Efficient Image Dataset Classification Difficulty Estimation for Predicting Deep-Learning Accuracy

Florian Scheidegger∗†, Roxana Istrate‡†, Giovanni Mariani†
Luca Benini∗§, Costas Bekas†, Cristiano Malossi†
∗ETH Zürich, Switzerland †IBM Research - Zürich, Switzerland
‡Queen’s University of Belfast, United Kingdom §Università di Bologna, Italy

Abstract—In the deep-learning community new algorithms are published at an incredible pace. Therefore, solving an image classification problem for new datasets becomes a challenging task, as it requires to re-evaluate published algorithms and their different configurations in order to find a close to optimal classifier. To facilitate this process, before biasing our decision towards a class of neural networks or running an expensive search over the network space, we propose to estimate the classification difficulty of the dataset. Our method computes a single number that characterizes the dataset difficulty 27× faster than training state-of-the-art networks. The proposed method can be used in combination with network topology and hyper-parameter search optimizers to efficiently drive the search towards promising neural-network configurations.

I. INTRODUCTION

Convolutional Neural Networks (CNNs) gained popularity in recent years thanks to the availability of powerful GPUs that enable to efficiently train accurate classification models [1]. For building practical applications, the deep-learning community shares a common interest in reducing the development cycle, while increasing model accuracy and keeping infrastructure and power consumption expenditure under control. Many publications address these conflicting goals [2], [3], [4]. Most machine-learning approaches require a human in the loop responsible for taking crucial decisions such as defining the network, finding good combinations of hyper-parameters and performing adequate preprocessing on the input data. To overcome the problem of manual selection various automated approaches such as Grid Search, Random Search [5], Bayesian optimization [6] or Hyperband optimization [7] have been proposed. These methods operate autonomously and improve model performance, however they still have two limiting factors. First, they require a definition of the search space. Second, they consume a large amount of resources for a single optimization task.

In this paper we propose automated methods for quantifying the difficulty of a classification problem in terms of how hard it is to reach high accuracy for a given dataset. The proposed method can be used in combination with architecture search optimizers to efficiently drive the search towards promising configurations, avoiding the exploration of unsuitable networks. Consciously or not, the characterization of dataset difficulty is a process followed by every deep learning architect. When looking for a well-performing model for a new dataset, common practice is to try state-of-the-art networks to evaluate how hard is to classify the images in the dataset. Since datasets are large and models complex, the process of training, comparing, and selecting a few state-of-the-art deep networks becomes a computationally heavy task. We propose to optimize this step by providing a classification difficulty estimator, that provides insights into the classification task and can be used to rapidly confine the exploration to a few promising networks. We aim to construct dataset characterizations that run orders of magnitude faster than the actual training and have high correlation with state-of-the-art network accuracies.

In summary, our main contributions are the following:

• We propose and evaluate three different dataset complexity scoring pipelines.
• We conduct various deep learning experiments with fixed hyper-parameter and data augmentation configurations that run on thirteen datasets.
• We evaluate approximate computing techniques, such as subsampling and early stopping, in order to reduce the execution time without affecting the end results.
The remainder of the paper is organized as follows. Section II describes the related work, Section III introduces the notation used throughout the paper, Section IV details the adopted methodologies, Section V examines the results, and Section VI concludes the current work.

II. RELATED WORK

The topic of difficulty estimation of a dataset is scarcely explored in the literature. In [8], the authors address this problem by using, but use as reference the human response time for solving a visual search task. As compared to our technique which focuses on defining how easily separable are the different classes in a dataset, their technique analyses a dataset difficulty on an image based approach and employs two VGG-like [9] networks that work as encoders and extract features that are further passed through a regressor. The complexity of this solution comes from passing the full dataset through the VGG-like networks, since the VGG family includes one of the largest state-of-the-art network with 138M parameters and 15 GFLOPs/inference.

Figure 1 shows state-of-the-art accuracies achieved by thirteen of the most commonly used image classification datasets: MNIST [10], GTSRB [11], svhn [12], CIFAR10 [13], flowers1, flowers102 [14], fashion MNIST [15], food101 [16], CIFAR100 [13], stl10 [17], textures [18], indoor67 [19], and places [20]. As expected, more results are available for the highly curated datasets, such as MNIST and CIFAR10/100. The same holds for datasets introduced as part of a competition, such as the German Traffic Sign Recognition Benchmark (GTSRB), where the authors published the full performance list of over a hundred machine learning algorithms. For less popular datasets, results are harder to acquire. In this category fits the flowers dataset, which was introduced in a tensorflow tutorial with the purpose of explaining transfer learning. In that case, transferred learning provides superior classification performance over any CNN trained from scratch. General embeddings obtained from pretrained models help increasing performance for a specific task, especially when the data is limited [22]. Figure 1 reveals that all mentioned algorithms, even non-CNN based ones, easily reach accuracies above 95% on MNIST. This observation motivated the authors of fashion MNIST to introduce a more diverse dataset in terms of images, but equal to MNIST in terms of number of training/validation samples, image sizes and number of classes. This allows any algorithm designed for MNIST to run without modification on fashion MNIST. Their initial study demonstrates a wider spread in performance among different algorithms and consistently lower performance when compared with the same algorithms evaluated on MNIST [15].

To the best of our knowledge, there is no published work that focuses on automatically ranking classification difficulty among datasets.

III. NOTATION

In this work, we refer to a dataset with the quadruple \( \mathcal{D} := (X_{\text{train}}, y_{\text{train}}, X_{\text{test}}, y_{\text{test}}) \), where \( X_{\text{train}} \in \mathbb{R}^{n_{\text{train}} \times d} \) and \( X_{\text{test}} \in \mathbb{R}^{n_{\text{test}} \times d} \) are the training and testing inputs, and \( y_{\text{train}} \in \{1, C\}^{n_{\text{train}}} \) and \( y_{\text{test}} \in \{1, C\}^{n_{\text{test}}} \) are the training and testing labels. We assume that the datasets come already split into train and test sets, as this is commonly the case for published data. We denote the input dimension as \( d \), the number of input samples as \( n_{\text{train}} \) for training and \( n_{\text{test}} \) for testing, and the number of classes as \( C \). \( \mathcal{M} \) refers to a model, including the network topology and related hyper-parameters, and it includes the training and data augmentation related hyper-parameters. Therefore, the tuple \((\mathcal{D}, \mathcal{M})\) specifies a deep learning training run of model \( \mathcal{M} \) on dataset \( \mathcal{D} \). We denote with Top-1(\( \mathcal{D}, \mathcal{M}) \) the Top-1 accuracy classification performance of the training run. In all

---

1 Available at http://download.tensorflow.org/example_images/flower_photos.tgz
2 Available at https://www.tensorflow.org/tutorials/image_retraining
experiments, training is performed with \((X_{\text{train}}, y_{\text{train}})\) and performance is measured on \((X_{\text{test}}, y_{\text{test}})\).

### IV. Ranking Datasets

In this work we propose metrics to quantify the difficulty of image classification datasets. We propose dataset scoring functions \(r(D)\) to map a dataset \(D\) to a scalar real number with the goal of ranking different datasets in terms of classification accuracy estimates.

#### A. Silhouette Score

The silhouette score is a well established metric that compares tightness of same-class samples to separation of different-class samples [23]. Let \(i\) be one input sample, \(a(i)\) the average euclidean distance between the sample and all the points \(j\) belonging to the same class as \(i\), and \(b(i)\) the average distance between \(i\) and all points \(j\) of the closest different class. The silhouette of the \(i\)-th sample is computed [23] as follows:

\[
s(i) := \begin{cases} 
1 - a(i)/b(i), & \text{if } a(i) < b(i) \\
0, & \text{if } a(i) = b(i) \\
b(i)/a(i) - 1, & \text{if } a(i) > b(i).
\end{cases}
\]

The silhouette of one class is defined as the average over all samples belonging to that class and the overall silhouette score of the full dataset is defined as average over all samples. The definition of the quantities \(a(i)\) and \(b(i)\) are based on pairwise distances between two samples \(i\) and \(j\). The silhouette score complexity is \(O(\tilde{d} n^2)\), where \(n\) is the number of samples and \(\tilde{d}\) is the cost of computing the distance of one pair of samples as mean squared error (MSE) distance in the \(\mathbb{R}^d\). Since, the MSE distance in the original domain is a poor measurement for image similarities, we apply first a transformation \(\mathbb{R}^d \rightarrow \mathbb{R}^d\) that maps images into a space that better reflects distances between image pairs.

Table I provides details on the applied pipelines. We decided to include a resizing of the images to a small resolution of \(8 \times 8\) pixels, applying principal component analysis (PCA) to reduce the dimension to 10, and using a fixed encoding based on a pretrained CNN inference. We considered as encoder a ResNet-50 [24] network pretrained on ImageNet [25] to produce generalized per image feature vectors of dimensionality 1000 by taking the output of the last fully connected layer before applying the non-linearity. Additional to the MSE distance, we used the structural dissimilarity index DSSIM [26] to compare images with a metric that captures spatial information. Due the squared complexity, we applied heavy subsampling and run all computations with a maximum of 1000 randomly selected samples, resulting in a distance matrix with at most \(1M\) entries. Table I states timing among the different pipelines. For fast execution, it is crucial to operate in a low dimensional space and to use a simple distance metric.

#### B. K-means Clustering

The complexity of the silhouette scores detailed in Subsection IV-A scales with \(n^2\), and computing it is a slow process even after subsampling. In general, the complexity of a deep-learning job is \(O(c(M)n_{\text{train}}e)\), where \(e\) is the number of epochs. During one epoch the full training set consisting of \(n_{\text{train}}\) samples is fed once with a computational cost of \(c(M)\), where \(c(M)\) is a model dependent constant. Even though complex models might have large computational cost of \(c(M)\), the asymptotic behaviour of a training job is linear in \(n\). For this reason, the asymptotic behaviour of the silhouette score computation of \(n^2\) is outperformed by the actual training job. Competitive scoring metrics should execute faster than a train job itself, thus we are looking for scores with at most linear complexity in \(n\).

We propose to run a (fast) clustering algorithm to produce class labels \(\hat{y}\) and evaluate the full dataset based on metrics that compare \(\hat{y}\) against the ground truth labels \(y\). We assess the following known scores: adjusted mutual information [27], adjusted rand index [28], completeness, homogeneity and the v-measure [29]. Additionally, we propose an own tailored score based on the estimation of the confusion matrix built between the cluster indices and the true labels. Clustering algorithms work in an unsupervised fashion, meaning that the clusters are not assigned to any label. To estimate the confusion matrix we require an one-to-one mapping between the clusters and the labels. A naïve solution computes all possible permutations, and selects the one that maximizes the

### Table I

| Score | Transformation | \(\tilde{d}\) | Distance | Speedup |
|-------|----------------|---------|----------|---------|
| \(S_1\) | None | \(d\) | MSE | 31.3× |
| \(S_2\) | None | \(d\) | DSSIM | 1.0 (Ref) |
| \(S_3\) | Resize image | \(s^2\) | MSE | 48.4× |
| \(S_4\) | Resize image | \(s^2\) | DSSIM | 1.3× |
| \(S_5\) | PCA | 10 | MSE | 72.8× |
| \(S_6\) | Autoencoder | 1000 | MSE | 6.4× |
trace normalized by the amount of total data points. Due to the factorial complexity, we propose to compute an approximate estimate with the following greedy algorithm. First, we search the maximum values per row and assign the clusters to the corresponding label. If the greedy strategy resolves to a bijective mapping, the accuracy is computed based on the found confusion matrix. Otherwise, the non-bijective contradictions are solved with extensive search. Since this approach still results in a worst case factorial complexity, we set an maximum limit of contradictions to be solved by exhaustive search to seven, and solve other contradictions by assigning an initial permutation. Hence, the accuracy estimate is the maximum achieved over all values obtained with permutations on the remaining problematic locations. It turns out that the proposed procedure is stable and fast to evaluate and it builds a lower bound to the optimal value. In few cases our chosen maximum limit of seven contradictions was exceeded.

C. Probe Nets

As alternative scoring metric we also investigate the possibility of training a small predefined neural network and score the dataset based on its accuracy. We call this network a probe network. The probe net model \( M_{\text{probe}} \) must be general enough to be applied to any image classification task and considerably faster: \( c(M_{\text{probe}}) \ll c(M) \). Additionally, to further speedup the execution time we stop the training of the probe net after few epochs, before full convergence.

We propose to construct variations of two types of probe nets: static probe nets that have a fixed topology and dynamic probe nets that scale the topology according to the number of classes. The regular probe
net consists of three convolutional layers, each followed by batch normalization, max pooling of size $2 \times 2$, and ReLU activations, which are defined element-wise as $x \mapsto \max(0, x)$. We used eight kernels in the first layer and doubled the number of kernels per layer. We provide wide and narrow variations that scale the number of kernels per layer up and down by $4 \times$, respectively. Shallow and deep variations are obtained by subtracting and adding two layers, respectively. Since doubling the kernel sizes per layer leads to different tensor shapes between the last convolution and the $C$-way softmax, the non-normalized shallow and deep probe nets have a considerable different number of trainable parameters. We define normalized probe networks to match the number of trainable parameters of the output layer of the regular probe net. We construct dynamic nets with a more complex topology to account for more classes. This is achieved either by scaling dependent on $C$ the number of hidden units in an multilayer perceptron mlp, the number of filters (filter depth scaled probe nets), or the number of stacked filters (length scaled probe net). Figure 2 shows the ten proposed prob net architectures.

V. Results

In order to perform a fair evaluation, we fix hyper-parameters throughout the experiments. We use a ResNet-20 topology to compute the reference accuracy on all datasets resized to $32 \times 32$ pixels. We follow the data augmentation described in [30]. We use the RMSProp [31] optimizer to minimize the average cross entropy with a learning rate of $10^{-4}$. All evaluations employ the He initialization [32] with a gain factor 1.0 and a constant batch size of 32. Training is run for 100 epochs. Results presented in Figure 3, Figure 4, and Figure 5 state the computed classification difficulty score on the $x$-axis plotted against the ResNet-20 accuracy reference that is shared among all plots. An ideal dataset
difficulty score should obey a linear dependency and match the reference accuracies.

A. Silhouette Score

Figure 3 compares the scores based on the proposed pipelines in Table I with the reference accuracies of a ResNet-20 [24]. The silhouette score based on the DSSIM pairwise distance outperforms the MSE based distance, as the former preserves spatial information of the image domain. Similarly, the (more expensive) computation with the full image domain slightly outperforms the resized counterpart, since it benefits from more information. Subfigure c) shows results when the silhouette score is applied on the PCA reduced space and in Subfigure f) the PCA is replaced by the autoencoder. Although reducing the dimensionality is often beneficial, we obtained the best correlation results with the DSSIM based distance on the original domain, with a weak correlation of \( R^2 = 0.31 \).

B. K-means Clustering

For the evaluation of the proposed \( k \)-means based scoring pipeline (see Subsection IV-B), we cluster the images in \( C \) clusters, where \( C \) is the known number of categories in the dataset. For a faster convergence, we initialize the centroids with the average image of each class. \( k \)-means runs based on the euclidean \( L_2 \) distance with a stopping tolerance of \( 10^{-4} \) and a maximum of 300 iterations without random restarts. Figure 4 shows the regression among proposed scores and the obtained reference accuracy of a ResNet-20.

Among columns different scores are assessed, such as the accuracy on the estimated confusion matrix (AECM), the adjusted mutual information score, the adjusted random score and the \( v \)-measure. Results for the homogeneity score and the completeness score are omitted since they are highly correlated with the \( v \)-measure. Except the \( v \)-measure computed in the encoded setting, the scores are weakly positively correlated with the obtained ResNet-20 reference. Resizing to a small dimension of \( 8 \times 8 \) only marginally affects results, applying PCA helps to improve predictions from around \( R^2 = 0.25 \) up to \( R^2 = 0.31 \). The encoder based pipeline provides an embedding that is in the same order of quality (\( R^2 = 0.30 \)). Among the score computations the one based on AECM outperforms the other scores. The weak performance of the \( k \)-means clustering is due to known limitations, such as no global minimum guarantee and poor distance metric that ignores the spatial information. \( k \)-means clustering based pipelines are 5.2\( \times \) (no pretransformation) up to 50.5\( \times \) (PCA pretransformation) faster than silhouette score based pipelines (comparison includes the faster MSE timings) when comparing execution times in terms of average per input sample.

C. Probe Nets

All proposed probe nets, as presented in Figure 2, are trained with the same constant configuration and data augmentation parameters as explained in Section V. Results are obtained after training for 100 epochs. Figure 5 shows all obtained correlations between runs of the ten proposed probe nets against the reference. All probe nets share a high correlation with the reference ResNet-20 of \( R^2 > 0.88 \) and consistently outperform results achieved with the \( k \)-means based approach, see Subsection V-B. Subfigure a) shows an increasing correlation of \( R^2 = 0.89 \) to \( R^2 = 0.98 \) between narrow, regular, and wide probe nets and the reference. This can be explained by the better generalization ability of the network with more degrees of freedom, at the cost of an increased execution time. For more details see Table II. Deep probe nets topologies outperform their shallow counterparts. This effect is even more prominent in the normalized case, Subfigure b) versus Subfigure d). We observe that a better generalization performance is mainly driven by a larger amount of tuneable parameters that comes at the cost of increased execution timings. Subfigures c) and e) of Figure 5 show the results for probe nets that dynamically adapt the architecture topology to the number of classes. The dependency of the architecture on the number of classes implies different execution times on datasets with different number of classes. The \( mlp \) can not compete with the CNN counterparts.

D. Efficient Evaluation of Probe Nets

As presented in Subsection V-C probe nets have a good predictive behaviour of what a reference network achieves on a given dataset. However, that information is only valuable if it can be computed order of magnitudes faster than training large models. The way probe nets are constructed give them an inherent computational benefit over the full model. In addition, we exploit early stopping of the learning to further reduce the computational time of the probe net. Note that we can stop the probe net before convergence, since we are interested in the learning trend that characterizes the problem’s difficulty, not in the final accuracy. Figure 6 shows how the prediction quality improves for a regular probe net with an increasing amount of epochs for which
Fig. 4. Prediction performance of $k$-means based dataset scores: the $x$-axis states variations of the $k$-means based scores from Subsection IV-B and the $y$-axis states the model performance obtained by a state-of-the-art architecture, a ResNet-20 topology. Rows state results for applying none, resizing, PCA, and encoding as pretransformations. Columns state results for four metrics comparing the clustering with the ground truth classification. Scores based on the estimated confusion matrix outperform the others and applying PCA or an embedding is beneficial for accuracy prediction.

It is trained on all datasets. Within a few epochs the regression quality reaches a saturation at about $R^2 = 0.95$. The mean accuracy difference between the probe nets and the reference ResNets (trained till convergence) is further decreased, meaning that the probe nets are not yet converged and are still increasing their own classification performance. Figure 6 highlights the achieved quality at epoch 1, 5 and 10. Figure 7 presents both intermediate
and after convergence results. With increasing number of epochs, the regression moves from the top-left corner towards the identity line in the middle. As few as 5 epochs are enough to reach the full dataset performance prediction ability, well before that actual probe net has converged.

VI. CONCLUSION

We formulated the question to compute a ranking among datasets that reflect their inherent classification difficulty. We suggested three processing pipelines, a silhouette based score, a \(k\)-means clustering based and a probe net based evaluation pipeline. The main drawback of the silhouette based approach is the high complexity, which scales with the squared number of samples. We proposed efficient score computing pipelines based on \(k\)-means and probe nets that scale linear in the number of samples. \(k\)-means delivers results one complexity class faster and with similar prediction quality as the silhouette approach, reaching a weak correlation with reference models of \(R^2 = 0.31\). Finally, we presented the probe nets, which are small networks, and apply standard deep learning techniques in order to compute predictions that are strongly correlated with the reference from \(R^2 = 0.89\) up to \(R^2 = 0.99\). Even the worst performing probe net outperforms silhouette and \(k\)-means based scoring with a wide quality margin. We further evaluated the fact of early stopping to reduce the data score evaluation time and observed little to no performance drop. Leveraging the small architectures of probe nets and early stopping allows to perform dataset scoring \(27\times\) faster than the required training time of the actual reference model.

ACKNOWLEDGEMENTS

We would like to thank Dr. Dario Garcia Gasulla from the Barcelona Supercomputing Center for discussion and advise. This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 732631, project OPRECOMP.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1026–1034.
[2] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, “Binarized neural networks: Training deep neural networks with weights and activations constrained to \(+1\ or-1\),” arXiv preprint arXiv:1602.02830, 2016.
[3] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, “Deep learning with limited numerical precision,” in International Conference on Machine Learning, 2015, pp. 1737–1746.
Regression quality

Average accuracy difference

Fig. 6. Evolution of the prediction quality over training epochs of the \textit{regular} probe net. The regression quality saturates around $R^2 = 0.96$ within a few epochs while the average accuracy difference between the probe net and the reference is further decreased for longer training.

[4] M. H. J. A. B. Hanzhang Hu, Debadeepta Dey, “Anytime neural network: a versatile trade-off between computation and accuracy,” 2017.

[5] J. Bergstra and Y. Bengio, “Random search for hyper-parameter optimization,” \textit{Journal of Machine Learning Research}, vol. 13, no. Feb, pp. 281–305, 2012.

[6] J. Snoek, H. Larochelle, and R. P. Adams, “Practical bayesian optimization of machine learning algorithms,” in \textit{Advances in Neural Information Processing Systems 25}, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 2951–2959. [Online]. Available: http://papers.nips.cc/paper/4522-practical-bayesian-optimization-of-machine-learning-algorithms.pdf

[7] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, “Hyperband: Bandit-based configuration evaluation for hyperparameter optimization,” 2016.

[8] R. Tudor Ionescu, B. Alexe, M. Leordeanu, M. Popescu, D. P. Papadopoulos, and V. Ferrari, “How hard can it be? estimating the difficulty of visual search in an image,” in \textit{Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition}, 2016, pp. 2157–2166.

[9] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” \textit{arXiv preprint arXiv:1409.1556}, 2014.

[10] L. Deng, “The mnist database of handwritten digit images for machine learning research [best of the web],” \textit{IEEE Signal Processing Magazine}, vol. 29, no. 6, pp. 141–142, 2012.

[11] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, “The german traffic sign recognition benchmark: A multi-class classification competition,” in \textit{The 2011 International Joint Conference on Neural Networks}, July 2011, pp. 1453–1460.

[12] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, “Reading digits in natural images with unsupervised feature learning,” in \textit{NIPS workshop on deep learning and unsupervised feature learning}, vol. 2011, no. 2, 2011, p. 5.

[13] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” 2009.

[14] M. E. Nilsback and A. Zisserman, “Automated flower classification over a large number of classes,” in \textit{2008 Sixth Indian Conference on Computer Vision, Graphics Image Processing}, Dec 2008, pp. 722–729.

[15] H. Xiao, K. Rasul, and R. Vollgraf. (2017) Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.

[16] L. Bossard, M. Guillaumin, and L. Van Gool, “Food-101 – mining discriminative components with random forests,” in \textit{Computer Vision – ECCV 2014}, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 446–461.

[17] A. Coates, A. Ng, and H. Lee, “An analysis of single-layer networks in unsupervised feature learning,” in \textit{Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics}, ser. Proceedings of Machine Learning Research, G. Gordon, D. Dunson, and M. Dudík, Eds., vol. 15. Fort Lauderdale, FL, USA: PMLR, 11–13 Apr 2011, pp. 215–223. [Online]. Available: http://proceedings.mlr.press/v15/coates11a.html

[18] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, “Describing textures in the wild,” in \textit{Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition}, ser. CVPR ’14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 3606–3613. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2014.461

[19] A. Quattoni and A. Torralba, “Recognizing indoor scenes,”
in 2009 IEEE Conference on Computer Vision and Pattern Recognition, June 2009, pp. 413–420.

[20] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, “Places: A 10 million image database for scene recognition,” IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017.

[21] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, Deep learning. MIT press Cambridge, 2016, vol. 1.

[22] D. Garcia-Gasulla, A. Vilalta, F. Parés, J. Moreno, E. Ayguadé, J. Labarta, U. Cortés, and T. Suzumura, “An out-of-the-box full-network embedding for convolutional neural networks,” CoRR, vol. abs/1705.07706, 2017. [Online]. Available: http://arxiv.org/abs/1705.07706

[23] P. J. Rousseuw, “Silhouettes: a graphical aid to the interpretation and validation of cluster analysis,” Journal of computational and applied mathematics, vol. 20, pp. 53–65, 1987.

[24] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[25] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in IEEE CVPR, 2009, pp. 248–255.

[26] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600–612, April 2004.

[27] N. X. Vinh, J. Epps, and J. Bailey, “Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance,” Journal of Machine Learning Research, vol. 11, no. Oct, pp. 2837–2854, 2010.

[28] L. Hubert and P. Arabie, “Comparing partitions,” Journal of classification, vol. 2, no. 1, pp. 193–218, 1985.

[29] A. Rosenberg and J. Hirschberg, “V-measure: A conditional entropy-based external cluster evaluation measure,” in Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL), 2007.

[30] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, “Deeply-supervised nets,” in Artificial Intelligence and Statistics, 2015, pp. 562–570.

[31] T. Tieleman and G. Hinton, “Lecture 6.5-rmsprop, coursera: Neural networks for machine learning,” University of Toronto, Technical Report, 2012.

[32] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1026–1034.