwise probabilities [6] and finally using a pairwise coupling method to obtain multiclass prediction probabilities [7].

The same approach can be applied to deep neural networks. An additional simplification compared to SVM is that the step of fitting sigmoid is not necessary, since neural networks typically use soft-max for their final layers that directly output prediction probabilities. On the other hand, compared to SVM, the process of training of neural networks allows for many more hyperparameters.

In this paper we will carry out the process on MNIST digit classification task [8] illustrating the potential and possible pitfalls of the approach (Figure 1). Even for the basic MNIST task we had to severely limit the number of investigated training procedures. A key restriction we adopted is that we trained the two-class networks only with examples belonging to the corresponding two classes. Such approach, dubbed 'ignoring irrelevant alternatives', promises to speed up the training process for the two-class networks by reducing the size of the training dataset as well as to cut the time needed to train the whole multi-class model. Moreover, it is philosophically consistent with the assumption of the independence of irrelevant alternatives in the softmax layer commonly used in neural networks.

Our restriction, and pairwise decomposition of multiclass classification itself are not without potential problems. A key issue is that of extrapolating prediction probabilities to classes that a two-class classifier has never seen during training (e.g. the insight of G. Hinton described in the work of Hastie and Tibshirani [9]). Another potential problem to guard against is that the proposed classification scheme may require much more parameters since as much as 10 x 9 / 2 = 45 neural networks need to be trained instead of one.

2 Methodology outline

MNIST dataset of handwritten digits (zero to nine) is a widely used benchmark task in which convolutional networks proved quite successful [8]. It consists of 60000 training samples and 10000 testing samples. Throughout this work we will use 8-layer feed-forward networks de-

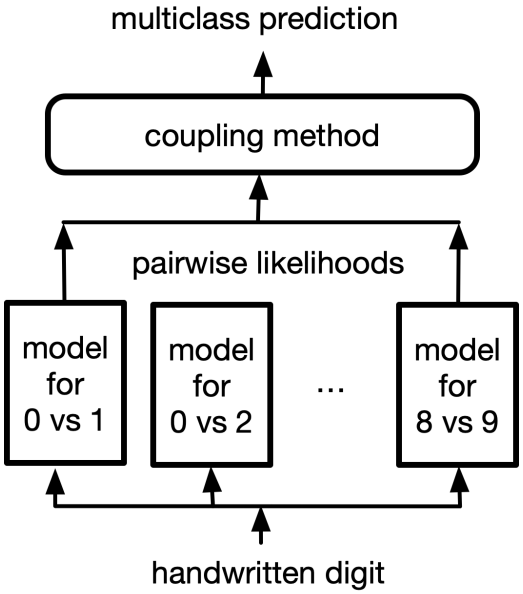

Figure 1: The schema of a classification system built using pairwise classifiers.

Layer name	**Layer type**	**Output shape**	**Number of parameters**
conv2d_1	convolutional	(?, 26,26, 32)	320
conv2d_2	convolutional	(?, 24,24,64)	18496
maxpool2d_1	max_pooling	(?, 12, 12, 64)	0
dropout_1	dropout	(?, 12, 12, 64)	0
flatten_1	flatten	(?, 9216)	0
dense_1	dense	(?, 128)	1179776
dropout_2	dropout	(?, 128)	0
dense_2	dense	(?, 10)	1290

Table 1: Structure of Keras network for MNIST classification

rived from the network structure of the sample network defined for solving MNIST classification in Keras framework [10]. The network has eight layers totaling 1,199,882 parameters. The underlying tensor shapes and number of parameters are given in the Table 1. The optimization criterion is crossentropy loss and the model is trained with Adadelta optimizer.

The networks we used differed from this one by changing the dropout probabilities in layers dropout_1 and dropout_2 and the number of kernels in conv2d_1, conv2d_2 and the number of neurons dense_1 layer. Moreover, since we use smaller training datasets, we compensated by increasing the number of training epochs to 24 from 12.

The network achieves about 99.13% average success rate classifying all digits (Fig. 2 left). The trained instance of the network can be viewed also as a two-class classifier for any pair of digits. The right part of Figure 2 summarizes measured test accuracy as a 2-7 classifier.

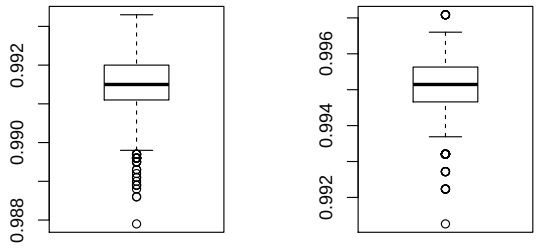

Figure 2: Boxplots summarizing the performance of the Keras network on MNIST task (left), and as a pairwise classifier or digits 2 vs. 7 (right).

For the coupling method we used the method of Wu-Lin-Weng [11], which is commonly applied in SVM libraries [7]. The method is nontrainable i.e. there are no adjustable parameters.

3 Models using large CNN

By a large CNN studied in this section we mean a convolutional network that differs from the Keras sample network only in the values of the dropout parameters in layers dropout_1 and dropout_2. Moreover, all large CNN networks were trained only on pairs of digits, and the number of training epochs was increased to 24.

3.1 Dependence on dropout parameters

Dropout layers serve to suppress overfitting [12]. The number of parameters of Keras neural networks greatly outnumbers the number of training samples, and without dropout it would be difficult to achieve high accuracy reliably. The optimal value of dropout probabilities may be different from the original Keras network, since a) the task has changed, and so did the training set, b) we increased the number of epochs. Therefore we systematically sampled dropout parameter space and measured mean performance of the network. The results are shown in Figure 3. Overall the performance did not vary too much. Moreover the Figure implies that larger values of dropout are appropriate for the first dropout layer and smaller values of dropout are appropriate for the the second dropout layer. Notable however is that training only on relevant examples i.e. samples of twos and sevens resulted in lower performance than obtained using original Keras network (Fig. 2 right) for any choice of dropout parameters.

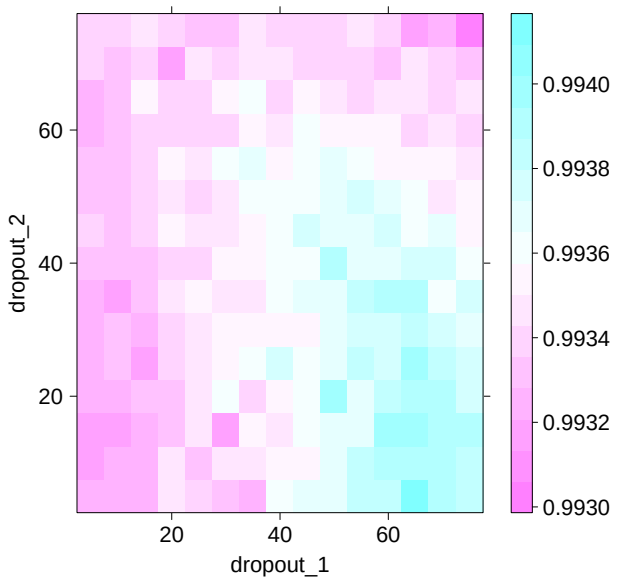

Figure 3: Measured performance of large CNN network on classifying 2's vs. 7's when the dropout probability of the two dropout layers was varied.

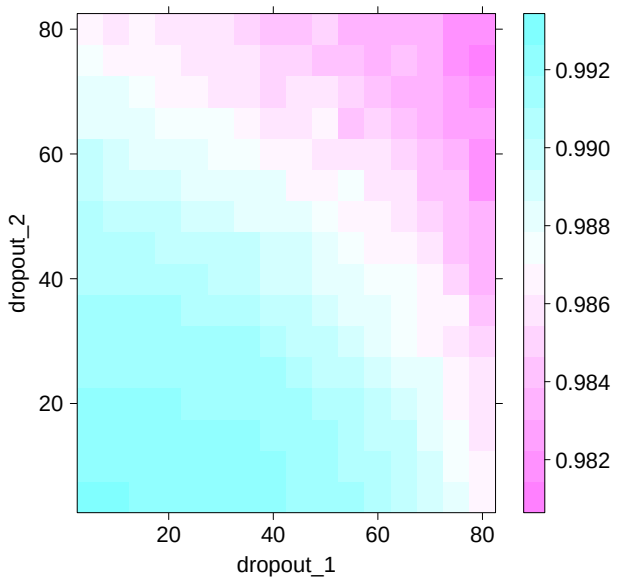

Figure 4: Measured performance of small CNN network on classifying 2's vs. 7's when the dropout probability of the two dropout layers was varied.

dropout probability in dropout_1 layer	dropout probability in dropout_2 layer	error-rate of the model on MNIST classification
0.25	0.50	0.95%
0.65	0.05	0.95 %
0.65	0.15	0.96 %
0.70	0.15	0.96 %

Table 2: Error rates of the complete pairwise models built using large CNN networks

3.2 Performance of complete pairwise models

Coupling models using to build complete pairwise models have built-in redudancy. This redundancy has been shown to ameliorate subpar performance of individual pairwise classifiers in simulations with synthetic data [11]. It is therefore worthwhile to evaluate the performance of complete pairwise models. We did so for several choices of dropout parameters: one for the values provided in the original Keras network and the rest for dropout values proved moderately better in classifying 2's vs. 7's (Figure 3). Only one experiment for each value was conducted, since the number of parameters of the complete model is impractcally large and the results are only for establishing a comparison benchmark. The results are shown in Table 2. We can surmise that the performance dropped slightly (by 0.1%) compared to the Keras network, despite the fact that at the same time the number of parameters increased 45-fold.

4 Models built using small CNN

By a small CNN studied in this section we mean a neural network that differs from the Keras network by having using 4 kernels at conv2d_1 layer (rather than 32), 4 kernels at conv2d_2 layers (rather than 64), 16 neurons at dense_1 layer (rather than 128), and possibly having a different dropout probabilities for layers dropout_1 and dropout_2. This ad hoc choice was made so that the network uses only 9,454 parameters, so a complete model built from such networks would use 425,430 parameters, which is about 35% of the size of the original Keras network. Again, all small CNN networks considered within this section were trained only on samples for pairs of digits and the number of epochs was increased to 24.

4.1 Dependence on dropout parameters

Since the small CNN networks have so few parameters one may hypothesize that dropout is not needed to prevent overfitting. In order to verify this hypothesis we again systematically explored varying dropout values as shown in Figure 4. The results are consistent with the hypothesis.

4.2 Average performance of pairwise models

In view of the dropout experiments further small networks in this section were trained without dropout. We created two series of series of sets of networks: (A) trained on all available training samples for corresponding digits, and (B) setting aside 10% of the training samples as a validation dataset, to be used for model selection in Section 4.3.

Each series consisted of 20 sets, and each set contained 45 pairwise networks corresponding to every pair of distinct digits. The mean error rates are shown in Table 3.

4.3 Performance of pairwise models

Having trained multiple sets of pairwise models, we proceeded to complete multi-class MNIST classification models using pairwise coupling. There are two distinct ways how to select pairwise networks into a complete model – we can take all networks from a single set, or we can choose for each pair of digits a model from various sets for which we expect the smallest error. There are again multiple ways to do the latter – we may choose based on its (pairwise) error rate, or its (pairwise) loss, and we may measure those on either the training set or the validation set. The results for various combinations of these paarameters are shown in Table 4.

From the table we can see that the best result was error rate 1.24% achieved by a model built from A series of networks which were trained on the full training set without setting aside a validation subset.

In order to contemplate the hypothesis that none of these selection criteria is reliable (e.g. if sever overfitting occurs on the train set), we can also compare selection based on the results on the test pairwise dataset. The results for this last choice are of course not indicative of real performance of a method, but could be considered as an estimate of an upper bound based on a hypothetical method for choosing the best individual pairwise classifiers. The results shown in Table 5 indicate that there would be significant improvement, which would rival the precision achieved by the original Keras network (see Fig. 1 left).

5 Discussion

Our experimental results are tantalizing. At the end of Section 4 we were able to exhibit neural pairwise classification models with weights trained only on training data from MNIST that show comparable performance to Keras classification network, yet needing only 35% of its weights. Moreover, they are modular and individual networks in the models can be trained in parallel. However, we are unable to provide an algorithm that would arrive at such classification models.

Similarly there are plenty of large CNN networks of the structure considered in this paper that top classification rate 99.5% on the problem of classifying twos from sevens in MNIST dataset. Yet, we were not able to find a combination of dropout hyperparameters that would yield the same performance, when training the network only on twos and sevens. The unsettling suggestion is that it is advantageous to use irrelevant examples for training deep convolutional neural networks.

It seems more attention should be paid to binary image classification problems with convolutional networks.

pair of digits	Series A. trained without validation subset		Series B. trained with validation subset		
	train	**test**	**train**	**validate**	**test**
0-1	0.02	0.06	0.01	0.13	0.06
0-2	0.03	0.29	0.03	0.26	0.26
0-3	0.04	0.10	0.01	0.22	0.11
0-4	0.02	0.09	0.03	0.17	0.09
0-5	0.02	0.27	0.04	0.23	0.24
0-6	0.13	0.44	0.12	0.37	0.44
0-7	0.02	0.25	0.01	0.16	0.28
0-8	0.09	0.45	0.11	0.38	0.52
0-9	0.13	0.27	0.12	0.34	0.32
1-2	0.08	0.29	0.08	0.31	0.29
1-3	0.03	0.13	0.03	0.18	0.11
1-4	0.09	0.02	0.10	0.29	0.04
1-5	0.01	0.13	0.01	0.15	0.13
1-6	0.02	0.18	0.02	0.11	0.21
1-7	0.16	0.25	0.16	0.31	0.24
1-8	0.13	0.18	0.12	0.40	0.18
1-9	0.09	0.24	0.11	0.24	0.24
2-3	0.08	0.29	0.05	0.47	0.29
2-4	0.03	0.28	0.03	0.21	0.29
2-5	0	0.09	0	0.17	0.08
2-6	0.01	0.32	0	0.15	0.32
2-7	0.12	0.72	0.14	0.45	0.76
2-8	0.08	0.43	0.07	0.45	0.40
2-9	0.03	0.36	0.02	0.26	0.35
3-4	0.01	0.12	0.02	0.17	0.13
3-5	0.12	0.53	0.09	0.57	0.51
3-6	0.01	0.07	0	0.12	0.09
3-7	0.05	0.41	0.17	0.45	0.56
3-8	0.05	0.35	0.09	0.55	0.42
3-9	0.10	0.45	0.09	0.55	0.42
4-5	0.00	0.04	0	0.18	0.03
4-6	0.02	0.28	0.04	0.26	0.29
4-7	0.03	0.16	0.07	0.32	0.18
4-8	0.05	0.20	0.09	0.36	0.25
4-9	0.25	0.77	0.20	0.79	0.72
5-6	0.19	0.63	0.13	0.41	0.57
5-7	0.00	0.14	0.00	0.19	0.11
5-8	0.12	0.36	0.14	0.63	0.42
5-9	0.08	0.35	0.07	0.45	0.30
6-7	0.00	0.13	0.00	0.10	0.13
6-8	0.08	0.39	0.08	0.40	0.40
6-9	0.00	0.26	0.01	0.11	0.25
7-8	0.12	0.39	0.14	0.38	0.45
7-9	0.08	0.57	0.11	0.60	0.70
8-9	0.14	0.53	0.17	0.64	0.62
Average	0.07	0.29	0.07	0.32	0.31

Table 3: Average classification erors for pairwise networks (in percent)

set of classifiers	selection criterion	criterion measured on	error on MNIST
A	none / select all from withing a single set	N/A	1.48 %
B	none / select all from withing a single set	N/A	1.54 %
A	loss	training dataset	**1.24** %
B	loss	training dataset	1.38 %
B	loss	validation dataset	1.32 %
A	error rate	training dataset	1.27 %
B	error rate	training dataset	1.33 %
B	error rate	testing dataset	1.35 %

Table 4: Performance on MNIST classification

set of classifiers	selection criterion	criterion measured on	error on MNIST
A	error rate	testing dataset	0.88 %
B	error rate	testing dataset	0.83 %

Table 5: Performance on MNIST classification

On the surface they may seem trivial and impractical, but one may expect niche applications such as classification of medical images [13] would naturally have only two classes to consider. Binary classification problems of images may also be more amenable to theoretical analysis.

The pairwise models considered in this paper were formed from share-nothing networks. The hope was that the pairwise coupling method would be able to counteract their smaller individual capacity/precision and achieve equal or better multiclass classification performance by combining and extracting diversity of information contained within these networks.

Share-nothing approach is not viable for problems with larger number of categories than MNIST. There are other approaches to pairwise multi-class classification with neural networks that use partial sharing of learning capacity by pairwise classifiers. It is also possible to create ensemble models without pairwise coupling [14, 15]. We hope to investigate these questions in future works.

Acknowledgment

Work on this paper was partially supported by grants VEGA 2/0144/18 and APVV-14-0560. We also thank Google Inc. for providing us with education credit #32870744 that we used on Google Compute cloud service.

References

[1] LeCun Y., Bengio Y., Hinton G., Deep learning. Nature **521**, (2015), 436–444

[2] Hestness J. et al, Deep learning is predictable, empirically, arXiV:1712.00409

[3] Jouppi N.P. et al, In-datacenter performance analysis of a tensor processing unit," in Proceedings of ISCA'17, Toronto, ON, Canada, (2017)

[4] Cortes C., Vapnik V.N., Support-vector networks, Machine learning, 20, (3), (1995), 273–297

[5] Boser B.E., Guyon M.I., Vapnik V.N., A training algorithm for optimal margin classifiers, Proceedings of the fifth annual workshop on Computational learning theory – COLT'92, (1992), p.144

[6] Platt J., Probabilities for SVM machines, in Advances in Large Margin classifiers, Smola A.J., Bartlett P.L., Scholkopf B., Schuurmans, D., Eds. MIT Press, (2000), 61–74

[7] Chang, C-C., Lin C-J., LIBSVM: A library for support vector machines, ACM Transactions on Intelligent System and Technology, vol 2., issue 3, (2011)

[8] LeCun, Y., The MNIST database of handwritten digits, (1998), http://yan.lecun.com/exdb/mnist

[9] Hastie T., Tibshirani R., Classification by pairwise coupling, http://fisher.utstat.toronto.edu/pub/tibs/coupling.ps

[10] Chollet F. et al, Keras, (2015), https://keras.io

[11] Wu T-F., Lin C-J., Weng R.C., "Probability estimates for multi-class classification by pairwise coupling", Jounral of Machine Learning Reserach, (2004), 975–1005

[12] Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R., Dropout: A simple way to prevent neural networks from overfitting, Jounral of Machine Learning Research, **15**, (2014), 1929–1958

[13] Litjens G. et al, A survey of deep learning in medical image analysis, Medical Image Analysis, **42**, (2017), 60–88

[14] Hartono P., Ensemble of Perceptrons with Confidence Measure for Piecewise Linear Decomposition, IEEE Int. Joint Conf. on Neural Networks (IJCNN 2011), (2011), 648–653

[15] P. Hartono, Ensemble of Linear Experts as an Interpretable Piecewise Linear Classifier, Innovative Computing, Information and Control Express Letters Vol. 2, No. 3, (2008), 295–303