Fashionable Learning

- A deep study on how networks learn

Problem Formulation

Understand how neural networks learn

Implement several networks to get a better insight into how they work and identify features in the classification of images

Visualize the training

Use visualization tools to see how the networks change during training

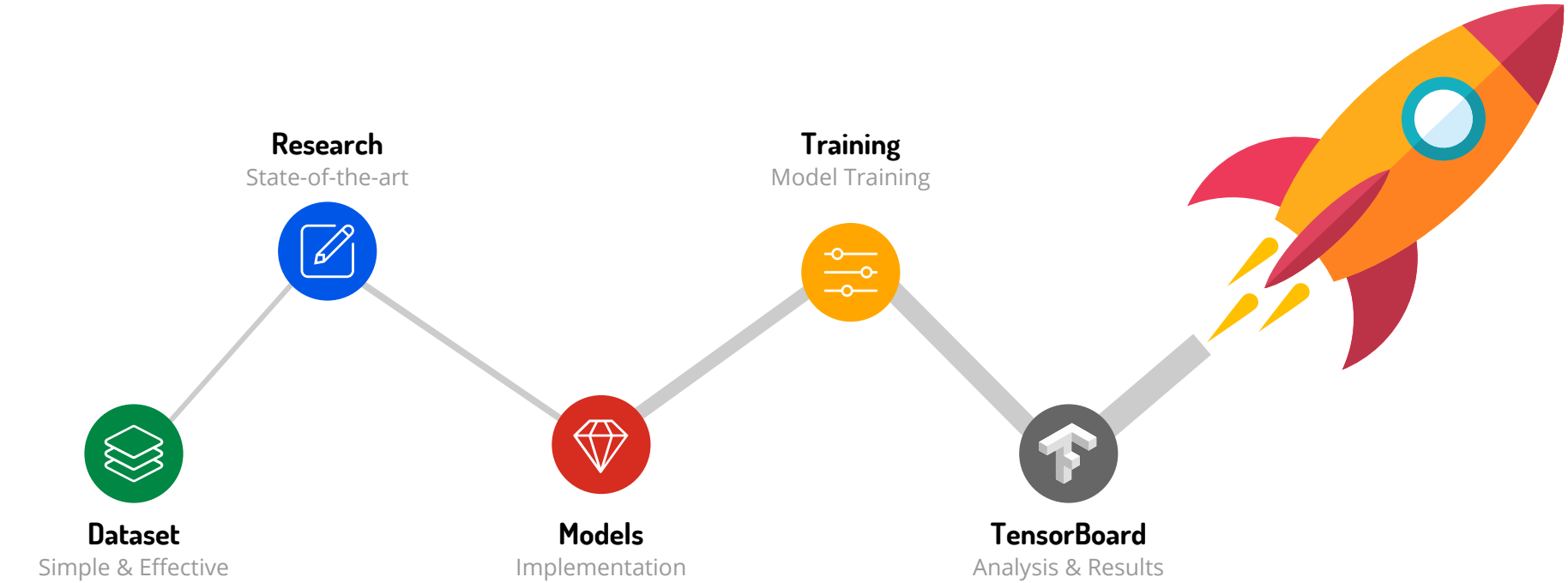
Find cutting-edge NNs

Compare our results with state-of-the art neural networks

Dataset

Zalandoresearch's Fashion-MNIST - a more complicated MNIST-like dataset used for experimentation in deep learning

Technical Approach



Dataset

Fashion-MNIST

TRAINING

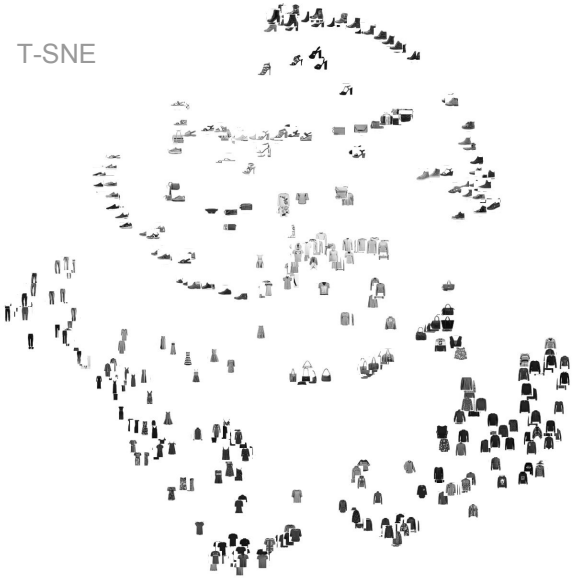
60 000

VALIDATION

5000

TEST

10 000

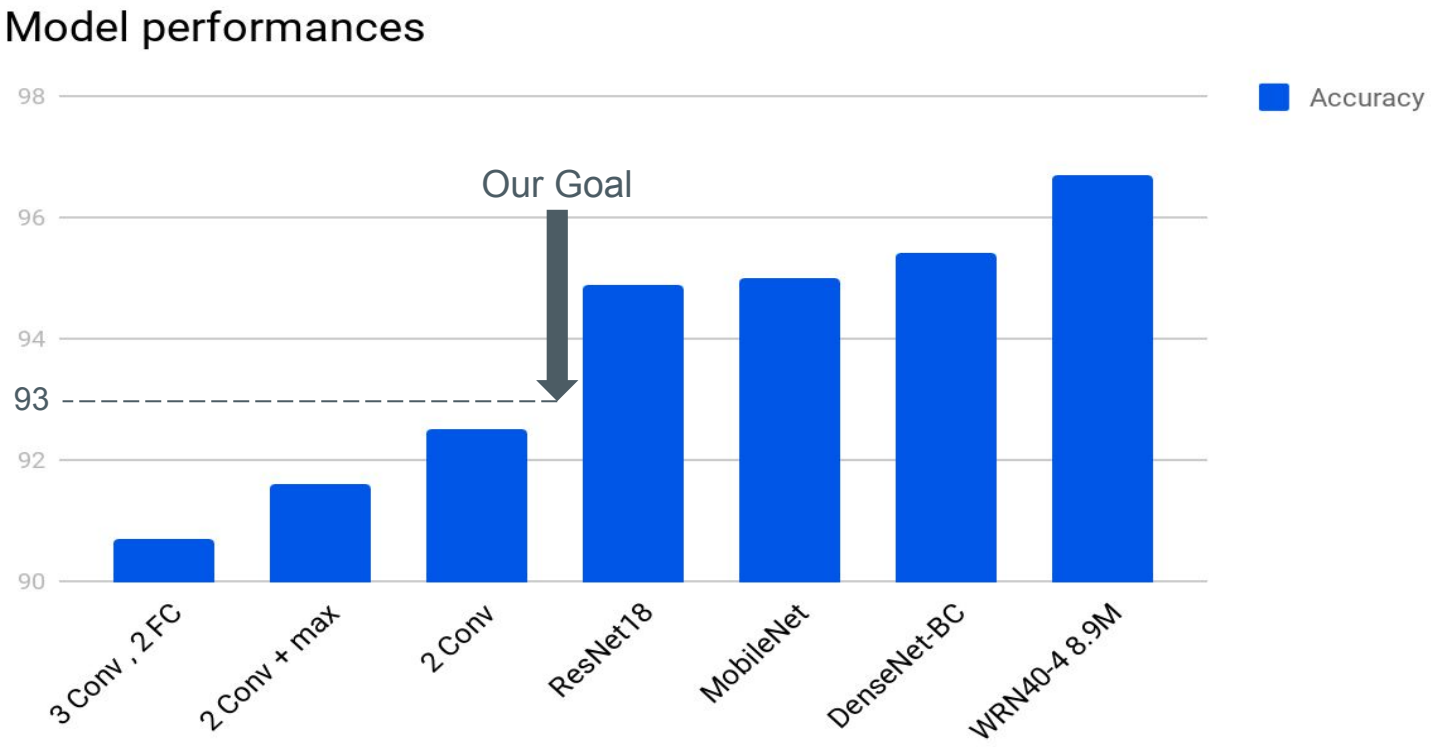


Why did we choose this dataset?

Simple and effective for our purpose

Research

A look into existing models, and a benchmarking of them



Models

An overview of the models we chose to implement

Logistic Regression

Artificial Neural Network (ANN)

Dense Layers

1 2 3

Convolutional Neural Network (CNN)

Convolutional Layers

1 2 3

VGG16

Expected Increasing Accuracy

Training

Tweaking and Tuning

EPOCHS

50

ITERATIONS

600

BATCH SIZE

100

Learning Rate

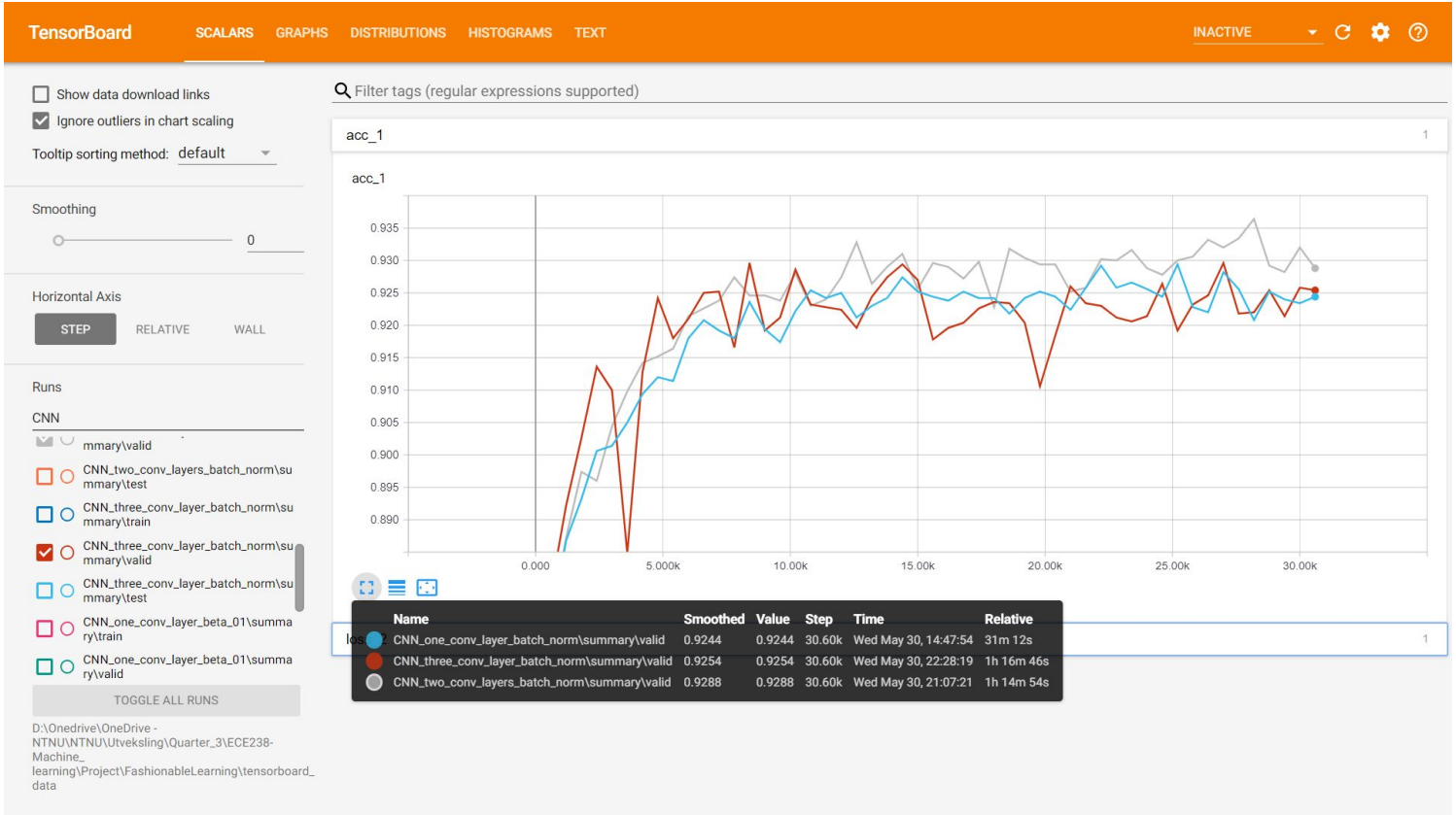
Regularization

Batch Normalization

Dropout Probability

TensorBoard

TensorFlow's visualization tool



ANALYSIS & RESULTS

What did we find?

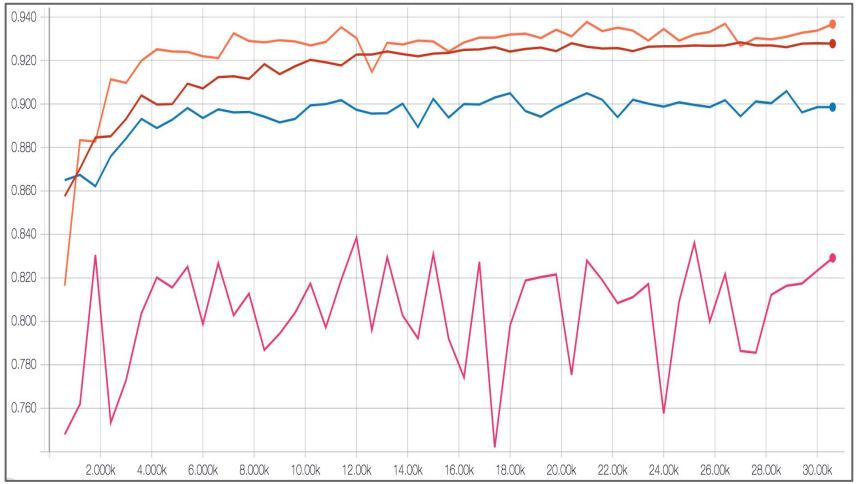
TODO: REMOVE

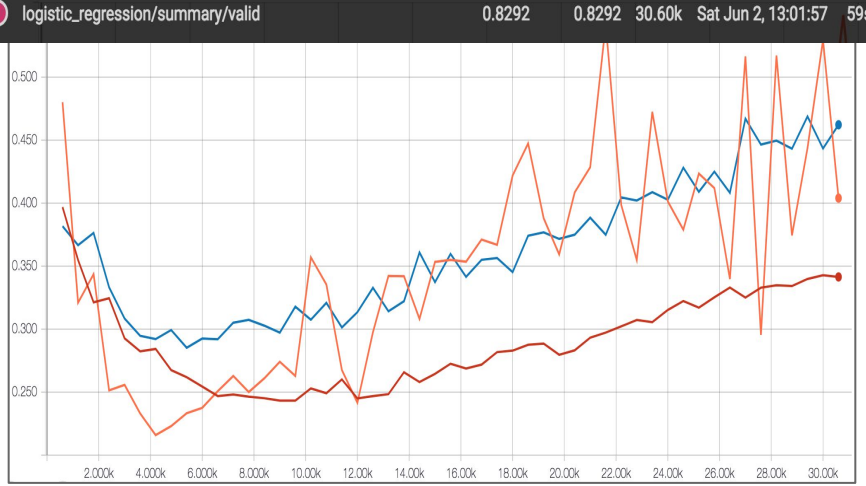
Analysis & Results Outline

- Comparison
  - All networks (perhaps even with random forest as a representative for traditional machine learning?) (Acc + Loss)
  - The 3 ANNs (Acc + Loss)
  - The 3 CNNs
    - Acc + Loss
    - Histograms
    - Distributions
  - Accuracy vs computation time. Maybe compare the best ANN with the best CNN
- Deep-dive into the best-performing CNN (one layer)
  - Filter evolution (gif)
  - (Should we say something about how networks learn in general?)
  - How do the weights and biases (etc.) change over time?
  - Do all layers learn?
- Overfitting

Comparison of all Ne

Name	Smoothed	Value	Step	Time	Relative
CNN_one_conv_layer_dropout_07_lr_0001_09999/summary/valid	0.9278	0.9278	30.60k	Sat Jun 2, 22:06:14	4m 31s
VGG16/summary/valid	0.9368	0.9368	30.60k	Sun Jun 3, 04:19:18	11h 8m 48s
dense_one_layer/summary/valid	0.8986	0.8986	30.60k	Sat Jun 2, 11:51:29	4m 42s
logistic_regression/summary/valid	0.8292	0.8292	30.60k	Sat Jun 2, 13:01:57	59s





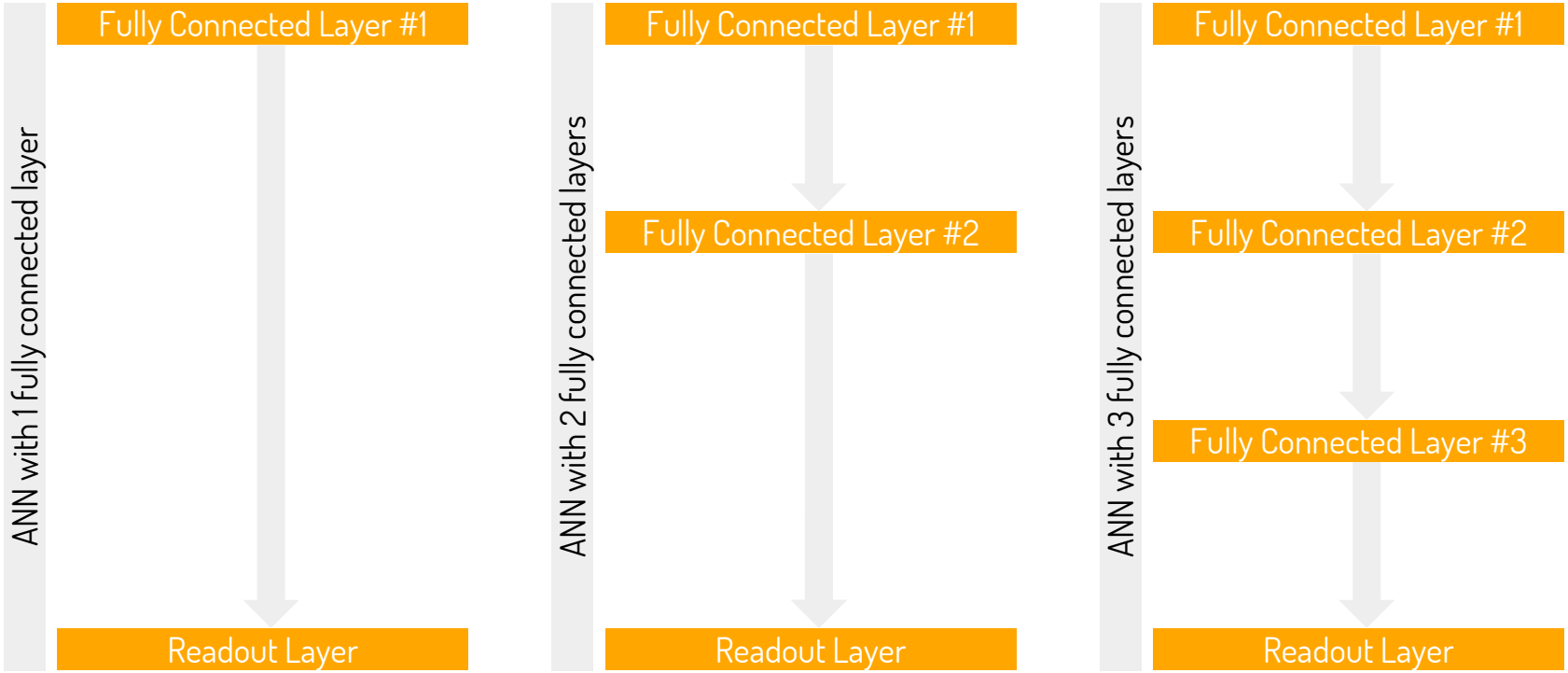
Accuracy
VGG16 has the highest accuracy, as expected

Loss
CNN with one layer has the lowest loss, which is also more stable across epochs than the other networks.

Test Accuracy			
LR	ANN	CNN	VGG16
0.8092	0.8960	0.9243	0.9265

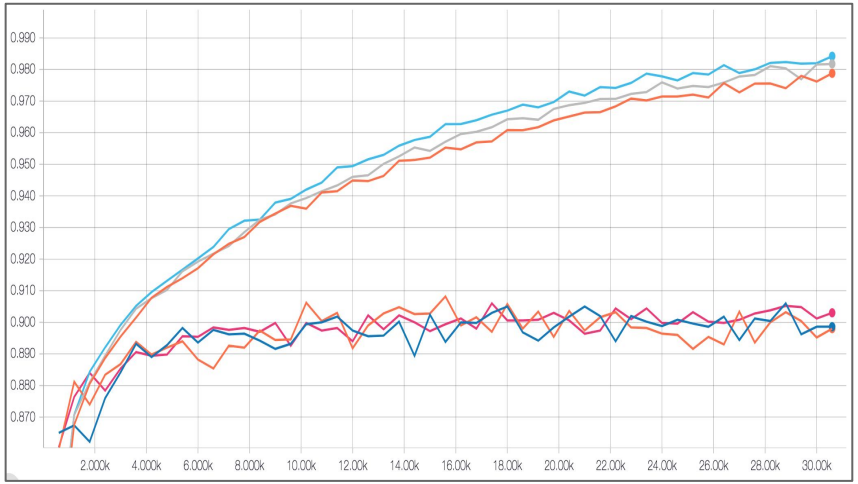
Test Loss			
LR	ANN	CNN	VGG16
11.546	0.4950	0.3481	0.4840

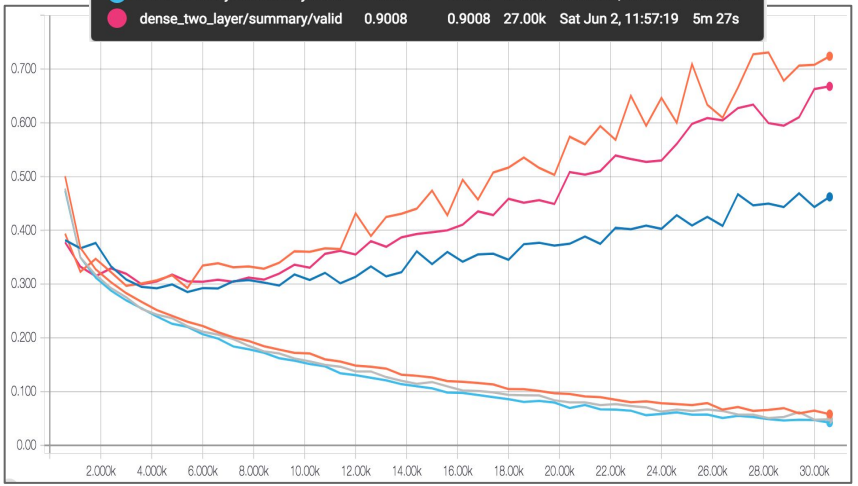
ANN comparisons (1/3)



ANN comparisons (2/3)

	Name	Smoothed	Value	Step	Time	Relative
	dense_one_layer/summary/train	0.9728	0.9728	27.00k	Sat Jun 2, 11:50:55	4m 7s
	dense_one_layer/summary/valid	0.8944	0.8944	27.00k	Sat Jun 2, 11:50:55	4m 7s
	dense_three_layer/summary/train	0.9778	0.9778	27.00k	Sat Jun 2, 12:11:41	6m 47s
	dense_three_layer/summary/valid	0.9034	0.9034	27.00k	Sat Jun 2, 12:11:41	6m 47s
	dense_two_layer/summary/train	0.9789	0.9789	27.00k	Sat Jun 2, 11:57:19	5m 27s
	dense_two_layer/summary/valid	0.9008	0.9008	27.00k	Sat Jun 2, 11:57:19	5m 27s





Accuracy

Negligible differences between each ANN

Loss

The ANNs with more layers have a higher loss

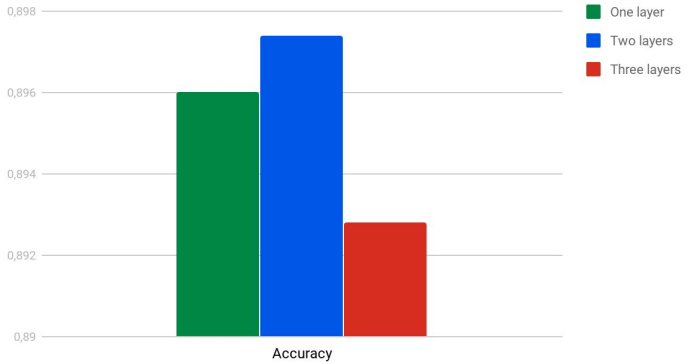
Overfitting occurs after a few epochs

Test Accuracy		
1	2	3
0.8960	0.8974	0.8928

Test Loss		
1	2	3
0.4950	0.7346	0.7644

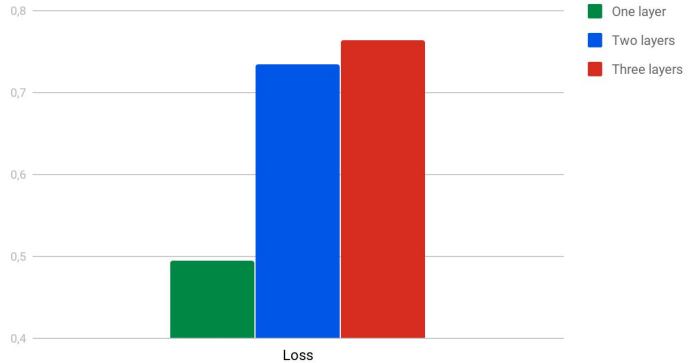
ANN comparisons (3/3)

Test accuracy

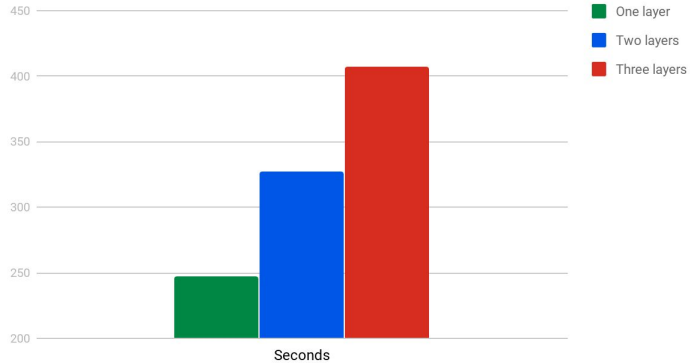


The one layer ANN is the neural network of choice (similar accuracy, and lower loss and comp. time than the others)

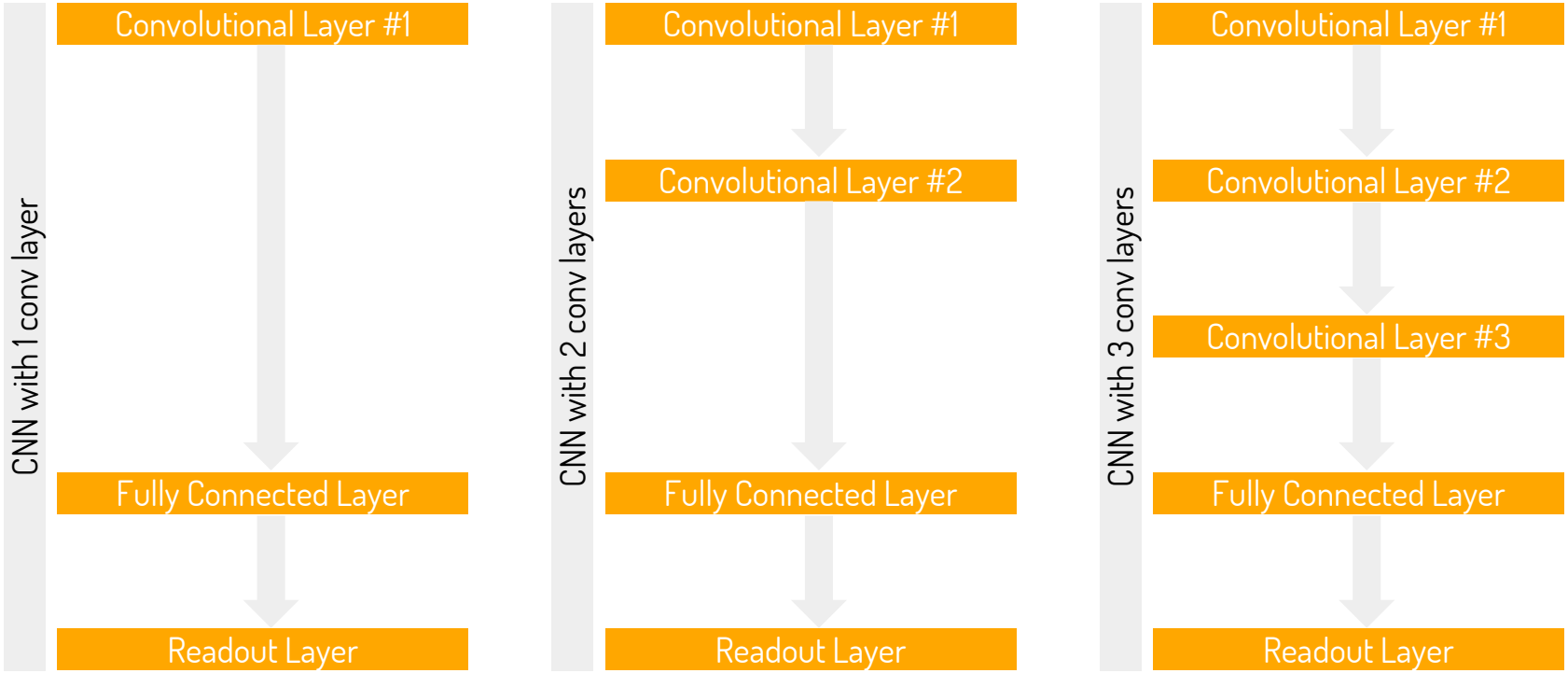
Test loss



Computation time

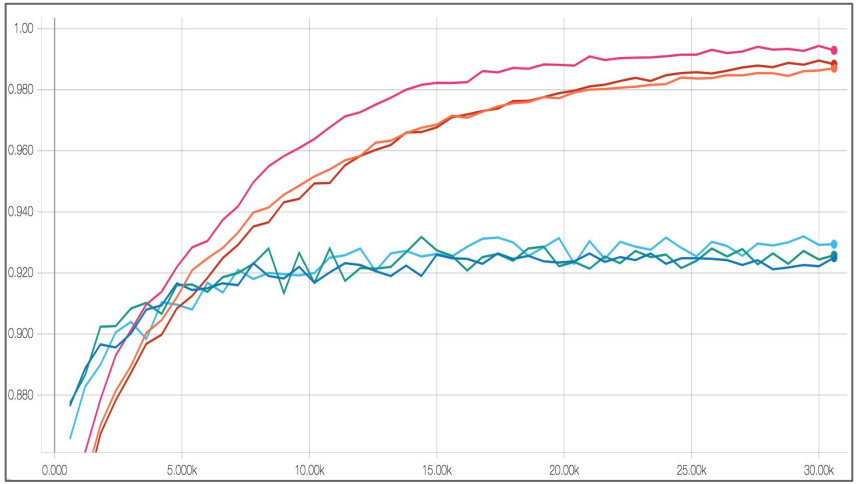


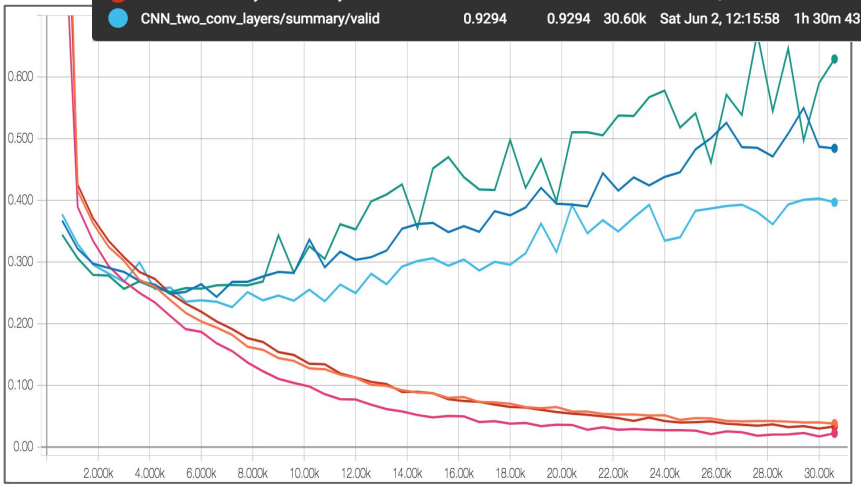
CNN comparisons (1/4)



CNN comparisons (2/4)

Name	Smoothed	Value	Step	Time	Relative
CNN_one_conv_layer_dropout_05/summary/train	0.9870	0.9870	30.60k	Sat Jun 2, 11:43:26	1h 0m 17s
CNN_one_conv_layer_dropout_05/summary/valid	0.9250	0.9250	30.60k	Sat Jun 2, 11:43:26	1h 0m 17s
CNN_three_conv_layers/summary/train	0.9929	0.9929	30.60k	Sat Jun 2, 12:43:52	1h 54m 14s
CNN_three_conv_layers/summary/valid	0.9258	0.9258	30.60k	Sat Jun 2, 12:43:52	1h 54m 14s
CNN_two_conv_layers/summary/train	0.9883	0.9883	30.60k	Sat Jun 2, 12:15:58	1h 30m 43s
CNN_two_conv_layers/summary/valid	0.9294	0.9294	30.60k	Sat Jun 2, 12:15:58	1h 30m 43s





Accuracy

Negligible differences between each CNN

Loss

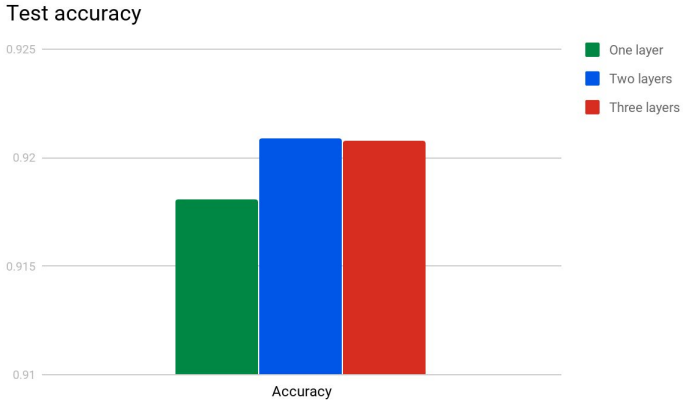
The CNN with two convolution layers has the smallest loss for validation data

Overfitting occurs after a few epochs

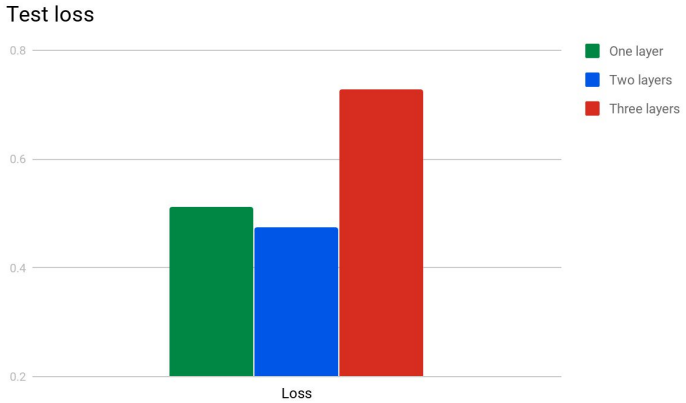
Test Accuracy		
1	2	3
0.9181	0.9209	0.9208

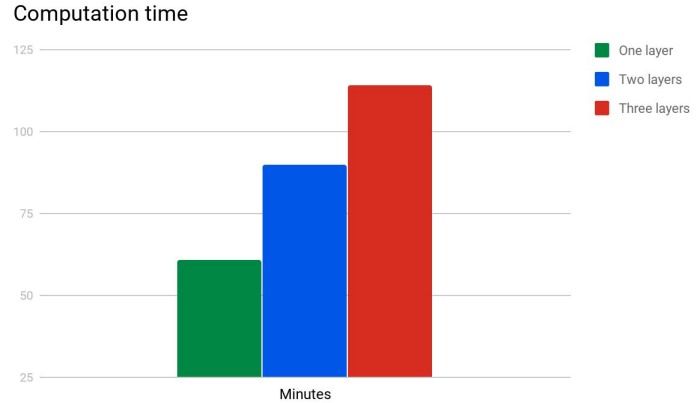
Test Loss		
1	2	3
0.5122	0.4740	0.7275

CNN comparisons (3/4)



Even though the two-convolutional-layer CNN has higher accuracy and lower loss than the one-convolutional-layer, the difference is negligible compared to the difference in computation time. Hence, the one-convolutional-layer CNN is the CNN of choice.

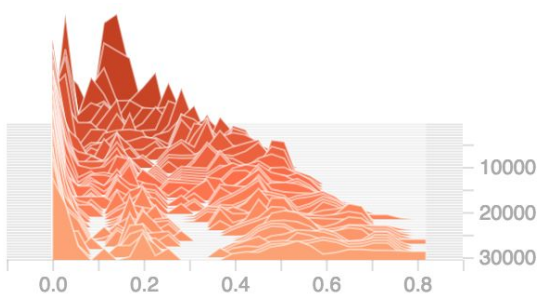


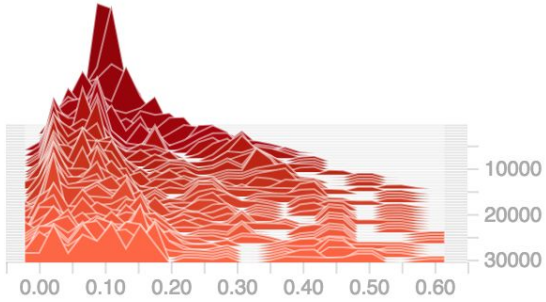


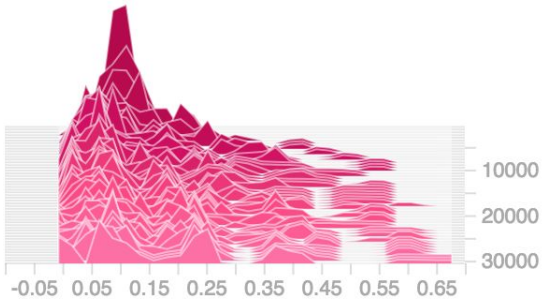
CNN comparisons (4/4) - Histograms

Convolutional Layer #1		
CNN with 1 conv layer	CNN with 2 conv layers	CNN with 3 conv layers

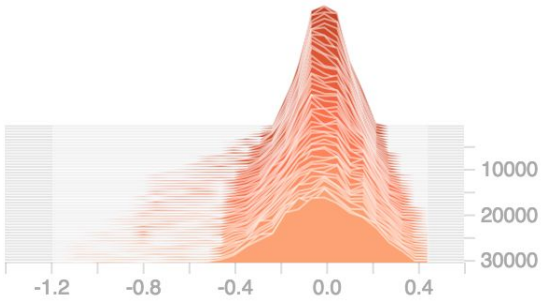
Bias

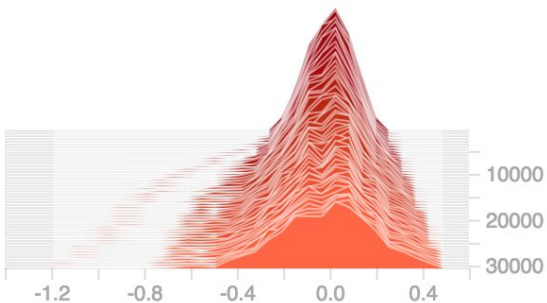


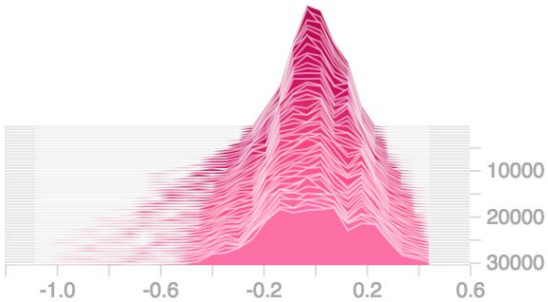




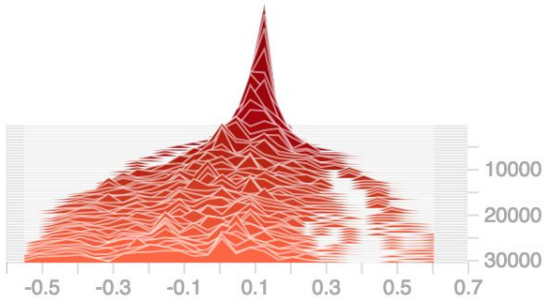
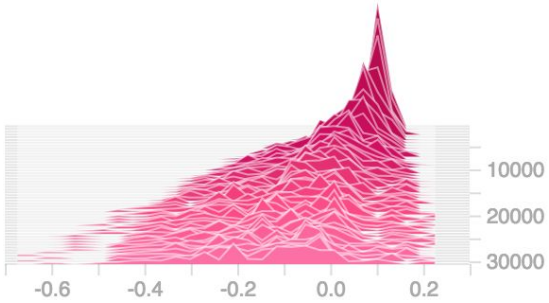
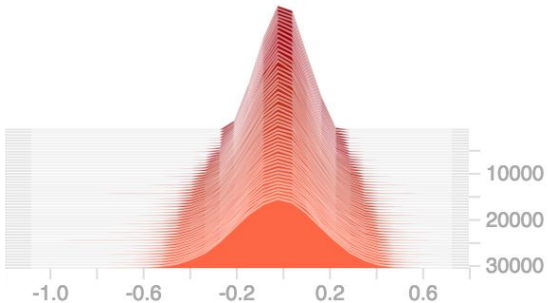
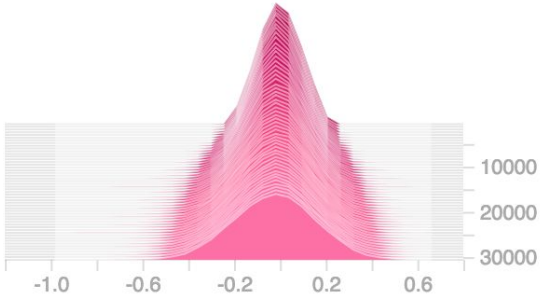
Weight







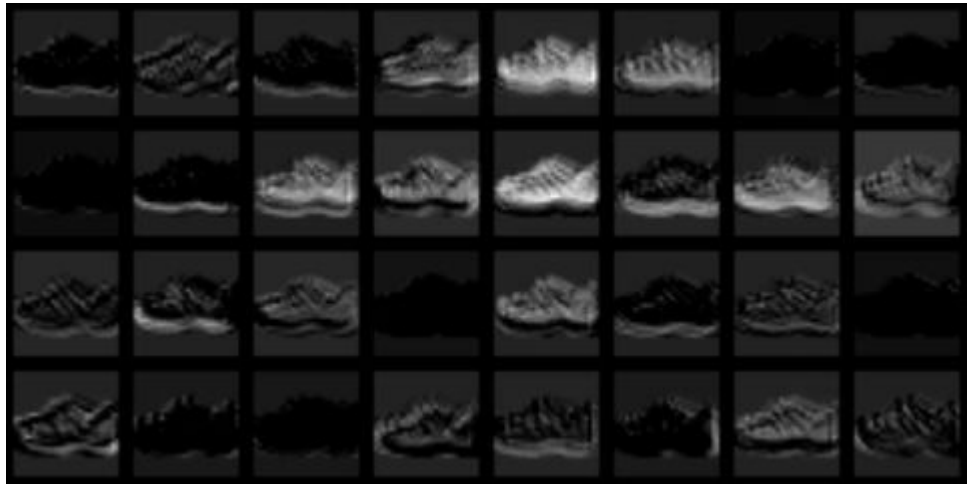
CNN comparisons (4/4) - Histograms

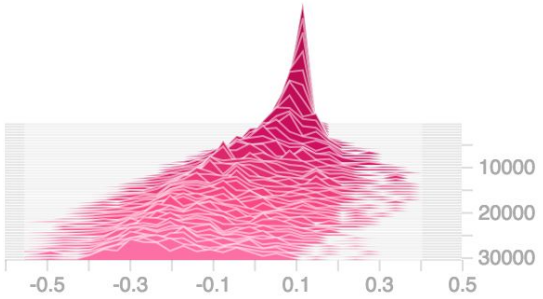
	Convolutional Layer #2		
	CNN with 1 conv layer	CNN with 2 conv layers	CNN with 3 conv layers
Bias			
Weight			

CNN comparisons (4/4) - Histograms

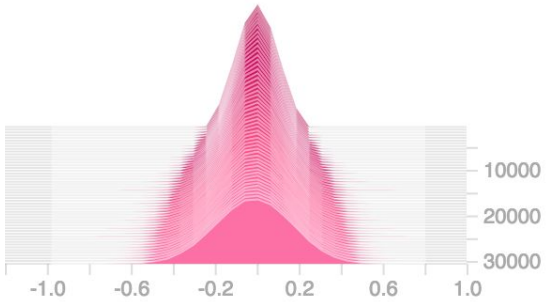
Convolutional Layer #3		
CNN with 1 conv layer	CNN with 2 conv layers	CNN with 3 conv layers

Bias





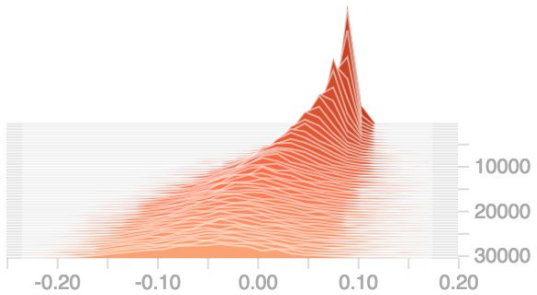
Weight

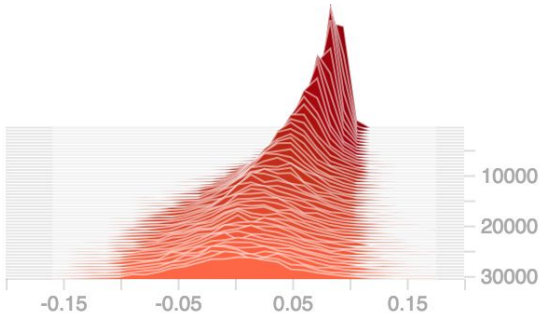


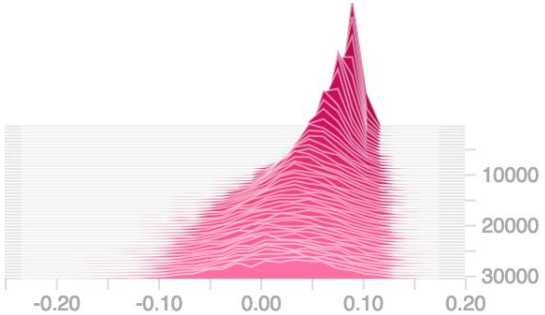
CNN comparisons (4/4) - Histograms

Fully Connected Layer		
CNN with 1 conv layer	CNN with 2 conv layers	CNN with 3 conv layers

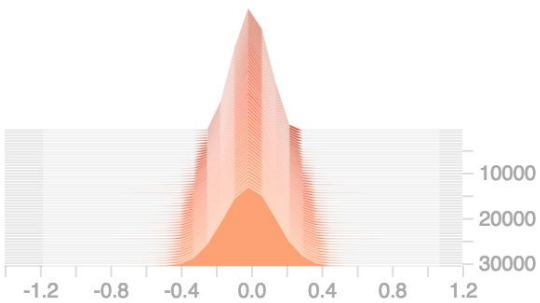
Bias

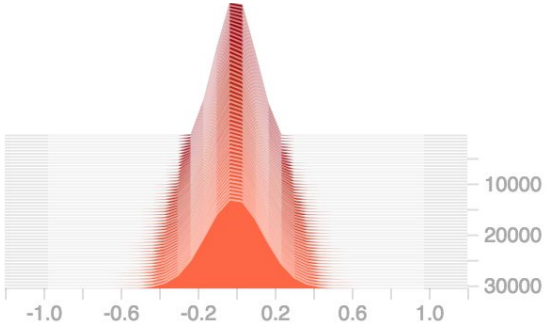


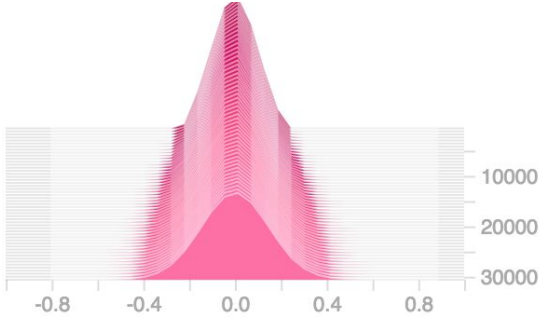




Weight



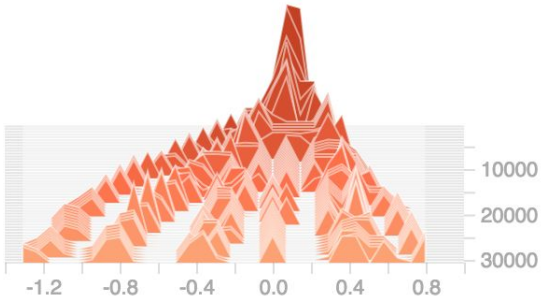


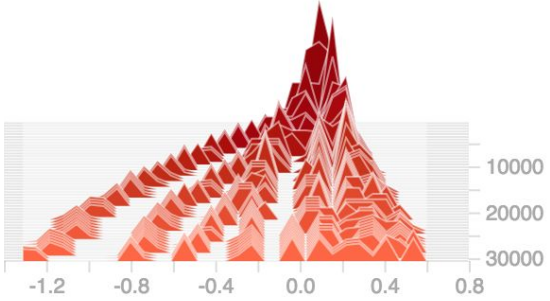


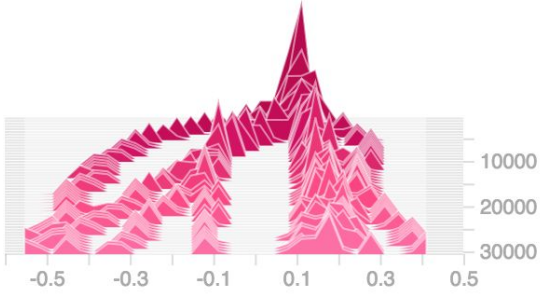
CNN comparisons (4/4) - Histograms

Readout Layer		
CNN with 1 conv layer	CNN with 2 conv layers	CNN with 3 conv layers

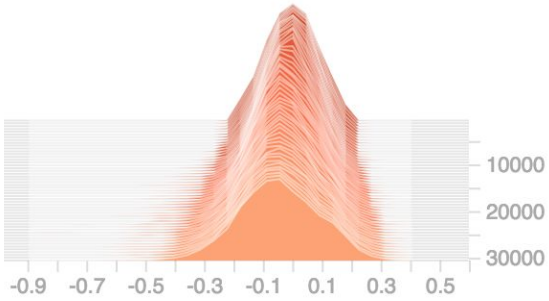
Bias

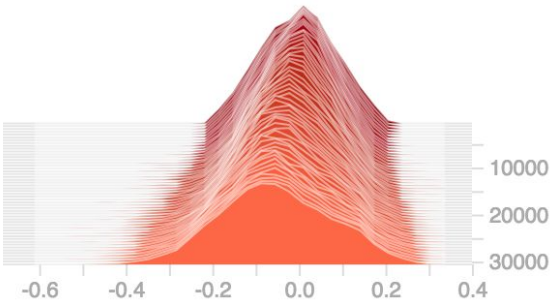


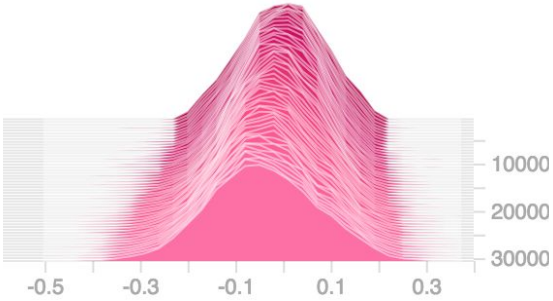




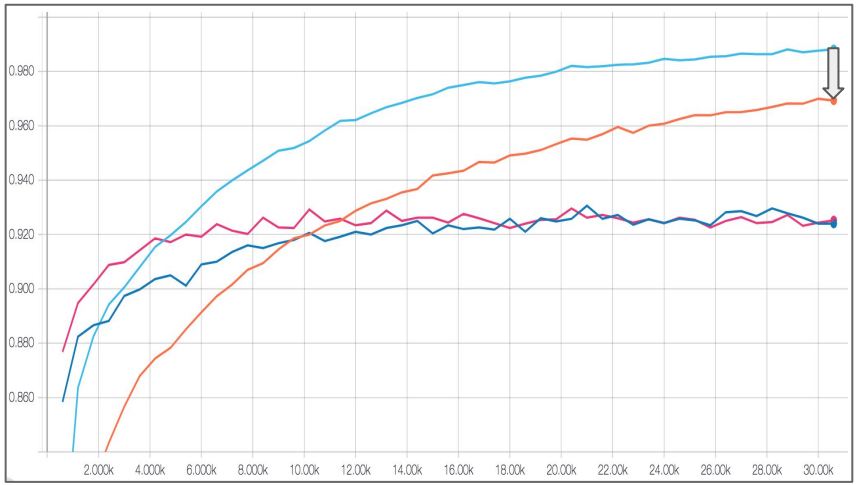
Weight

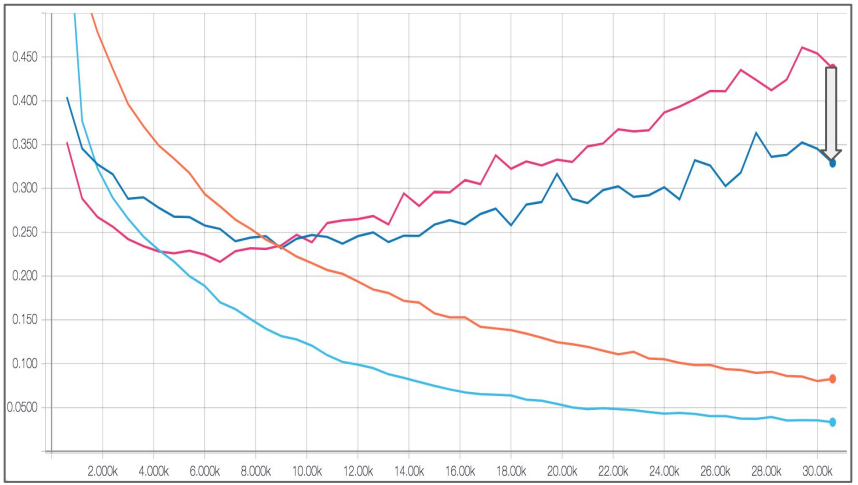






Less Overfitting with Batch Normalization





Accuracy

Very similar validation accuracy for both with and without batch normalization

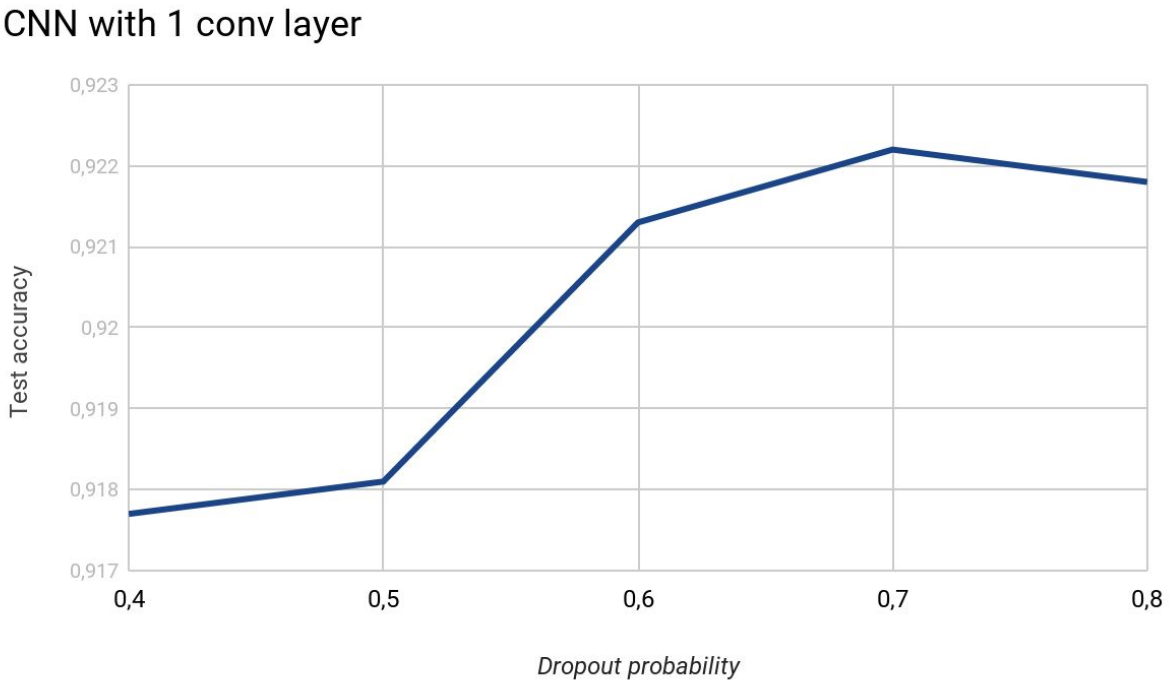
Loss

Batch normalization reduces the loss for the validation data, which implies less overfitting

Name	Smoothed	Value	Step	Time	Relative
CNN_one_conv_layer_dropout_07/summary/train	0.9693	0.9693	30.60k	Sat Jun 2, 19:43:52	4m 14s
CNN_one_conv_layer_dropout_07/summary/valid	0.9240	0.9240	30.60k	Sat Jun 2, 19:43:52	4m 14s
CNN_one_conv_layer_no_batch_norm/summary/train	0.9882	0.9882	30.60k	Sun Jun 3, 13:07:46	21m 34s
CNN_one_conv_layer_no_batch_norm/summary/valid	0.9252	0.9252	30.60k	Sun Jun 3, 13:07:46	21m 34s

Dropout Probability

Can dramatically increase accuracy, with the added benefit of a lower loss



Dropout Probability

0,7

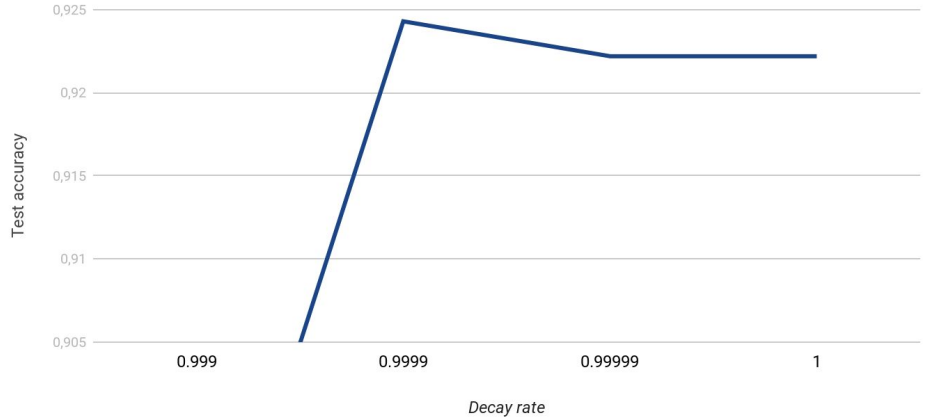
Max Test Accuracy

92,22%

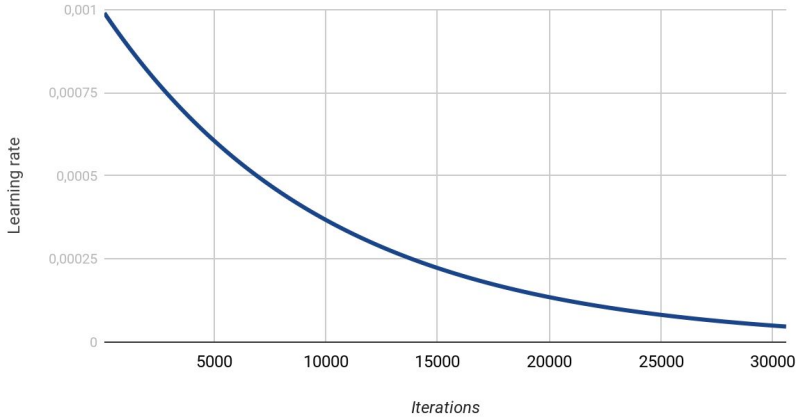
Learning Rate

Lower steps might be needed to allow training, but at the risk of overfitting

CNN with 1 conv layer



Adaptive learning rate



Decay Rate	Initial Learning Rate	Max Test Accuracy
0.9999	0.001	92.43%

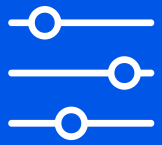
$$L_r[k] = L_r[0] \cdot d_r^{\,k}$$

LEARNING OUTCOME

Our takeaways and ideas

TensorFlow + TensorBoard

Create neural networks of varying complexity



Tuning and Tweaking

Learning rate, batch normalization, dropout probability, regularization, etc.

Visualization

See how the networks evolve as they are being trained

Analysis

Find the optimal model and training strategy for a given dataset

A deeper insight

For future projects we will more easily see the needed type and complexity of a suitable network

THANK YOU