

MSDS 458

Assignment #2

Setu Madhavi Namburu

1. Abstract: Zalando is Europe's leading online fashion platform for women, men and children. In order to promote their online sales by launching recommender systems, the management hired a data scientist to build deep learning models using their catalog of product images dataset, Fashion-MNIST. The requirement is, whenever a user searches for a product from the catalog, the recommender engine should recommend similar products with high accuracy. The data scientist is given a month to test and validate different deep learning models using Keras library in python and come up with final recommendations to the management.

2. Introduction: Zalanda is Europe's largest online fashion platform. Fashion-MNIST is a dataset of product images taken from their online catalog. The purpose of this research is to build a legible deep learning model to classify product images into corresponding product category to be used in their recommendation engine. The data scientist is tasked with building various DNN/CNN models using Keras Library in Python and come up with best model/set of models meeting the requirements: 1. High test accuracy, 2. Reasonable computational performance with stability, 3. Feature Interpretability & 4. Meeting time to market (1month). The report summarizes the intuition behind DNN/CNN models by synthesizing the experimental results based on this dataset & set of recommendations along with executive summary to the management.

2.1 Data Description: The Fashion-MNIST also consists of 70,000 grayscale images of 28x28 pixels (pixels of the images representing grey-white scale, values range from 0-254) divided into 60,000 training images and 10,000 test images. Each image is labeled with an integer from 0 to 9 representing 10 classes: 'T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat', 'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot', respectively. Figure 1 shows counts of train and test images in each of the classes, representing a perfectly balanced dataset. Figure 2 gives a glimpse of some of the images with their class labels. As seen in the example images, there seem to be three major categories of products: bags, footwear, clothing. There can be several strategies to classify these products: build 3 models to classify products

into these three major categories and build other models to classify sub-products within each category, build one single classifier in one shot. Until we conduct several experiments we wouldn't know the answer upfront. If a single model can be built to classify all 10 products with high accuracy that will be the preferable model.

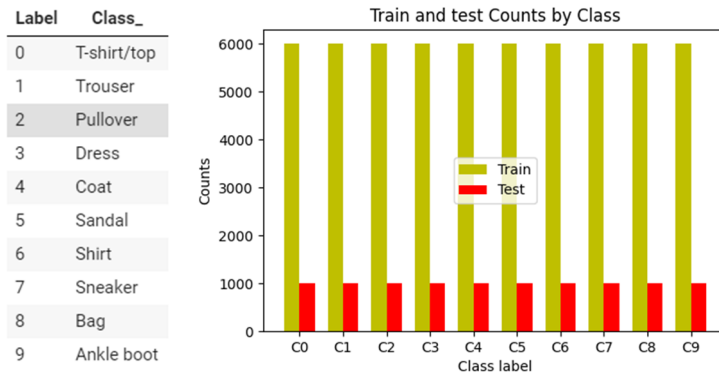


Figure 1. Class descriptions and their train/test counts

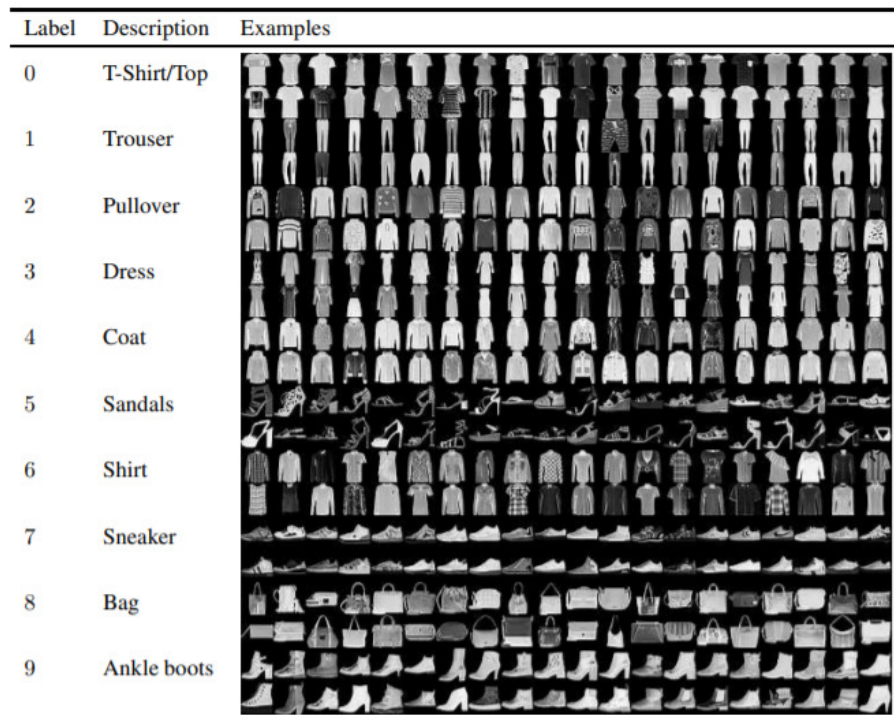


Figure 2. Example images from online catalog

2.2 Exploratory Data Analysis (EDA): To visualize the pixel data, the 28*28 numpy arrays from each image are reshaped into 60000*784 train, 10000*784 test 2-D arrays & the values are divided by 255 (as the scale ranges from 0-254) to normalize the data. T-SNE is a great visualization tool to observe separation between classes by maximizing the distance. Figure 3 shows scatter plot of couple of random pixels and 2D scatter plot of T-SNE.

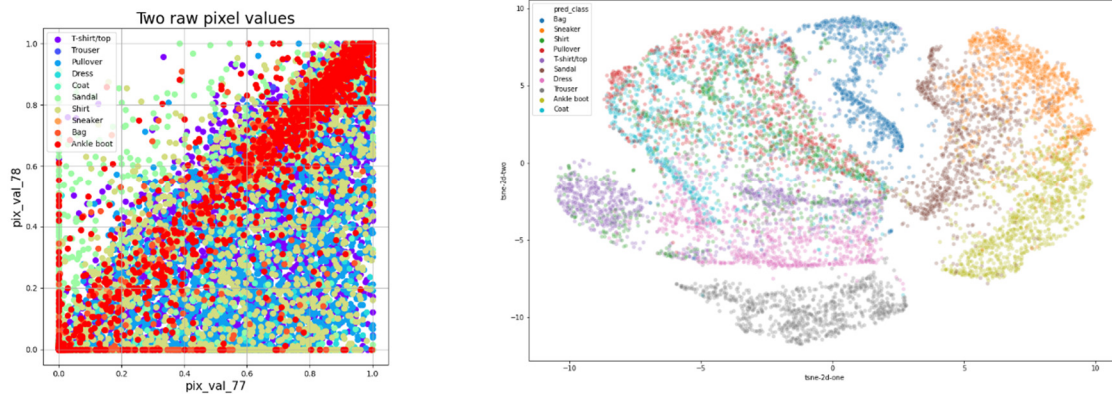


Figure 3. Visualization of image pixels using 2D scatter and T-SNE

Just two raw pixels do not seem to show any segregation, but the pattern observed using T-SNE is interesting. Trousers seem to have its own grey cluster, bags are in blue cluster while foot wear and remaining clothing shows two different major clusters (with somewhat overlapping clusters of categories within). This seem to be encouraging problem to build single classification model.

3. Literature Review: While Deep Neural Networks (DNN) are a major breakthrough in AI research, Convolutional Neural Networks (CNNs) have been proved to be second to none in tackling computer vision problems due to their powerful feature visualization feature (overcoming the major drawback of black box models). There have been several CNN architectures developed by major researchers in the field, for example, VGG, ResNet, Inception, Inception-ResNet, Xception are some of the names we may encounter [1]. Deshpande [2] presented an excellent summary of evolution of CNN

breakthroughs from 2012 -2015. Below table summarizes the ‘whys’ and ‘whats’ of these networks in a succinct manner.

Network Name	Year	Key Idea/Innovation	ILSRVC error rate	Who (Developers)	Technical details
AlexNet	2012	Large, deep CNN	15.4%	Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton	<ul style="list-style-type: none"> - 5 conv layers, max-pooling layers, dropout layers, and 3 fully connected layers - ReLU for the nonlinearity functions - data augmentation techniques that consisted of image translations, horizontal reflections, and patch extractions - batch stochastic gradient descent, with specific values for momentum and weight decay - Trained on two GTX 580 GPUs for five to six days
ZF Net	2013	Visualizing and Understanding Convolutional Neural Networks - at every layer of the trained CNN, attach a “deconvnet” which has a path back to the image pixels, examine what type of structures excite a given feature map	11.2%	Matthew Zeiler and Rob Fergus from NYU	<ul style="list-style-type: none"> - Very similar architecture to AlexNet - Instead of using 11x11 sized filters in the first layer (which is what AlexNet implemented), ZF Net used filters of size 7x7 and a decreased stride value - Used ReLUs for their activation functions, cross-entropy loss for the error function, and trained using batch stochastic gradient descent. - Trained on a GTX 580 GPU for twelve days. - Developed a visualization technique named Deconvolutional Network
VGG Net	2014	Simplicity and depth shrinking spatial dimensions but growing depth by creating effective receptive field with more number of smaller filters	7.3%	Karen Simonyan and Andrew Zisserman of the University of Oxford	<ul style="list-style-type: none"> - 19 layer CNN that strictly used 3x3 filters with stride and pad of 1, along with 2x2 maxpooling layers with stride 2 - Used a form of localization as regression - Used scale jittering as one data augmentation technique during training. - Used ReLU layers after each conv layer and trained with batch gradient descent. - Trained on 4 Nvidia Titan Black GPUs for two to three weeks.
GoogLeNet	2015	The idea of CNN layers didn’t always have to be stacked up sequentially. Introduced Inception module – performs pooling and conv operations in parallel, notable consideration on memory and power usage.	6.7%	Google	<ul style="list-style-type: none"> - 1x1 conv operations before the 3x3 and 5x5 layers – Idea of dimensionality reduction - Used 9 Inception modules in the whole architecture, with over 100 layers in total! - Uses 12x fewer parameters than AlexNet - During testing, multiple crops of the same image were created, fed into the network, and the softmax probabilities were averaged to give us the final solution. - Trained on “a few high-end GPUs within a week”
ResNet	2015	Introduction of Residual block - It is easier to optimize the residual mapping than to optimize the original, unreferenced mapping	3.6%	Microsoft Research Asia	<ul style="list-style-type: none"> - “Ultra-deep” 152 layer network - Trained on an 8 GPU machine for two to three weeks
Region based CNNs	2013/2014/2015	Problem of object detection using two process steps – region proposal step and classification step. Being able to determine that a specific object is in an image is one thing, but being able to determine that object’s exact location is a huge jump in knowledge for the computer.		Ross Girshick and his group at UC Berkeley	<ul style="list-style-type: none"> - Selective search method for region proposal - these proposals are then “warped” into an image size that can be fed into a trained CNN (AlexNet in this case) that extracts a feature vector for each region - Linear SVMs are used to train using CNN vector outputs to classify objects - CNN vectors are also fed into a bounding box regressor to obtain the most accurate coordinates - Non-maxima suppression is then used to suppress bounding boxes that have a significant overlap with each other. - ConvNets to SVMs to bounding box regressors – to simplify this complex R-CNN pipeline, region proposal network (RPN) was introduced after the last conv layer
GANs	2014	Making the discriminator aware of “internal representation of the data”. Adversarial examples are basically the images that fool ConvNets. The analogy used in the paper is that the generative model is like “a team of counterfeiters, trying to produce and use fake currency” while the discriminative model is like “the police, trying to detect the counterfeit currency”.		Yann LeCun.	<ul style="list-style-type: none"> - Consists of two models - a generative model and a discriminative model - can be thought of as a zero-sum or minimax two player game - Can be used as a feature extractor that can be used in CNN
Spatial Transformer Network	2015	Back to basics – Instead of making changes to the main CNN architecture itself, the authors worry about making changes to the image before it is fed into the specific conv layer. The basic idea is that this module transforms the input image in a way so that the subsequent layers have an easier time making a classification.		Google Deepmind	<ul style="list-style-type: none"> - Two contributions - pose normalization (scenarios where the object is tilted or scaled) and spatial attention (bringing attention to the correct object in a crowded image) using transformer framework in a dynamic way - Implements the simple idea of making affine transformations to the input image in order to help models become more invariant to translation, scale, and rotation <ul style="list-style-type: none"> - A localization network which takes in the input volume and outputs parameters of the spatial transformation that should be applied. The parameters, or theta, can be 6 dimensional for an affine transformation. - The creation of a sampling grid that is the result of warping the regular grid with the affine transformation (theta) created in the localization network. - A sampler whose purpose is to perform a warping of the input feature map.

Table 1. Evolution of CNN breakthroughs from 2012-2015

4. Methods: In this report, analysis and synthesis of experimental results from one 2-layer DNN model and one 2-layer CNN model are discussed in detail to understand the advantage of CNN over DNN for computer vision problems. Next, several experiments are conducted using various architectures of DNN and CNN and the performance metrics are evaluated in terms of classification performance and stability of the models. All experiments are conducted in Google Collaboratory with GPU (12.72 GB RAM & 68.4GB disk space). Table 2 & 3 present the experimental set up for 8 DNN models and 8 CNN models with various configurations respectively. Exp_key serves as the ID for tracking experiments and reporting the results. The data is restructured into no.of images x 784 2D array for DNN, no.of images x 28 x 28 x 1 4D tensors for CNN (last dimension represents the channel, 3 for RGB, 1 for

greyscale). In all the networks, ReLU activation functions are used in the inner layers and softmax in the output layer with SparseCategoricalCrossentropy as the loss function to minimize the error.

Exp_Key	No. of Layers	Hidden Units	Drop Out	Drop Out Percentage	Optimizer	Regularization
DNN_1L_No_Dropout_Exp1	1	[54]	No	N/A	"adam"	No
DNN_2L_No_Dropout_Exp1	2	[54, 128]	No	N/A	"adam"	No
DNN_2L_No_Dropout_Exp2	2	[128,238]	Yes	[0,20]	"adam"	No
DNN_2L_No_Dropout_Exp3	2	[54, 128]	No	N/A	"rmsprop"	No
DNN_2L_No_Dropout_Exp4	2	[128,238]	Yes	[0,20]	"rmsprop"	No
DNN_2L_No_Dropout_Exp5	2	[128,238]	Yes	[0,20]	"adam"	[N/A,L2]
DNN_3L_No_Dropout_Exp1	3	[54,128,54]	No	N/A	"adam"	No
DNN_3L_No_Dropout_Exp2	3	[54,128,238]	Yes	[0,0,20]	"adam"	No
DNN_3L_No_Dropout_Exp3	3	[54,128,238]	Yes	[0,0,20]	"rmsprop"	[N/A,L2,L2]

Table 2. List of DNN Experiments

Exp_Key	No. of CNN Layers	No. of Filters	No. of Dense Layers	No. of Hidden Units	Drop Out	Drop Out Percentage	Data Augmentation	Optimizer	Batch size	Conv Layer Kernel size	Conv Layer Strides	Maxpool Layer Kernel size	Maxpool Layer Strides
CNN_2L_No_Dropout_Exp1	2	[32,64]	1	[1024]	No	N/A	No	"adam"	512	(3,3)	1	(2,2)	2
CNN_2L_No_Dropout_Exp2	2	[32,64]	1	[1024]	Yes	[20]	No	"adam"	512	(3,3)	1	(2,2)	2
CNN_2L_No_Dropout_Exp3	2	[32,64]	2	[1024,54]	No	N/A	No	"adam"	512	(3,3)	1	(2,2)	2
CNN_3L_No_Dropout_Exp4	3	[32,64,128]	1	[1024]	No	N/A	No	"adam"	512	(3,3)	1	(2,2)	2
CNN_3L_No_Dropout_Exp5	3	[32,64,128]	1	[1024]	Yes	[50]	No	"adam"	512	(3,3)	1	(2,2)	2
CNN_1L_No_Dropout_Exp6	1	[64]	1	[1024]	Yes	[50]	No	"adam"	512	(3,3)	1	(2,2)	2
CNN_1L_No_Dropout_Exp7	1	[64]	1	[128]	Yes	[10]	Yes	"adam"	512	(3,3)	1	(2,2)	2
CNN_2L_No_Dropout_Exp8	2	[32,64]	1	[1024]	Yes	[20]	Yes	"adam"	512	(3,3)	1	(2,2)	2

Table 3. List of CNN Experiments

5. Results: In the following sections, results of 2 Layer DNN model and 2 Layer CNN model with one dense layer (shown in figure 4) and other experiments are discussed through visualizations.

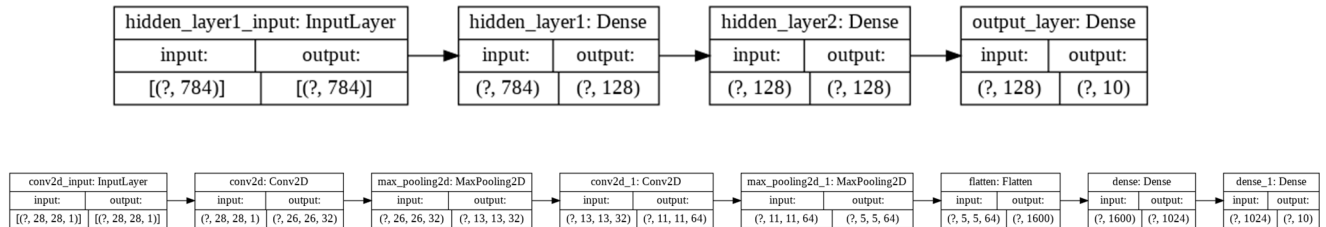


Figure 4. 2-Layer DNN, 2-Layer CNN

5.1 DNN and CNN Models Comparison: The 2 layer DNN fully connected with 128 hidden units in each layer, CNN with 2 Conv layers, 2 maxpool layers followed by one fully connected dense layer are trained with 20 epochs using “adam” optimizer. Figure 5 show their loss and accuracy performance metrics. Both models are not that stable, but the intermittent results are studied further to understand how each of these networks works with the data.

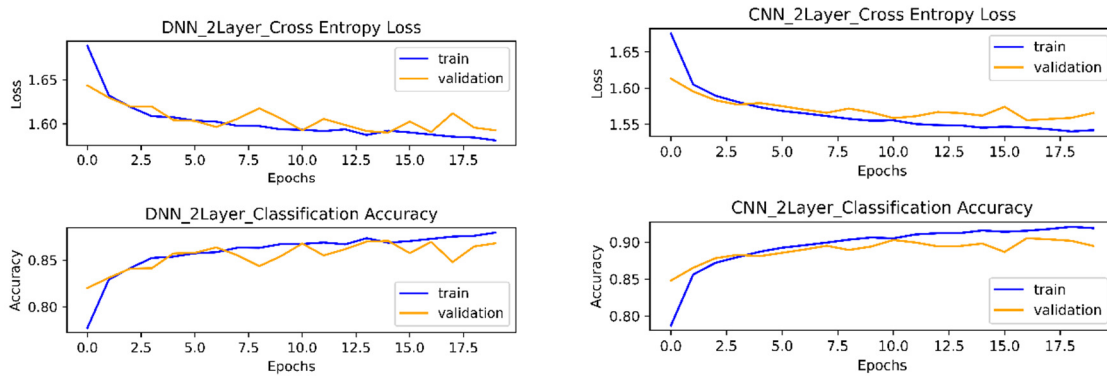


Figure 5. Performance metrics of DNN and CNN networks

CNN got 89% overall test accuracy while DNN gave test accuracy of 85.7%. Figure 6 shows the comparison of confusion matrix style classifications for sneakers vs bags with both the networks. Clearly, DNN does not have spatial representation power whereas CNN is able to represent the inner features, for example bags vs sneakers using convolution operations resulting in better performance.

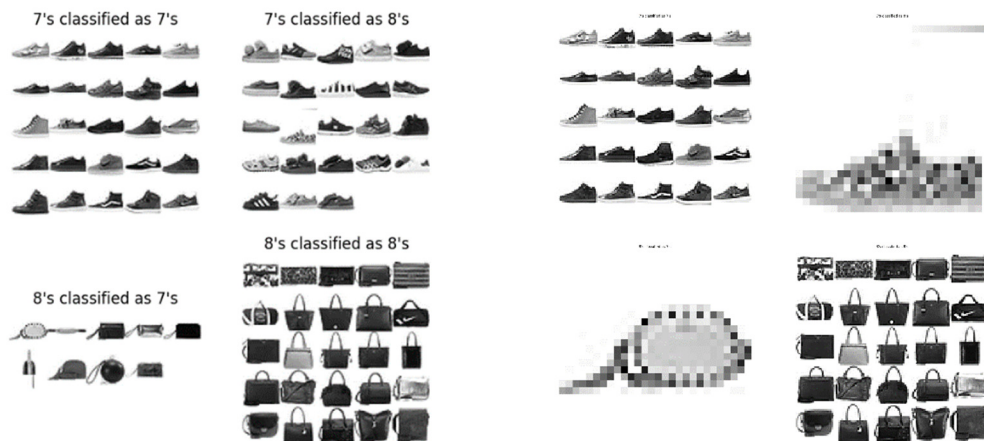


Figure 6. Classification of sneakers vs bags with DNN & CNN

Figure 7 shows the scatter plot of two activations from DNN. We can start seeing the difficulty of not being able to get further into feature representations beyond activation values from DNN.

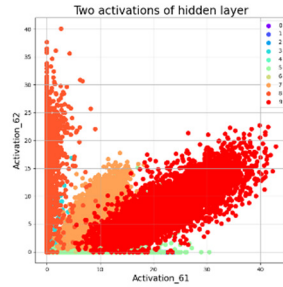



Figure 7. Scatter plot of DNN activations from 2nd layer

That is where CNN visualization feature comes very handy in understanding the low level and high-level features captured by each of the layers in CNN network. Figure 8 shows inner representations of bag's image  in each of the layers of our network. As can be seen the first layer captures the high-level features (almost preserving the shape) and as the network gets deeper it captures more abstract features from first layer. The filter operations on conv and maxpool layers act as feature extractors by capturing complex features from the images.

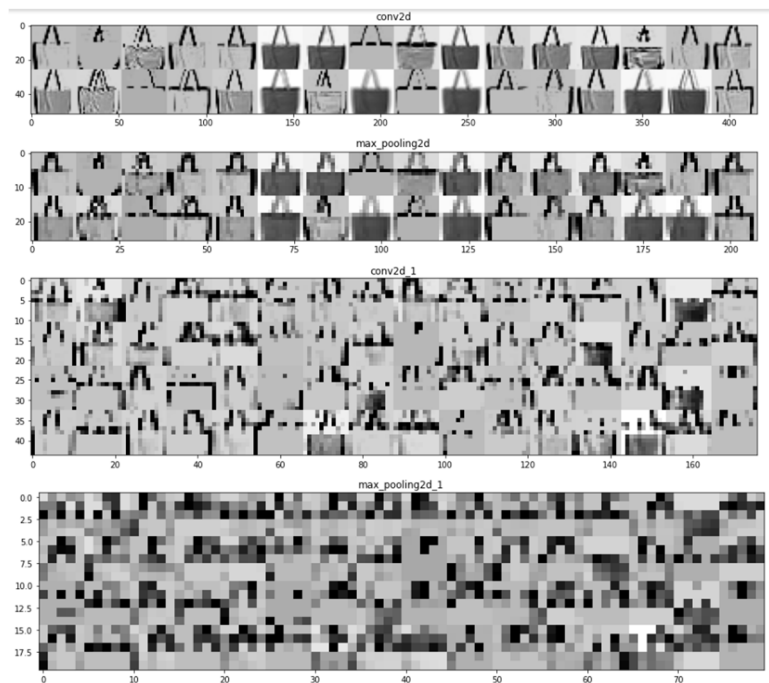


Figure 8. Representation of bag's image features in convolution and maxpool layers

5.2 Experimental Results & Discussion: As mentioned in section 4, experiments are conducted using 8

DNN and 8 CNN model architectures with different settings (optimizers, drop outs, number of layers, number of hidden units/filters, regularization, data augmentation). Table 4 and Table 5 presents the results from DNN and CNN experiments respectively.

exp_key	test_accuracy	train_accuracy	train_loss	train_time	validation_accuracy	validation_loss
DNN_1L_No_Dropout_Exp1	0.83850	0.87190	1.59185	0 days 00:01:09.289808000	0.86063	1.60177
DNN_2L_No_Dropout_Exp1	0.80270	0.80342	1.65788	0 days 00:01:14.962169000	0.79860	1.66212
DNN_2L_Dropout_Exp2	0.85280	0.85107	1.60990	0 days 00:01:16.797644000	0.85274	1.60798
DNN_2L_No_Dropout_Exp3	0.85630	0.85516	1.60613	0 days 00:01:22.414350000	0.84983	1.61116
DNN_2L_Dropout_Exp4	0.84340	0.84358	1.61738	0 days 00:01:28.394744000	0.84110	1.61977
DNN_2L_Dropout_Exp5	0.86720	0.86740	1.60781	0 days 00:01:20.288768000	0.85918	1.61384
DNN_3L_No_Dropout_Exp1	0.85350	0.85162	1.60941	0 days 00:01:19.781059000	0.84869	1.61203
DNN_3L_Dropout_Exp2	0.84040	0.83616	1.62481	0 days 00:01:21.127042000	0.83272	1.62804
DNN_3L_Dropout_Exp3	0.85690	0.86204	1.61459	0 days 00:01:39.820548000	0.85301	1.62085

Table 3. DNN Models Experimental Results

exp_key	test_accuracy	train_accuracy	train_loss	train_time	validation_accuracy	validation_loss
CNN_2L_No_Dropout_Exp1	0.8977	0.8799	1.5839	0 days 00:00:22.984013000	0.8783	1.5841
CNN_2L_Dropout_Exp2	0.9022	0.8744	1.5890	0 days 00:00:21.182181000	0.8806	1.5818
CNN_2L_No_Dropout_Exp3	0.9094	0.8633	1.5993	0 days 00:00:40.091262000	0.8624	1.5992
CNN_3L_No_Dropout_Exp4	0.8687	0.8333	1.6302	0 days 00:00:21.882057000	0.8349	1.6268
CNN_3L_Dropout_Exp5	0.8545	0.7970	1.6652	0 days 00:00:21.838053000	0.8178	1.6433
CNN_1L_Dropout_Exp6	0.9137	0.9012	1.5621	0 days 00:00:39.713825000	0.8981	1.5644
CNN_1L_Dropout_DA_Exp7	0.8981	0.8815	1.5847	0 days 00:03:07.702240000	0.8814	1.5835
CNN_2L_Dropout_DA_Exp8	0.8946	0.8577	1.6057	0 days 00:03:10.076071000	0.8670	1.5952

Table 4. CNN Models Experimental Results

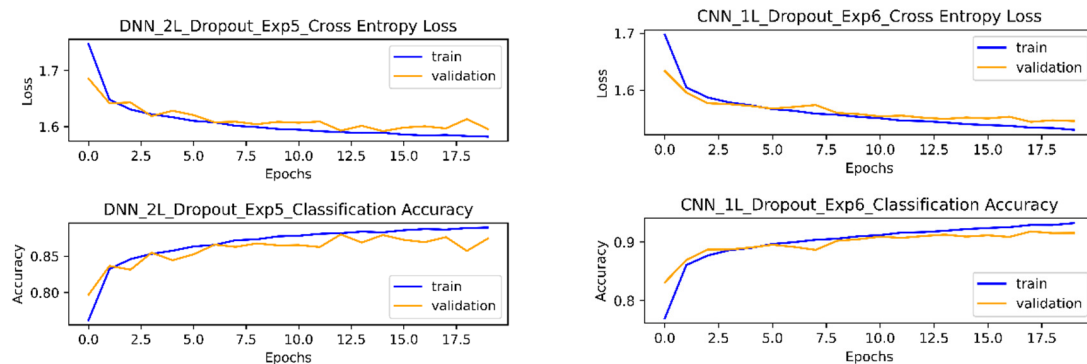


Figure 9. Stability performance metrics of best DNN and CNN models

From the tables, CNN models are the outperformers in terms of test accuracy. CNN is also better in terms of stability as shown in figure 9 (based on best models from each). Top 3 models in terms of test accuracy and training time are highlighted in green, yellow, red as shown in Table 4. Before qualifying CNN_1L_Dropout_Exp6 as the best model just based on test accuracy, its performance is compared against another better stable model, CNN_2L_Dropout_Exp2 in Figure 10.

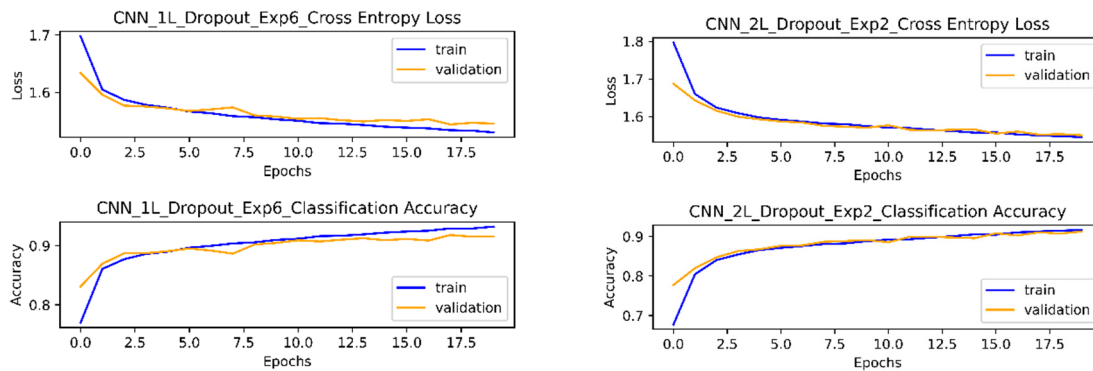


Figure 10. Stability comparison of two best CNN models

While CNN_2L_Dropout_Exp2 is clearly the winner in terms of stability, the classification accuracies of individual classes are compared between both the models. As we observed from EDA that Shirt may be hard to classify, the hypothesis is verified from individual classifiers. While both models performed relatively poorly for class 6/shirt, CNN_1L_Dropout_Exp6 performed equal or better than the other model in the remaining classes.

Class Name	CNN_1L_Dropout_Exp6	CNN_2L_Dropout_Exp2
T-shirt/top	0.94	0.89
Trouser	0.99	0.99
Pullover	0.9	0.81
Dress	0.95	0.95
Coat	0.93	0.9
Sandal	0.99	0.98
Shirt	0.81	0.78
Sneaker	0.99	0.97
Bag	0.99	0.99
Ankle boot	0.97	0.97

Table 5. Classification accuracies of individual classes

So among the experiments conducted, the simple model CNN with one conv/max pool layer with 64 filters and 1 dense layer (with 1024 hidden nodes) with 50% drop out with adam optimizer and batch processing is chosen as the best model. Figure 11 shows T-SNE visualization of test image activations in 2D. Compared to the raw data T-SNE, the rest of the classes got much better separation except class6/shirt. It would be interesting to conduct further experiments to understand the extracted features to improve class 6 rate. It may be required to develop another binary model (6 vs not 6) to improve its performance and ensemble the outputs with overall classifier.

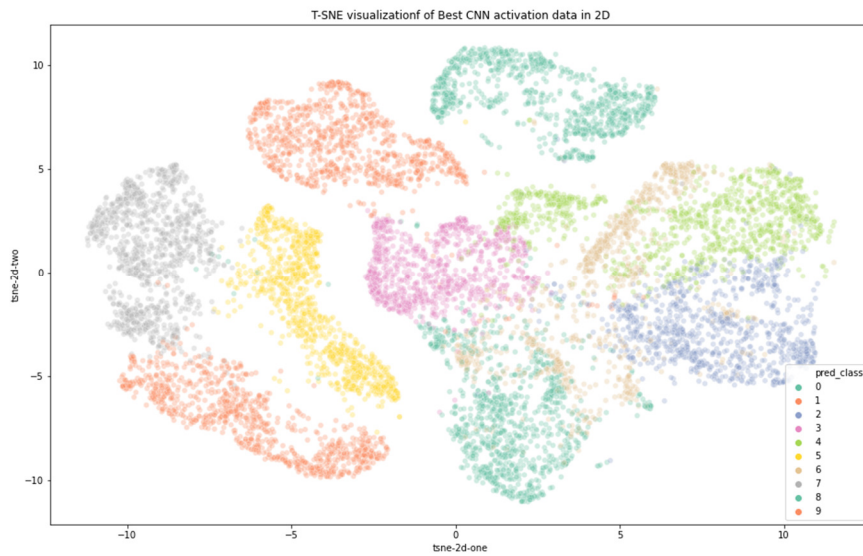


Figure 11. T-SNE visualization of best CNN activations

6. Conclusion: Zalanda, Europe's largest on-line fashion retailer is planning to launch online recommender system with deep learning models that can automatically classify catalog products. With the requirements of 1. High test accuracy, 2. Reasonable computational performance with stability, 3. Features understanding and 4. Time to design (1 month), data scientist conducted several

experiments using various DNN and CNN architectures. Among the models evaluated, a simple CNN architecture with 1 64 filter conv/maxpool layer and 1 dense layer with 1024 hidden units and 50% drop out is recommended as the final model for deployment. While the stability of this model is not the best among other competing models, the classification accuracy of hard to classify products such as 'Shirt' is given higher weightage for final selection. Further strategies may be explored in future to improve the accuracy of this and other products (e.g., building ensembles, binary classifiers, pre-trained models).

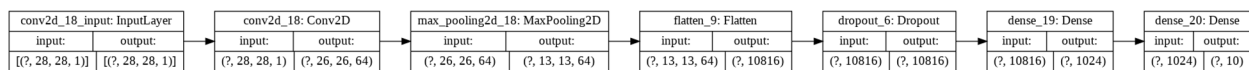


Figure 12. Selected best CNN Model

Lessons Learned:

- As we move from DNNs to CNNs, the experimental space seem to become vast (100s of network architectures and tunable parameters)
- There is no single glide path, experimenting with data from given use case and making intuitive judgements seem to be the way to go
- While ultimate best model may be possible, it is ok to move sub-optimal models to production if the technical caveats are understood and well documented (under the given time and resource constraints).

7. References:

- [1] Chollet, F. 2018. Deep Learning with Python. Shelter Island, N.Y.: Manning. [ISBN-13: 978-1617294433]
- [2] Deshpande, A. (2016, August 24). The 9 Deep Learning Papers You Need to Know About (Understanding CNNs Part 3). Blogpost on [adeshpande3.github](https://github.com/adeshpande3). Online Access