Assignment 2: Deep Neural Networks and CNNs for Computer Vision

Claire Boetticher, MSDS 458, SEC 56

Colab, Experiments 1-4: https://colab.research.google.com/drive/13P3NNINbL0-iL2xAuwDnpVE-81TC74i5?usp=sharing

Colab, Experiment 5: https://colab.research.google.com/drive/14rqoY5VKccLeeXAMTI6uVOpQNjwFrbf9?usp=sharing

Abstract

Deep learning neural networks provide powerful advancements in a variety of challenging problems, for example image classification. Where simple neural networks have been found to perform relatively well with certain tasks, such as digit classification, images from photos have proven much more challenging given the astounding variety in how objects, even of the same class, can look vastly different depending on size, shape, detail, and positioning (to name only a few). The Fashion MNIST database, with labeled articles of clothing across ten categories, offers a more complex dataset for benchmarking complex dense and convolutional neural networks and how they compare in classification accuracy and process time. This study explores varying architectures of both DNNs and CNNs with varying layer depths and widths with the objective of configuring effective models. Specific hyperparameter settings are adjusted to evaluate potential effects on classification performance and process time as well, employing visualizations to reflect performance of each model across training epochs. Greater understanding of the intricacies of these architectures and the numerous ways they can be adjusted lead to more performant models more likely to work well in real-life image classification scenarios, with applications in retail and far beyond.

Introduction

This study is a comparative exploration of dense and convolutional neural networks for image classification, with a two-fold goal: develop a neural network model for highly-accurate and generalizable classification and gain an in-depth understanding of how various factors affect fitting and ultimate test set performance with networks of differing topologies and hyperparameter settings. This will shed light on how hidden nodes learn to extract features from inputs; with additional layers, the objective is to better understand how each successive layer extracts progressively abstract and generalized features. Given the ever-present challenge posed by fitting these models and determining an ideal tradeoff between complexity and generalizability, methods for addressing over- and underfitting are employed as well and evaluated for impact on performance with training, validation, and test data. While the Fashion MNIST database contains ten relatively specific image types, this study offers insights into designing models for more complex and varying image classification tasks, for example facial recognition.

Literature review

Pioneering computer vision work from Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton (2012) is still seen as enormously influential with its record-breaking classification accuracy and relatively simple architectural layout of convolutional, max pooling, dropout, and fully-connected layers. With this ILSVRC (ImageNet Large-Scale Visual Recognition Challenge) submission came a host of subsequent ground-breaking CNN developments and submissions, including Matthew Zeiler and Rob Fergus (2013) and Karen Simonyan and Andrew Zisserman (2014). Building on these developments in network topologies, more recent work has explored the numerous hyperparameters available to train deep neural networks. Dmytro Mishkin, Nikolay Sergievskiy, and Jiri Matas systematically evaluate some key

components of the hyperparameter space, such as pooling variants, classifier design, image pre-processing, and learning parameters. Guangyong Chen, Pengfei Chen, Yujun Shi, Chang-Yu Hsieh, Benben Liao, and Shengyu Zhang discuss a novel approach to applying batch normalization and dropout with the addition of an Independent Component (IC) layer (2019).

Methods

The Fashion MNIST dataset contains 70,000 gray-scale images of clothing products from Zalando, a European online fashion platform, each labeled with an integer from zero to nine representing ten product classes: 'T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat', 'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot', respectively. Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it from 0 to 255, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. The label distribution in the training and testing sets appears relatively balanced across zero to nine, which obviates the need for adjustment by up-sampling, down-sampling, or other methods. The 70,000 images are divided into a set of 60,000 training images (of which 5,000 are held back for validation) and 10,000 test images, then reshaped and normalized. The study consists of five separate experiments, with varying neural networks evaluated for performance (measured by classification accuracy) and processing time. Architectures are defined as follows:

Type of Neural Net	Network Architecture	Trainable Parameters
DNN 1	1 hidden layer (5 nodes)	3,985
DNN 2	2 hidden layers (5 / 10 nodes)	7,965
CNN 3a	2 convolutional layers (32 and 64 filters) 2 max pooling layers	225,034
CNN 3b	2 convolutional layers (32 and 64 filters) with L2 regularization 2 max pooling layers	225,034
CNN 3c	Image augmentation 2 convolutional layers (32 and 64 filters) 2 max pooling layers	225,034
CNN 4a	3 convolutional layers (32 and 2x64 filters) 3 max pooling layers	93,322
CNN 4b	3 convolutional layers (32 and 2x64 filters) with L2 regularization 3 max pooling layers	93,322

All experimental networks consist of 784 input nodes, varying neural layers, and 10 output nodes corresponding to the ten fashion categories. Dense neural network 1 serves as a baseline model with one hidden layer. Model 2 extends the baseline with the addition of 1 additional hidden layer. Model experimentation in experiment 3 establishes a baseline convolutional neural network with 2 convolutional and max pooling layers. Employing L2 regularization, which constrains the original loss function, and image augmentation, which generates “new” training samples by introducing noise in an attempt to force the network to learn more robust features, enables an examination of effects on model fit, classification performance, and process time. Given the potential of overfitting with this sizeable increase in layer width, and thus tunable parameters, attention is given to that scenario in model performance with both training and validation

data. Experiment 4 evaluates deeper CNNs with the addition of a convolutional and max pooling layer. Again, L2 regularization is employed to assess its potential to address overfitting and produce a more generalizable model. Experiment 5 repeats all experiments 1 through 4 with dropout on hidden layers to determine the effect of reducing model complexity and mitigating overfitting; model configurations from experiments 1 through 4 are maintained to enable a more accurate comparison. The following hyperparameters are employed across all experiments:

- Batch size: 100 / Epochs: 20
- Activation: ReLU
- Optimizer: Adam
- Loss function: sparse categorical cross-entropy

Results

The table below shows model fitting and evaluation results :

	model	loss	accuracy	val_loss	val_accuracy	training time (sec)	testing time (sec)
0	DNN (2 layers)	0.445179	0.845818	0.446092	0.8482	33.456518	0.959970
1	DNN (3 layers)	0.393804	0.862691	0.418285	0.8578	33.351694	0.907039
2	CNN (2 layers)	0.062694	0.977255	0.342600	0.9102	46.694116	1.092569
3	CNN (2 layers L2)	0.520660	0.857909	0.493949	0.8732	43.898222	1.081623
4	CNN (2 layers aug)	1.025814	0.621236	0.765337	0.7222	436.083312	1.024631
5	CNN (3 layers)	0.082464	0.969909	0.314664	0.9150	44.815734	1.112644
6	CNN (3 layers L2)	0.589668	0.846236	0.581835	0.8528	48.277778	1.097991
7	DNN (2 layers DO)	1.039818	0.627073	0.687543	0.8116	25.842328	0.654728
8	DNN (3 layers DO)	0.979232	0.608491	0.672728	0.7602	25.392829	0.637817
9	CNN (2 layers DO)	0.146241	0.944364	0.230351	0.9198	43.519450	0.802108
10	CNN (2 layers L2 DO)	0.593767	0.836236	0.556045	0.8532	40.009027	0.826110
11	CNN (2 layers aug DO)	0.986620	0.632764	0.713365	0.7226	274.081636	0.759575
12	CNN (3 layers DO)	0.204254	0.922327	0.212962	0.9220	41.460279	0.794109
13	CNN (3 layers L2 DO)	0.674064	0.819782	0.629076	0.8384	42.996661	1.006160

The simple dense neural networks without dropout performed decently well given the simplicity of the models, especially compared to random chance of correct classification at 10 percent (see Appendix, Figures 1 and 2). Applying dropout to these models decreased their performance noticeably. The two-layer CNN with no dropout performed best in terms of training and validation accuracy and test performance, with relatively fast process time using GPU acceleration, closely followed by the three-layer CNN with no dropout (Figure 3). The two-layer baseline CNN performed slightly better than the baseline three-layer CNN (Figure 6). The closeness in performance metrics suggests that for this specific data, a deeper CNN architecture does not provide any notable advantage and may, in fact, add complexity that precludes a more generalizable model for unseen data. Potential overfitting is seen in the three-layer DNN and in the two- and three-layer CNNs, evidenced by the rise in validation loss mid-way through training (Figures 2, 3, 6). In terms of attempts to mitigate overfitting, L2 regularization lowered accuracy where applied but did mitigate some overfitting with CNNs (Figures 4 and 7). Data augmentation from Experiment 3, with and without dropout, decreased performance

drastically with much longer process time (Figures 5 and 12). Dropout seemed to lower performance across all experiments while speeding process time up very minimally. For this dataset and the specific configuration of these experiments, efforts to improve performance and generalizability with regularization, augmentation, and dropout did not provide those improvements in experimental models although L2 regularization did reduce overfitting at the minor expense of classification accuracy.

Conclusions

For the computer vision task at hand, with classification of clothing products for an online retailer, results suggest that simple CNNs offer accuracy in low 90 percentages and reliable performance on unseen test data, all with process times only slightly higher than dense neural networks (at least on these experiments conducted with GPU acceleration). The marked improvement in performance seen with CNNs and the relationships this study's CNNs have appeared to learn across these experiments suggests their utility for more complex computer vision tasks, such as facial recognition. Where the dense neural networks are treating all 784 pixels of each clothing item image as independent of each other, spatial correlations matter with CNNs. The filters in the convolutional layers are behaving as feature identifiers (e.g., features of straight edges, curves, and colors). The max pooling layers then summarize the presence of those features, culminating in the fully connected layer at the end of the network. This architecture's potential for accuracy and generalizability seems highly preferred over the simple dense network, suggesting the CNN's ability to learn features from the pixel inputs. The spatial correlation of features extracted and summarized across the CNN are preserved in a way that seems to more accurately identify each image's complexity, where the shape of sleeves or the sole of a shoe are "understood" in a way. This is a drastically more challenging task than classifying an item of clothing by its type: all faces are a type of human face, with certain features that all humans generally have (two eyes, two ears, a nose, a mouth, a chin). The computer vision task in facial recognition is identifying one face as belonging to one specific person, for authentication of devices for example. Depending on the application of the recognition system, training a model for this kind of task may require large amounts of training data (if possible) for each face, introducing noise if possible to attempt ensuring more robust feature identification; noise could amount to a data augmentation process, where a person's face seen at different angles and positions would improve the model's ability to recognize the face in different lights and with different backgrounds that may prove distracting. Image augmentation did not improve performance with the Fashion MNIST images, possibly because of the relative uniformity and quality of these images as is (assuming they were taken professionally for a retail site, with the express purpose of being clear to a user); with facial recognition systems trained on more varied photos, the augmentation could contribute to improved performance. Widely-available pre-trained models, such as FaceNet from Google, could serve as a useful baseline. Validation from users would be needed to score how well a system recognized the face with "enough" accuracy. Convolutional layers would be advantageous in a large-scale recognition scenario due to their ability to recognize the more meaningful features and how they map to other features (i.e., a person's nose's distance to each eye), likely leading to more accurate classification of that image correctly as associated with a specific person. A model employing convolutional layers would perform much better with extensive training data; barring that, a sufficient base of training data with image augmentation applied and additional mitigation of fitting issues could possibly suffice (again, depending on the

application for deployment). For an individual cell phone user, convolutional layers may add unnecessary complexity. It would be a relatively safe assumption that the cell phone owner would be the primary person trying to authenticate. In that case, the classification task would be closer to a binary case: the owner's face would need to be recognized with high accuracy (given owner training images) and then "not the owner" on the other hand. With sufficient training data for each class (images of the primary user and a varied sample of non-users), an accurate system could be devised at a smaller and less process-intensive scale.

References

Chen, G., Chen, P., Shi, Y., Hsieh, C., Liao, B. and Zhang, S. (2019). Rethinking the Usage of Batch Normalization and Dropout in the Training of Deep Neural Networks. https://arxiv.org/abs/1905.05928

Krizhevsky, A., Sutskever, I., and Hinton, G. (2012). ImageNet Classification with Deep Convolutional Neural Networks. https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

Mishkin, D., Sergievskiy, N., and Matas, J. (2016). Systematic Evaluation of CNN Advances on the ImageNet. https://arxiv.org/abs/1606.02228

Simonyan, K. and Zisserman, A. (2014). Very Deep Convolutional Neural Networks for Large-Scale Image Recognition. https://arxiv.org/pdf/1409.1556v6.pdf

Zeiler, M. and Fergus, R. (2013). Visualizing and Understanding Convolutional Neural Networks. https://arxiv.org/pdf/1311.2901v3.pdf

Appendix

Figure 1: Learning curves from model 1 (DNN 2 layers)

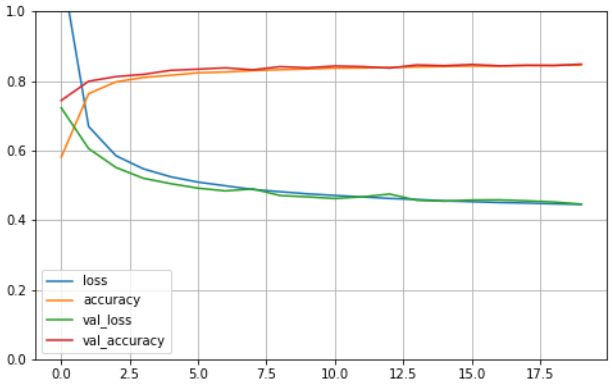

Figure 2: Learning curves from model 2 (DNN 3 layers)

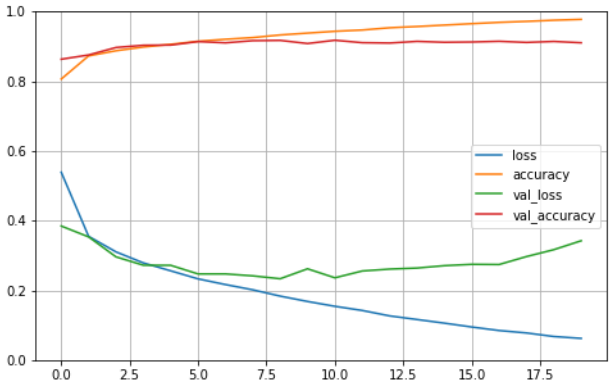

Figure 3: Learning curves from model 3 (baseline CNN 2 layers)

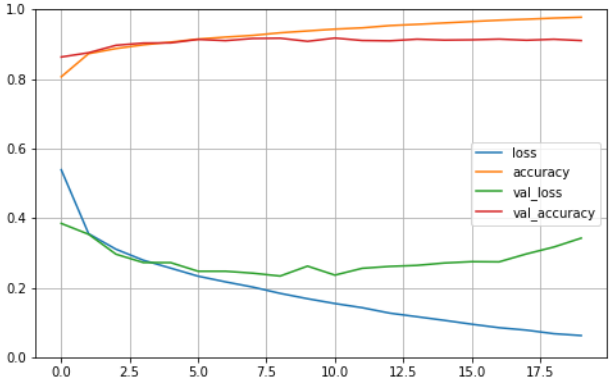

Figure 4: Learning curves from model 3b (CNN with L2 regularization)

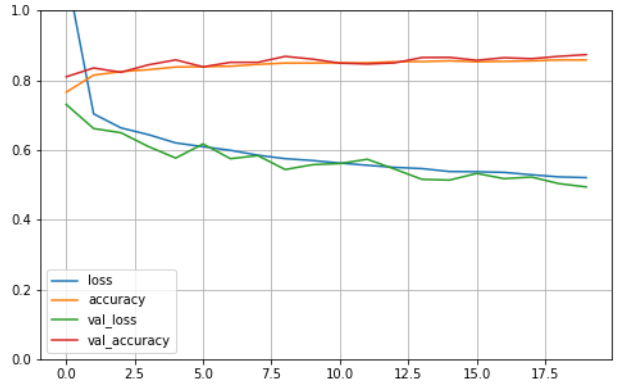

Figure 5: Learning curves from model 3c (CNN with image augmentation)

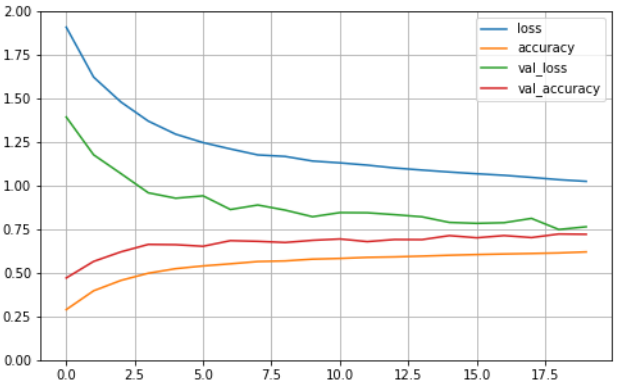

Figure 6: Learning curves from model 4a (baseline CNN with 3 layers)

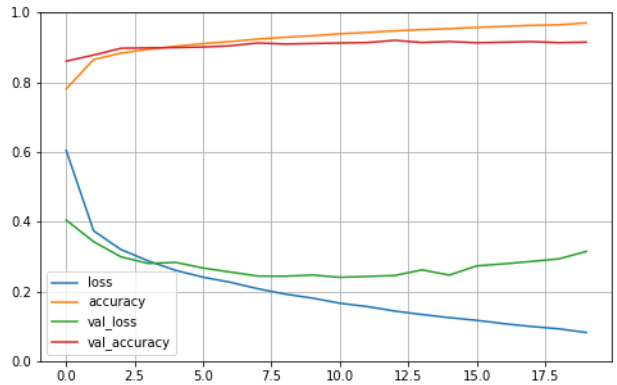

Figure 7: Learning curves from model 4b (CNN with L2 regularization)

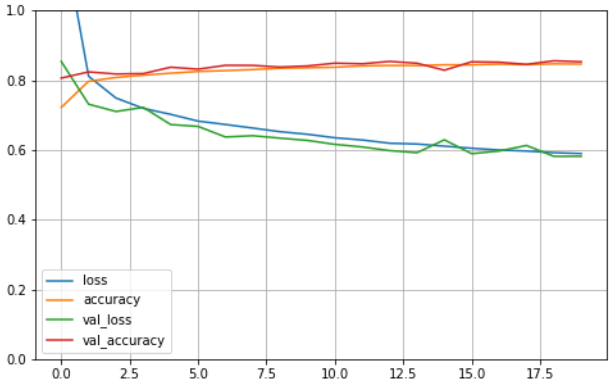

Figure 8: Learning curves from model 1 (DNN 2 layers with dropout)

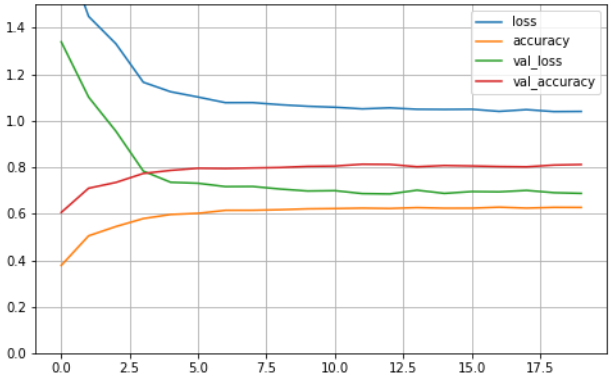

Figure 9: Learning curves from model 2 (DNN 3 layers with dropout)

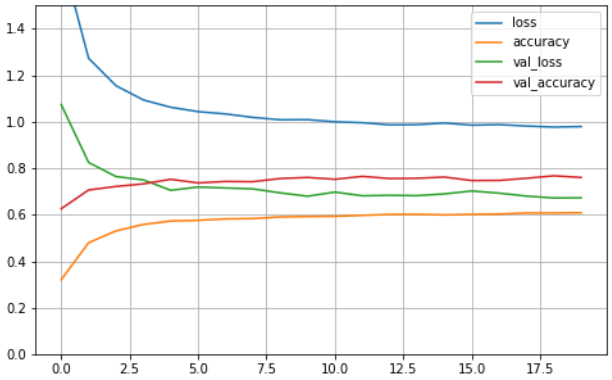

Figure 10: Learning curves from model 3a (baseline CNN with 2 layers + dropout)

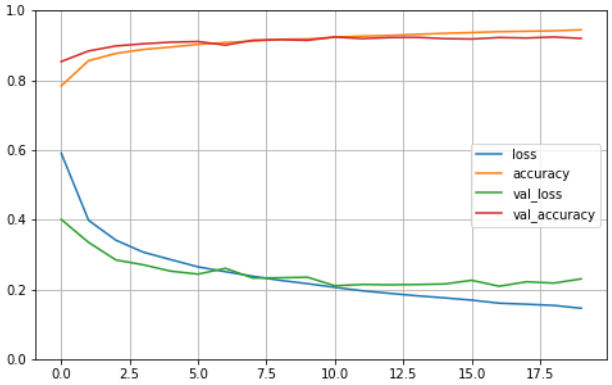

Figure 11: Learning curves from model 3b (CNN with L2 regularization + dropout)

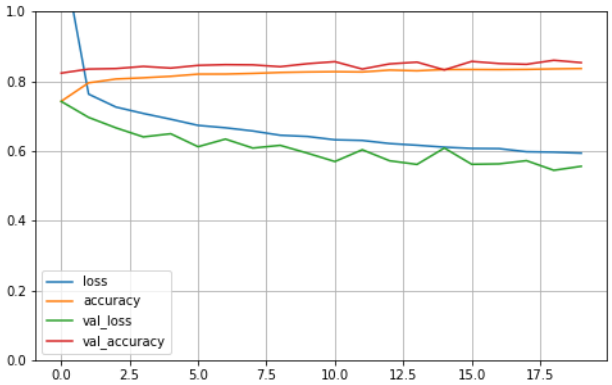

Figure 12: Learning curves from model 3c (CNN with image augmentation + dropout)

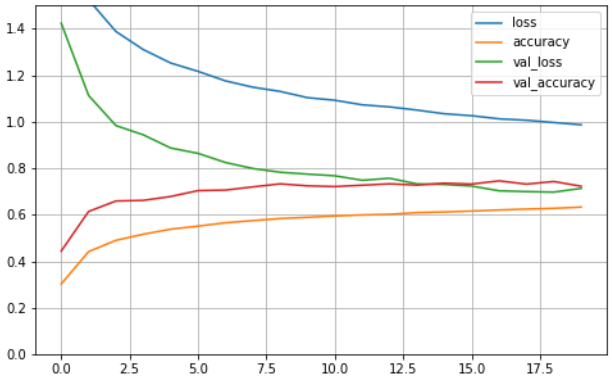

Figure 13: Learning curves from model 4a (baseline CNN with 3 layers + dropout)

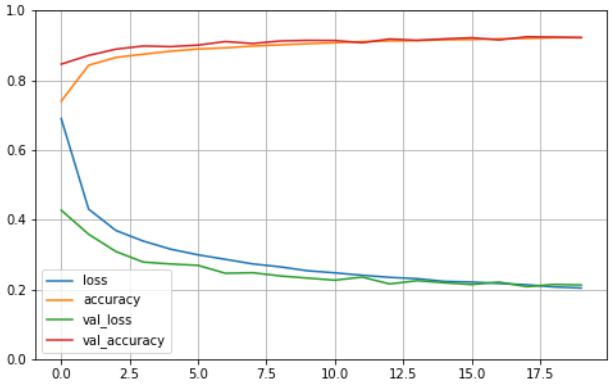

Figure 14: Learning curves from model 4b (CNN with L2 regularization + dropout)