A Greedy Algorithm to Cluster Specialists

Seb Arnold
arnolds@usc.edu

February 13, 2016

Abstract

With the recent advances in deep neural networks, several experiments involved the generalist-specialist paradigm for classification. However, until now no formal study compared the performance of different clustering algorithms for class assignment. In this paper we perform such a study, suggest slight modifications to the clustering procedures, and propose a novel algorithm designed to optimize the performance of of the specialist-generalist classification system. Our experiments on the CIFAR-10 and CIFAR-100 datasets allow us to investigate situations for varying number of classes on similar data. We find that our *greedy_pairs* clustering algorithm consistently outperforms other alternatives, while the choice of the confusion matrix has little impact on the final performance.

I Introduction

Designing an efficient classification system using deep neural networks is a complicated task, which often use a multitude of models arranged in ensembles. ([Dieleman et al., 2015], [Simonyan and Zisserman, 2014]) Those ensembles often lead to state-of-the-art results on a wide range of different tasks such as image classification ([Szegedy et al., 2015]), speech recognition ([Amodei et al., 2015]), and machine translation ([Sutskever et al., 2014]). The models are trained independently and in parallel, and different techniques can be used to merge their predictions.

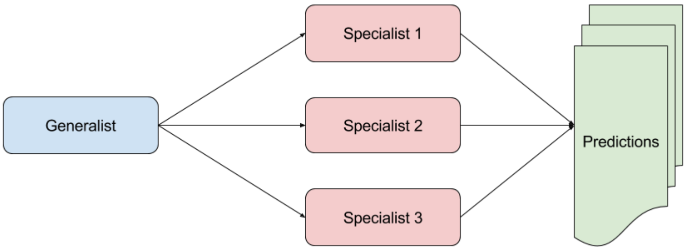

Figure 1: An example of specialist architecture with three specialists

A more structured alternative to ensembling is the use of the specialist-generalist framework. As described by [Bochereau and Bourgine, 1990], a natural analogy can be drawn from the medical field; a patient first consults a general practitioner who provides an initial diagnosis which is then refined by one or several specialists. In the case of classification, the doctors are replaced by neural networks and the final prediction is a combination of the specialists' outputs, and may or may not include the generalist's take.

In recent years, generalist and specialists have been studied under different circumstances. [Hinton et al., 2015] used specialists to create an efficient image classifier for a large private dataset. The final predictions of the specialists were then used to train a reduced classifier that achieved performance similar to the whole ensemble. [Kahou et al.,] describe a multimodal approach for emotion recognition in videos, based on specialists. Maybe closer to our work, [Warde-Farley et al., 2014] added "auxiliary heads" (acting as specialists) to their baseline network, using the precomputed features for both classification and clustering. They also underlined one of the main advantages of using specialists; a relatively low (and parallelizable) additional computational cost for increased performance.

II Clustering Algorithms

In order to assign classes to the specialist networks, we compare several clustering algorithms on the confusion matrix of the outputs of the generalist. This confusion matrix is computed on a held-out partition of the dataset. Following previous works, we started by considering two baseline clustering algorithms, namely Lloyd's K-Means algorithm and Spectral clustering, according to the formulation of [Ng et al.,]. In addition to those baseline algorithms, we evaluate the performance of two novel procedures specifically designed to improve the generalist-specialist paradigm. Those algorithms are described in the following paragraphs, and pseudo code is given in the Appendix.

We also experimented with different ways of building the confusion matrix. Besides the usual way (denoted here as *standard*) we tried three alternatives:

- *soft sum*: for each prediction, we use the raw model output instead of the one-hot multi-class output,
- *soft sum pred*: just like *soft sum*, but only add the prediction output to the confusion matrix, if the class was correctly predicted,
- *soft sum not pred*: like to *soft sum pred*, but only if the prediction output was incorrectly predicted.

As discussed in later sections, the influence of the confusion matrix is minimal. Nonetheless we include them for completeness purposes.

Both of our clustering algorithms further modify the confusion matrix A by computing $CM = \mathbf{A}^\top + \mathbf{A}$, which symmetrizes the matrix. We define the entries of the matrix to be the *animosity score* between two classes; given classes a and b, their animosity score is found at $CM_{a,b}$. We then initialize each cluster with non-overlapping pairs of classes yielding maximal animosity score. Finally, we greedily select the next classes to be added to the clusters, according to the following rules:

- In the case of *greedy single* clustering, a single class maximizing the overall animosity score is added to the cluster yielding the largest averaged sum of animosity towards this class. This partitions the classes in clusters, building on the intuition that classes that are hard to distinguish should be put together.

- In the case of *greedy pairs* clustering, we follow the same strategy as in *greedy single* clustering but act on pair of classes instead of single classes. In this case we allow the clusters to overlap, and one prediction might include the opinion of several specialists.

This process is repeated until all classes have been assigned to at least one cluster.

III Experiments

We investigate the performance of the aforementioned algorithms on the CIFAR-10 and CIFAR-100 datasets ([Krizhevsky, 2009]). Both datasets contain similar images, partitioned in 45'000 train, 5'000 validation, and 10'000 test images. They contain 10 and 100 classes respectively. For both experiments we train the generalist network on the train set only, and use the validation set for clustering purposes. As we are interested in the clustering performance we did not augment nor pre-process the images. Note that when trained on the horizontally flipped training and validation set our baseline algorithm reaches 10.18% and 32.22% misclassification error respectively, which is competitive with the current state-of-the-art presented in [Springenberg et al., 2014].

Following [Courbariaux et al., 2015], the baseline network is based on the conclusions of [Simonyan and Zisserman, 2014] and uses three pairs of batch-normalized convolutional layers, each followed by a max-pooling layer, and two fully-connected layers. The same model is used for specialists, whose weights are initialized with the trained weights of the generalist. [1] One major departure from the work of [Hinton et al., 2015] is that our specialists are predicting over the same classes as the generalist, i.e. we do not merge all classes outside of the cluster into a unique one. With regards to the generalist, a specialist is only biased towards a subset of the classes, since it has been fine-tuned to perform well on those ones.

CIFAR-10

For CIFAR-10 experiments, we considered up to five clusters, and all of the possible combinations of confusion matrix and clustering algorithms. The results for this experiments are reported in Table 1.

Results	standard	soft sum	soft sum pred	soft sum not pred
spectral	(0.7342, 2)	(0.4117, 3)	(0.4541, 4)	(0.4143, 2)
greedy singles	(0.2787, 3)	(0.2774, 2)	(0.3869, 4)	(0.2727, 2)
kmeans	(0.8037, 2)	(0.8037, 2)	(0.8034, 2)	(0.804, 2)
greedy pairs	(0.8584, 3)	(0.8483, 3)	(0.8473, 3)	(0.8611, 3)

Table 1: Experiment results for CIFAR-10

[1] The code for those experiments, is freely available online at github.com/seba-1511/specialists.

Interestingly, the choice of confusion matrix has only a limited impact on the overall performance, indicating that the emphasis should be put on the clustering algorithm. We notice that clustering with greedy pairs consistently yields better scores. However none of the specialist experiments is able to improve on the baseline, suggesting that specialists might not be the framework of choice when dealing with a small number of classes.

CIFAR-100

For CIFAR-100 we performed the exact same experiment as for CIFAR-10 but used more specialists, the largest experiments involving 28 clusters. The results are shown in Table 2.

Results	standard	soft sum	soft sum pred	soft sum not pred
spectral	(0.5828, 2)	(0.5713, 2)	(0.5755, 2)	(0.5795, 3)
greedy singles	(0.3834, 2)	(0.3733, 2)	(0.3803, 2)	(0.3551, 2)
kmeans	(0.5908, 2)	(0.5618, 2)	(0.5820, 3)	(0.5876, 2)
greedy pairs	(0.6141, 6)	(0.5993, 6)	(0.6111, 6)	(0.607, 6)

Table 2: Experiment results for CIFAR-100

Similarly to CIFAR-10, we observe that greedy pairs clustering outperforms the other clustering techniques, and that the different types of confusion matrix have a limited influence on the final score. We also notice that fewer clusters tend to work better. Finally, and unlike the results for CIFAR-10, some of the specialists are able to improve upon the generalist, which confirms our intuition that specialists are better suited to problems involving numerous output classes.

We suggest the following explanation for the improved performance of greedy pairs is the following. Allowing clusters to overlap leads to the assignment of difficult classes to multiple specialists. At inference time, more networks will influence the final prediction which is analogous to building a larger ensemble for difficult classes.

IV Conclusion and Future Work

We introduced a novel clustering algorithm for the specialist-generalist framework, which is able to consistently outperform other techniques. We also provided a preliminary study of the different factors coming into play when dealing with specialists, and concluded that the choice of confusion matrix from our proposed set only has little impact on the final classification outcome.

Despite our encouraging results with clustering techniques, no one of our specialists-based experiments came close to compete with the generalist model trained on the entire train and validation set. This was a surprising outcome and we suppose that this effect comes from the size of the datasets. In both cases, 5'000 images corresponds to 10% of the original training set and removing that many training examples has a drastic effect on both generalists and specialists. All the more so since we are not using any kind of data augmentation techniques, which could have moderated this downside. An obvious future step is to validate the presented ideas on a much larger dataset such as [Russakovsky et al., 2015] where splitting the train set would not hurt the train score as much.

Acknowledgments

We would like to thank Greg Ver Steeg, Gabriel Pereyra, and Oriol Vinyals for their comments and advices. We also thank Nervana Systems for providing GPUs as well as their help with their deep learning framework.

References

[Amodei et al., 2015] Amodei, D., Anubhai, R., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Chen, J., Chrzanowski, M., Coates, A., Diamos, G., et al. (2015). Deep speech 2: End-to-end speech recognition in english and mandarin. *arXiv preprint arXiv:1512.02595.*

[Bochereau and Bourgine, 1990] Bochereau, L. and Bourgine, P. (1990). A generalist-specialist paradigm for multilayer neural networks. In *Neural Networks, 1990., 1990 IJCNN International Joint Conference on*, pages 87–91. IEEE.

[Courbariaux et al., 2015] Courbariaux, M., Bengio, Y., and David, J.-P. (2015). Binaryconnect: Training deep neural networks with binary weights during propagations. In *Advances in Neural Information Processing Systems*, pages 3105–3113.

[Dieleman et al., 2015] Dieleman, S., Willett, K. W., and Dambre, J. (2015). Rotation-invariant convolutional neural networks for galaxy morphology prediction. *Monthly Notices of the Royal Astronomical Society*, 450(2):1441–1459.

[Hinton et al., 2015] Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531.*

[Kahou et al.,] Kahou, S. E., Bouthillier, X., Lamblin, P., Gulcehre, C., Michalski, V., Konda, K., Jean, S., Froumenty, P., Dauphin, Y., Boulanger-Lewandowski, N., et al. Emonets: Multimodal deep learning approaches for emotion recognition in video. *Journal on Multimodal User Interfaces*, pages 1–13.

[Krizhevsky, 2009] Krizhevsky, A. (2009). Learning multiple layers of features from tiny images.

[Ng et al.,] Ng, A. Y., Jordan, M. I., and Weiss, Y. On spectral clustering: Analysis and an algorithm.

[Russakovsky et al., 2015] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252.

[Simonyan and Zisserman, 2014] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556.*

[Springenberg et al., 2014] Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.

[Sutskever et al., 2014] Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

[Szegedy et al., 2015] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9.

[Warde-Farley et al., 2014] Warde-Farley, D., Rabinovich, A., and Anguelov, D. (2014). Self-informed neural network structure learning. *arXiv preprint arXiv:1412.6563*.

V Appendix

Greedy Pairs Pseudo Code

Algorithm 1 Greedy Pairs Clustering

```
procedure GreedyPairs($M, N$)        ▷ Confusion matrix M, number of clusters N
    $M \leftarrow M + M^T$
    Initialize N clusters with non-overlapping pairs maximizing the entries of M.
    while every class has not been assigned do
        Get the next pair $(a, b)$ maximizing the entry in M
        cluster = $\underset{\text{c in clusters}}{\text{argmin}}$ (Animosity(a, c) + Animosity(b, c))
        Assign(cluster, a, b)
    return clusters
```

Note: A python implementation of both greedy pairs and greedy single can be found at `http://www.github.com/seba-1511/specialists`.