Modified MNIST dataset: classification approaches

Group 69 - Daniel Borisov[1], Jonas Lehnert[2], and Bianca Granato[2]

Abstract—Computer vision is an important field through which many tools can be developed and insights gained. From motion capture, to fingerprint matching, the potential impact of highly accurate computer vision systems are great. Machine learning has been a huge boom for the field of computer vision research, and the use of receptive fields in convolutional neural networks has allowed for massive improvements in computer capabilities in regards to image classification. In order to explore the potential for machine learning in computer vision, we applied convolutional neural networks (CNN) to a modified MNIST data set. The MNIST set, which contains handwritten numbers between 0 and 10, has served as the 'hello world' of neural networks. Working with a modified MNIST set, containing three digits overlayed on top of a random background, we explored different CNN-based approaches to identify the largest digit in each image. We developed both *de novo* CNN and also used transfer learning i.e. a pre-trained image classification algorithm. We found that both models worked well at both generalising and classifying the data set, and found that the transfer learning model outperformed a newly trained CNN. Though we obtained high training and validation accuracy using both approaches, our results were sub par compared to other Kaggle competition submissions. Thus, we hope to improve our results and further apply a variety of different network architectures, such as a ResNet, in the future.

1 Introduction

Computer vision is a subset of computer science concerned with extracting information from images or videos. It is a highly interdisciplinary field with broad applications ranging from object tracking to scenery reconstruction from partially overlapping images (such as panoramic photos). Tools and techniques from this field are central to Optic Character Recognition (OCR), motion capture for animated movies, fingerprint matching, among others [1].

Computer vision has seen great advances since David Marr's influential 1982 paper on visual representation [2], especially over the past decade. Such unprecedented progress has been possible in part due to the availability of databases with millions of high-quality, labeled images. These data sets offer a common benchmark against which researchers can test newly developed algorithms and standardize best practices [3], [4]. Some of the most exciting developments in computer vision arose during the ImageNet annual object classification challenge, where image classifiers had to automatically label everyday objects, animals or scenery into one of 1000 categories. In particular, Graphic Processing Unit (GPU)-computing and deep neural networks were popularized in the 2012 competition and have since become standard in the field [5]. Modern computer vision research continues to use various deep learning constructs, especially convolutional neural networks (CNNs). Over time, these models have become more and more complex and can even be used to approximate human behavioural states such as attention [6].

In this exploratory study, we revisit the Modified National Institute of Standards and Technology (MNIST) data set, which was originally published in 1998 [7]. Here we use a modified version of the MNIST data set in each image contains 3 hand-written numbers[1]. Here, we attempt to use deep-learning tools to identify the highest-magnitude number presented in each image. We display several CNN architectures and also compare these to transfer learning models. Our results show that a CNN with a 7-layer structure, using rectified linear unit activation functions and a 3-layer dense architecture performs with a 95.5% accuracy. In contrast, pre-trained models such as VGG-16 perform with 96.3% validation accuracy.

2 Data set

The data consisted of 50,000 images of dimensions 128 x 128; pixel values ranged from 0 to 255. Each image contained three hand-written digits from the MNIST data set on a non-uniform background. Our goal was to predict the highest digit in each picture, with labels ranging from 0 to 9. For all experiments, the data was split into 80% training and 20% validation, unless otherwise indicated. Since the goal of the task was to find the largest number in the image, the distribution of labels is heavily skewed, as can be seen in figure 1. We opted to maintain the original data distribution for training and validation, though other approaches could have been utilized as discussed in [8].

3 Proposed Architecture

3.1 Convolutional Neural Network

CNNs provide a way for neural networks to take spatial relations into account when analyzing data sets with a dimensionality greater than one. By applying convolutional filter masks, with the convolutional filter defined by the weights to a kernel of a specific size, CNNs can create receptive field-style feature detectors for image classification tasks.

1. https://www.kaggle.com/c/modified-mnist

[1] *Department of Physiology, McGill University*
[2] *Quantitative Life Sciences, McGill University*
Project submitted November 13, 2019

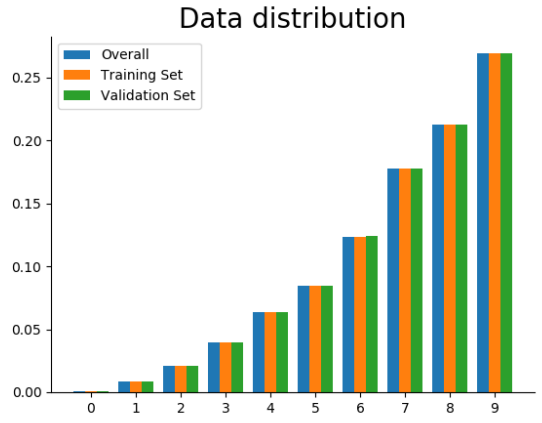

Fig. 1. Density of labels in the training, validation, and combined data sets. Values range from 0 to 9.

Due to the performance of convolutional neural networks when applied to the original MNIST data set classification problem, we decided to initially design a very simple CNN that would perform well on the original MNIST data set. This model consisted of two convolutional 32-neuron layers, after which the outputs were pooled and relayed into two 64-neuron layers. The convolutional layers were followed by a 1024-10-neuron dense layer structure. The model performed with 99% on the original MNIST data set, but could only provide accuracies of about 25% on the modified MNIST data. This simplified model served as our initial starting point from which we designed a more complex model that could more accurately classify the modified MNIST data-set, which incorporates various backgrounds images.

As shown in Fig. 2, this simplified model over-fit the data set, showing increases in model training accuracy, while showing concurrent decreases in validation accuracy, where the over-fitting was likely due to fitting to the background.

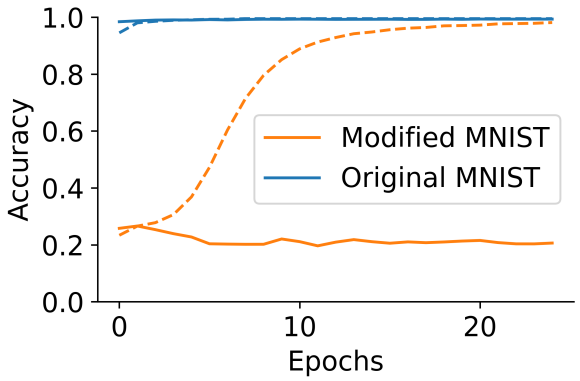

Fig. 2. Training and validation results of a simple convolutional neural network on both the original MNIST and the modified MNIST data sets. Training accuracy shown in dashes and validation accuracy shown as solid lines.

Therefore, we proposed that a more complex multi-layered convolutional neural network algorithm, utilising anti-overfitting measures such as regularization or dropout, would be able to correctly classify the modified MNIST data set when tested on both the training and validation sets.

The proposed model followed a 3-2-2 64xConv-Pool>128xConv-Pool>256xConv-Pool structure, followed by a flattening layer and a 512-node dense layer projecting to a 10-node output layer. Batch normalization was applied to the node outputs before being sent through a rectified linear unit (ReLU) activation function. Spatial 2D dropouts were tested at the convolutional layers and regular dropouts at the dense layer. L1 and L2 regularization was also implemented at each layer and performances compared to determine the best way to generalize the model to the data set. An extra dense layer after the 512-node dense was also added and its performance compared. A visual representation of the model is provided in Fig. 4 (Right).

It has previously been described that although the *Adam* adaptive optimiser shows superior training convergence, simple stochastic gradient descent (SGD) is able to produce better generalization after longer training sessions [9]. Thus, three different optimisers were also tested for the ability to fit the model.

3.2 Transfer Learning

For this comparative study we also implemented a different CNN classifier variations utilising the VGG 16 transfer learning algorithm. Most state-of-the-art image classifiers are learning-based and require large amounts of data and time for training. To reduce these requirements, it is possible to apply pre-trained algorithms and fine-tune their weights through a process called *transfer learning* [10]. This technique has been particularly popular in computer vision applications with success in localizing animals in ecology [11], classifying tumours in biomedicine [12], and target identification for safety systems [13], among others.

3.2.1 VGG-16

VGG-16 is a deep CNN with 16 weight-layers that accepts 224 x 224 colour images as input. It was trained to classify ImageNet images over 2-3 weeks on a 4-GPU system [14]. The trained network is available through the open-source library *Keras* and can be fine-tuned to learn new categories. When fine-tuning a network, layers whose weights will not be trained, such as general feature detection layers, should be frozen. We should also change the number of nodes in the final layer to correspond to the number of categories in the new data set. For the modified MNIST challenge, we had 10 different possible outcomes, and changed our final layers to match this output.

To use VGG-16, we first imported the pre-trained model without the top layers, specified input size and froze its first 5 layers. We then added four new layers at the end, including a classification layer specific for this task. Two versions of the model were created: one in which the input size was padded to correspond to the base model's input size (224 x 224 x 3) and another in which the input layer was modified to accept the modified MNIST images (128, 128). In both cases, images had to be converted from grey to RGB values by duplicating the 2D matrix three times, one for each RGB channel. In addition to padding, the augmented images were also thresholded to remove all values below 90% of the

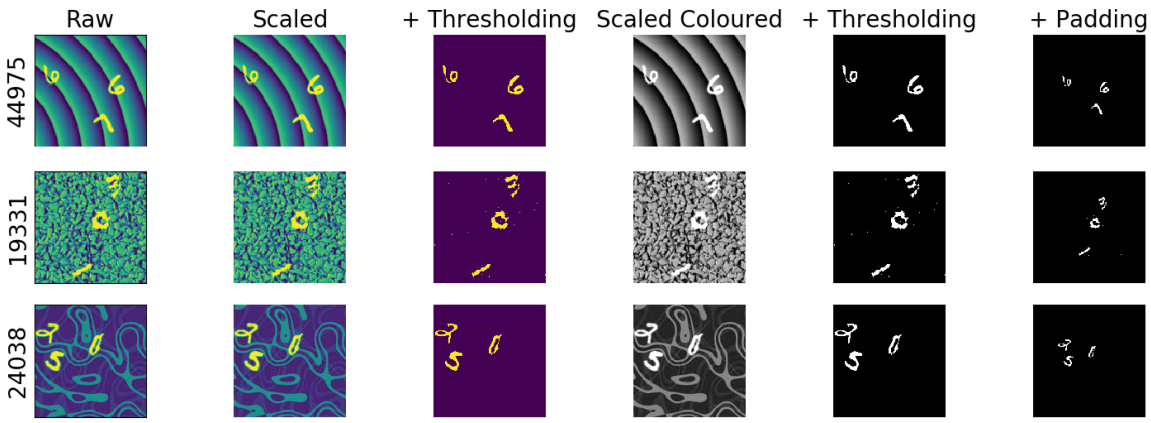

Fig. 3. Feature augmentation for images number 19331, 24038, and 44975. Raw images can be seen on the leftmost column. Scaled images were normalized by the maximum gray-value (255) with little to no loss of information. Thresholding was done by setting all values below 0.9 to zero. Finally, the images were centered and zero-padded, increasing their size from 128 x 128 to 224 x 224. Columns 1-3 were pseudocoloured using *matplotlib*'s default colourmap, and columns 4-6 had three channels (RGB).

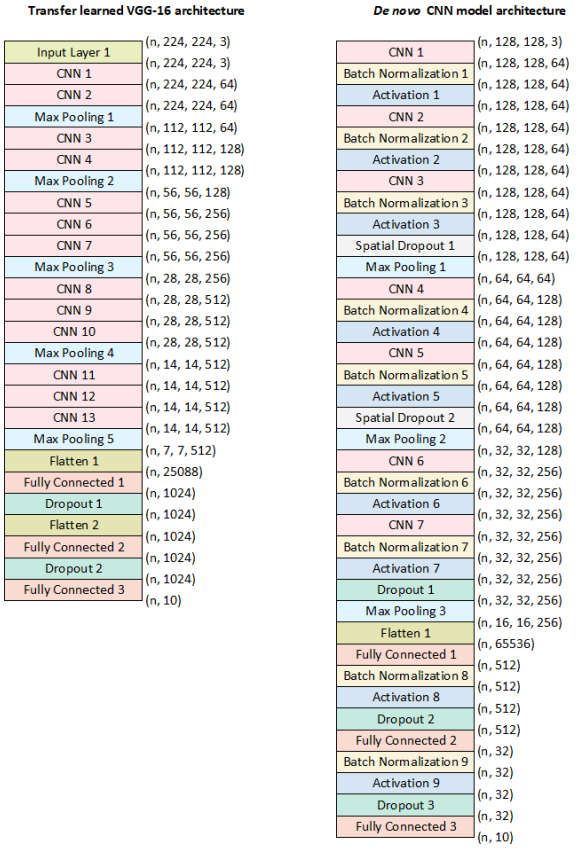

Fig. 4. **Left**: Architecture of one of the two VGG-16 used in this study after transfer learning. **Right:** Architecture of one of the CNN models. Input/output size for each layer is shown, n represents the number of images.

gray values and scaled to be between 0 and 1 by dividing by 255. An example of the augmentations performed can be seen in figure 3. The architecture for the transfer learned VGG-16 model can be seen in figure 4.

4 Results

4.1 CNN

The CNN used to classify the modified MNIST data set was trained with various modifications to determine the best model. The first parameter that was varied was the optimiser used to fit the model, where ADAM, SGD, and SGD with Nesterov momentum were tested, and the results of the training shown in Fig. 5. The results confirmed previous research that showed that while the Adam algorithm is better at faster convergence (Left), the SGD algorithm is able to converge to a higher validation accuracy (Right), and is thus better at generalizing the data set.

Furthermore, various structural and parametric changes were applied to the model in order to test their impact on the model's ability to classify the modified MNIST data set. As mentioned in the discussion on the proposed design, a simple CNN failed to adequately generalize the data and overfit to the background. As a result, it was proposed that the implementation of dropout layers would help prevent the model from relying on specific connections and overfitting to those weights, allowing it to generalize the results better. The implementation of the dropout, visualized in Figure 4 (Right), was done through two spatial dropouts at the first two convolutional sections, with a dropout probability of 0.2, and three normal dropouts at the last convolutional section, and finally a dropout after the two dense layers, with a dropout probability of 0.2, 0.5, and 0.5, respectively. The results comparing the addition of the dropout layers to a model without dropout is shown in Fig. 6 (Top), showing that the addition of dropout reduces training accuracy and convergence, but eventually improves the validation accuracy.

In order to try and circumvent any over-fitting that might have been still occurring, both L1 and L2 regularization was implemented. Both implementations utilised a regularization factor of 0.0001, and the training and validation results are shown in Fig 6 (Middle). The results display that implementation of L1 regularization reduced both training and validation accuracy, while L2 regularization was comparable to a non-regularized model, where the non-

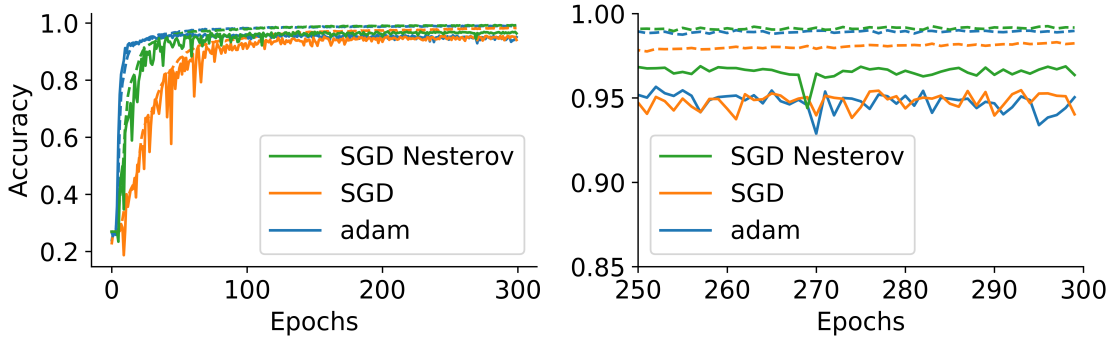

Fig. 5. Comparison of three different optimizer algorithms for fitting the model. The ADAM optimizer, stochastic gradient descent, and stochastic gradient descent with Nesterov momentum were compared. Training accuracy in dashed, validation accuracy in full line. **Left**: Training shown over all 300 epochs. **Right**: Training shown for the last 50 epochs

regularized model showed a slight increase in total training and validation accuracy.

Finally, the model architecture was modified to only include one dense layer after convolutional flattening, instead of two. Two variations of this modification, the first in which the same 512 nodes were used on the dense layer, and one in which the nodes were increased to 1024, were compared to the initially proposed model. The results of the training and validation, shown in Fig. 6 (Bottom), demonstrate that implementing two dense layers on the output reduced model stability during training, reduced training accuracy, and reduced convergence speed when compared to the single dense layer models, though validation accuracy was equivalent across all three. When comparing the 1024 and 512 node output models, it was found that the 512 node converged faster than the 1024 node, but was otherwise equivalent in the training and validation accuracy.

Thus it was determined that for a novel implementation of a convolutional neural network, a model utilising dropout, no regularisation, and a single dense hidden layer consisting of 512 nodes, trained using a SGD optimizer with Nesterov momentum, was the ideal CNN model for the modified MNIST data set max value classification problem.

4.2 Transfer Learning

Transfer learning, utilising the pre-trained VGG16 image classification model, was performed on two different data sets: the raw set, and a pre-processed augmented set. The model was trained over five epochs, since training it for longer could cause it to overfit. Model performance is shown in Fig. 7. It can be seen that both models perform extremely well, even after only a single training epoch, and can achieve an almost perfect training accuracy. When comparing the model that was fit to the augmented data set compared to the raw data set, it was found that the augmented data set (described in section 3.2) performed better, as it had a greater validation accuracy when compared to the performance of the model with fit to the raw data set.

4.3 Comparison of algorithm performance

The two different approaches were compared by looking at the performance of the best implemented CNN model and the best transfer learning VGG16 model. It was found that when looking at training, validation, and test set accuracies, the transfer learning model was able to slightly outperform the novel CNN model on all metrics. Similarly, when looking at other metrics, such as classification accuracy stability over epochs, convergence time, and generalisability, the transfer learning model outperformed the novel CNN implementation.

4.4 Kaggle Submission

The Kaggle competition test set served as a secondary validation step for comparing the trained models, and for selecting the best model. Due to the limited submission count of the Kaggle competition, the validation accuracy was used as the main metric for determining the best models to compare, and then the Kaggle test set was used to compare the final models produced. It was found that the transfer learning model performed the best on both the validation and test sets, and as a result was the predictions used for the Kaggle competition. The results indicated that for a subsection of the test set, the transfer learning model was able to achieve an accuracy of 96.3%. This result placed 64th out of 98 groups. Therefore it can be concluded that while our model performs really well for the classification, there is a lot of room for improvement, as some of the competition models were able to achieve a test set accuracy of over 99%.

5 Discussion

We were able to demonstrate that by utilising convolutional neural networks, it was possible to achieve modified MNIST data set classification accuracies exceeding 95%. Furthermore, when modifying CNN architectures and parameters, it was found that there exists tradeoffs between convergence speed, generalisation of the model to prevent over-fitting, and per-epoch stability.

The results of our experiments showed that dropouts can help increase generalisability, while coupling regularisation and dropouts could prove excessive and end up reducing training accuracy without an improvement on the validation accuracy.

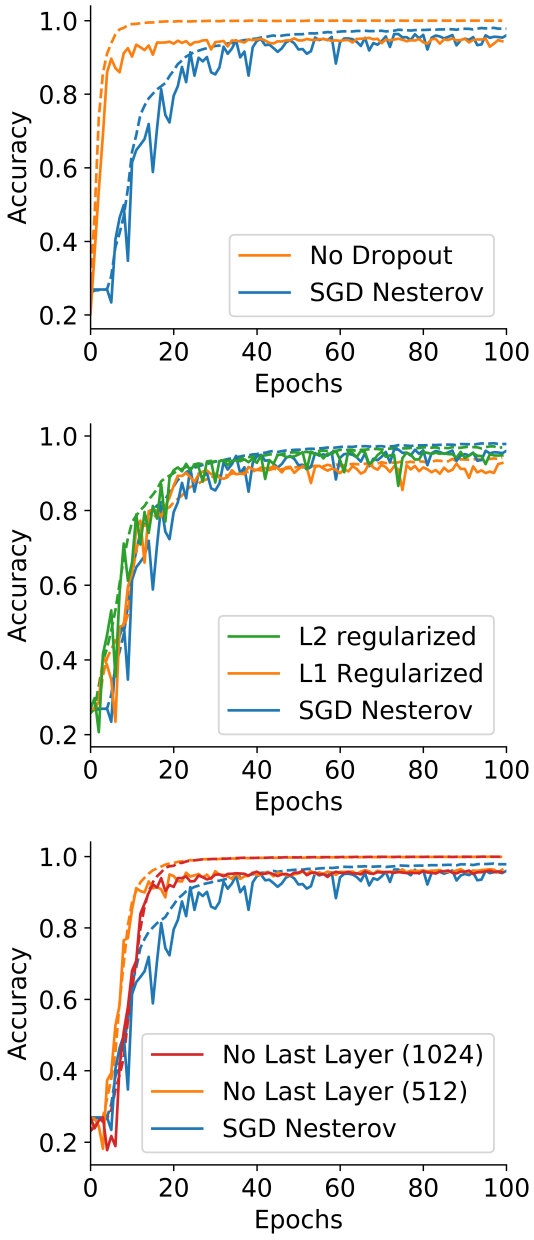

Fig. 6. Comparison of different structural and parametric changes on the performance of the model at correctly predicting the proper result. Training accuracy in dashed, validation accuracy in filled-in line. **Top**: Influence of the addition of dropout to the model, where the convolutions have a spatial dropout and the dense layers have a standard dropout. **Middle**: Influence of adding L1 or L2 regularization to the model. **Bottom**: Influence of utilising a 1024-node last layer, a 512-node last layer, or a 512-node layer followed by a 32-node layer.

Furthermore, our tests were also able to validate previous studies that showed that while the ADAM optimisation algorithm is superior for training speed, it fails to achieve the validation accuracy of a simple stochastic gradient descent algorithm that utilises Nesterov momentum when trained over enough epochs.

Future directions for attempting to improve on the validation and test accuracies in order to try and achieve the classification performance seen in the top groups of the Kaggle competition includes: more modifications to our CNN model architecture, testing other transfer learning models, and the implementation of other image classification architectures.

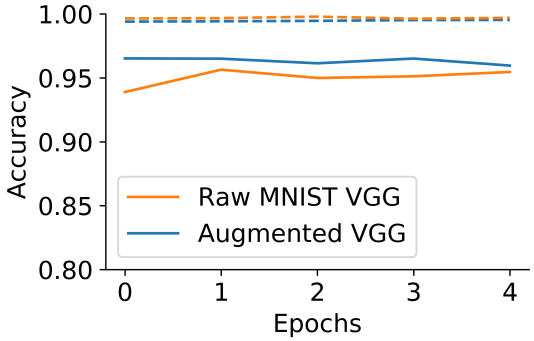

Fig. 7. Performance of the VGG16 transfer learning model to the classification of the modified MNIST data. A VGG16 fed raw data, and a VGG16 fed augmented data are compared. Training accuracy in dashed, validation accuracy in full line.

Modifications to our CNN architecture that would be worth exploring include variations of the convolutional kernel sizes, increasing convolution-pooling structures, and increasing filter counts for the convolutional layers. With more time to be able to train these models, we would be able to test more variation to find a model that works even better. Furthermore, by utilising better hardware, future work could include training the network for more epochs, as the transfer learning models were trained on multi-GPU systems for long periods of time. It is possible that our CNN models would have performed better than the transfer learning models had we the opportunity to run for more epochs.

Future directions also include using different transfer learning models to see if other implementations might perform better. This includes other models such as deeper VGG models.

Finally, another step could be the implementation of a new model architecture such as the Residual Neural Network (ResNet) for image classification. ResNets have been found to be successful image classifiers [15], and the implementation of such could improve our ability to classify the modified MNIST sets to competitive levels seen by the accuracies achieved in the top performing Kaggle groups.

6 Statement of Contributions

Jonas implemented the simple CNN model, aided in experimental design and produced figures in this report. Daniel implemented the high-performing CNN model, designed and performed computational experiments. Bianca implemented transfer learning of VGG16 and made schemes outlining our model structures. All authors contributed towards the project write-up.

REFERENCES

[1] R. Szeliski, Computer vision: algorithms and applications. Springer Science & Business Media, 2010.

[2] D. Marr, Vision: A computational investigation into the human representation and processing of visual information. WH San Francisco: Freeman and Company, 1982.

[3] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," International Journal of Computer Vision, vol. 111, no. 1, pp. 98–136, 2015. [Online]. Available: https://doi.org/10.1007/s11263-014-0733-5

[4] J. Deng, W. Dong, R. Socher, L. Li, L. Kai, and F.-F. Li, "Imagenet: A large-scale hierarchical image database," in 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, Conference Proceedings, pp. 248–255. [Online]. Available: https://ieeexplore.ieee.org/document/5206848/

[5] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," International Journal of Computer Vision, vol. 115, no. 3, pp. 211–252, 2015. [Online]. Available: https://doi.org/10.1007/s11263-015-0816-y

[6] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-alone self-attention in vision models," arXiv preprint arXiv:1906.05909, 2019.

[7] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[8] C. Huang, Y. Li, C. Change Loy, and X. Tang, "Learning deep representation for imbalanced classification," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 5375–5384.

[9] N. S. Keskar and R. Socher, "Improving generalization performance by switching from adam to sgd," arXiv preprint arXiv:1712.07628, 2017.

[10] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, "Boosting for transfer learning," in Proceedings of the 24th international conference on Machine learning. ACM, 2007, Conference Proceedings, pp. 193–200.

[11] A. Fritzler, S. Koitka, and C. M. Friedrich, "Recognizing bird species in audio files using transfer learning," in CLEF (Working Notes), 2017, Conference Proceedings.

[12] A. Menegola, M. Fornaciali, R. Pires, F. V. Bittencourt, S. Avila, and E. Valle, "Knowledge transfer for melanoma screening with deep learning," in 2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017). IEEE, 2017, Conference Proceedings, pp. 297–300.

[13] E. Alhadhrami, M. Al-Mufti, B. Taha, and N. Werghi, "Learned micro-doppler representations for targets classification based on spectrogram images," IEEE Access, vol. 7, pp. 139 377–139 387, 2019.

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

[15] 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.