Machine Learning Project: The CIFAR-10 Dataset

Derrick Hines and Tim Racs

University of Bonn

Practical Lab Numerical Simulation P4E1
Algorithms in Machine Learning and Their Applications

July 16, 2019

Overview

- CIFAR-10 dataset
- 60000 32x32 color images in 10 classes ($d = 3072$)
- 6000 images per class
- 50000 training images, 5000 of each class
- 10000 test images, 1000 of each class
- Objective: Build a classifier

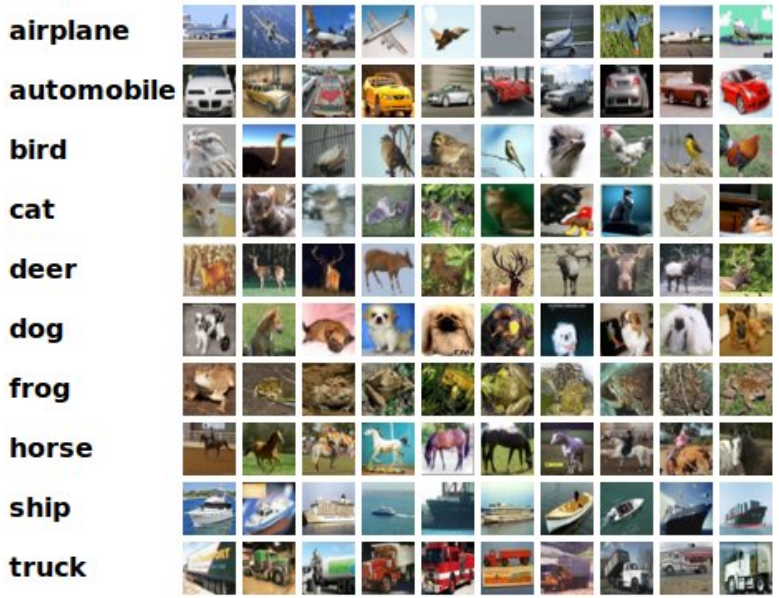
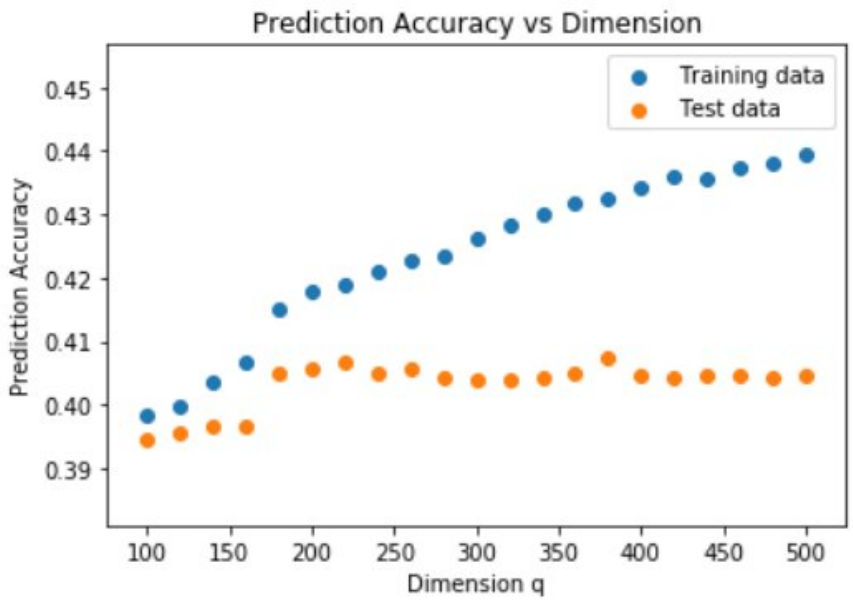


Figure: 10 images from each class: Learning Multiple Layers of Features from Tiny Images, Alex Krizhevsky, 2009.

1. Support Vector Machine
2. Convolutional Neural Network
3. Convolutional Neural Network + SVM

Support Vector Machine

- **Normalization:** Normalized data to $[0, 1]$
- **Dimensionality Reduction:** PCA with $q = 380$ principal components
- **Explained variance:** 97.49%
- **Classification:** Linear Support Vector Machine with one vs all classification
- **Penalty Parameter:** $C = 0.01$
- **Accuracy on test data:** 40.73%



Support Vector Machine

- All possible incorrect classifications occur
- Cats more likely to be classified as dogs than as cats

Table: Confusion Matrix SVM

	b'airplane'	b'automobile'	b'bird'	b'cat'	b'deer'	b'dog'	b'frog'	b'horse'	b'ship'	b'truck'
b'airplane'	505	45	25	19	20	21	35	57	185	88
b'automobile'	64	501	9	19	19	29	40	62	81	176
b'bird'	116	56	212	71	104	84	176	91	56	34
b'cat'	60	82	70	186	43	194	170	58	48	89
b'deer'	68	41	101	52	261	71	200	135	33	38
b'dog'	54	72	70	119	68	328	105	78	65	41
b'frog'	25	56	52	72	70	67	551	51	22	34
b'horse'	54	63	51	45	64	70	43	478	39	93
b'ship'	144	77	8	14	7	37	18	22	564	109
b'truck'	69	197	13	15	8	24	47	51	89	487

Figure: Two cats classified as dogs

- There is room for improvement!

Neural Network Architecture: 6 convolutional layers

- **Convolution-Layer**: 16 filters, $(3, 3)$ kernel, relu-activation, padding
- **BatchNormalization**
- **Convolution-Layer**: 32 filters, $(3, 3)$ kernel, elu-activation, padding
- **BatchNormalization**
- **MaxPooling**: $(2, 2)$ pool
- **Dropout**: $p = 0.25$
- **Convolution-Layer**: 64 filters, $(3, 3)$ kernel, elu-activation, padding
- **BatchNormalization**
- **Convolution-Layer**: 64 filters, $(3, 3)$ kernel, elu-activation, padding
- **BatchNormalization**
- **MaxPooling**: $(2, 2)$ pool
- **Dropout**: $p = 0.35$
- **Convolution-Layer**: 128 filters, $(3, 3)$ kernel, elu-activation, padding
- **BatchNormalization**
- **Convolution-Layer**: 128 filters, $(3, 3)$ kernel, elu-activation, padding
- **BatchNormalization**
- **MaxPooling**: $(2, 2)$ pool
- **Dropout**: $p = 0.45$
- **Flatten**
- **Dense**: 10 outputs, softmax-activation

Neural Network Parameters: 296 810

t	Layer (type)	Output Shape	Param
	conv2d_1 (Conv2D)	(None, 16, 32, 32)	448
	conv2d_2 (Conv2D)	(None, 16, 32, 32)	9248
	max_pooling2d_1 (MaxPooling2D)	(None, 8, 16, 32)	0
	dropout_1 (Dropout)	(None, 8, 16, 32)	0
	conv2d_3 (Conv2D)	(None, 8, 16, 64)	18496
	conv2d_4 (Conv2D)	(None, 8, 16, 64)	36928
	max_pooling2d_2 (MaxPooling2D)	(None, 4, 8, 64)	0
	dropout_2 (Dropout)	(None, 4, 8, 64)	0
	conv2d_5 (Conv2D)	(None, 4, 8, 128)	73856
	conv2d_6 (Conv2D)	(None, 4, 8, 128)	147584
	max_pooling2d_3 (MaxPooling2D)	(None, 2, 4, 128)	0
	dropout_3 (Dropout)	(None, 2, 4, 128)	0
	flatten_1 (Flatten)	(None, 1024)	0
	dense_1 (Dense)	(None, 10)	10250

t		
	Total params:	296,810
	Trainable params:	296,810
	Non-trainable params:	0

Neural Network Results

- 130 epochs
- 77.24% accuracy on test data (much better than SVM)

Table: Confusion Matrix CNN

	b'airplane'	b'automobile'	b'bird'	b'cat'	b'deer'	b'dog'	b'frog'	b'horse'	b'ship'	b'truck'
b'airplane'	805	7	32	17	10	5	12	5	76	31
b'automobile'	18	822	9	14	1	4	14	3	40	75
b'bird'	47	0	751	47	43	24	60	17	7	4
b'cat'	22	6	77	638	35	107	76	22	11	6
b'deer'	17	2	87	49	702	16	73	43	10	1
b'dog'	11	2	69	227	36	561	47	41	4	2
b'frog'	4	1	29	45	17	15	885	1	2	1
b'horse'	15	1	45	42	37	37	13	806	2	2
b'ship'	36	6	10	15	3	2	5	0	909	14
b'truck'	20	40	10	21	2	5	10	16	31	845

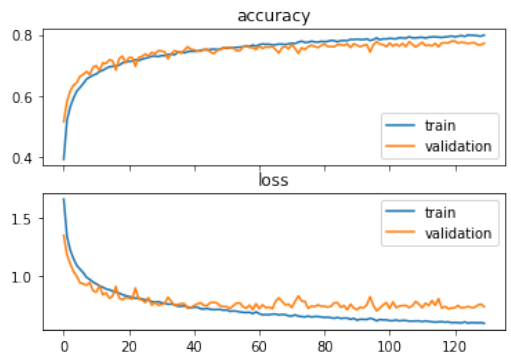
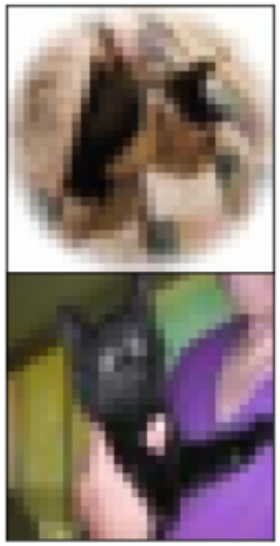


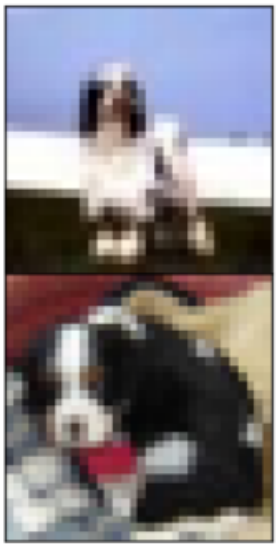
Figure: Accuracy and loss of the neural network

Some misclassified images by the CNN

Cats classified as dogs

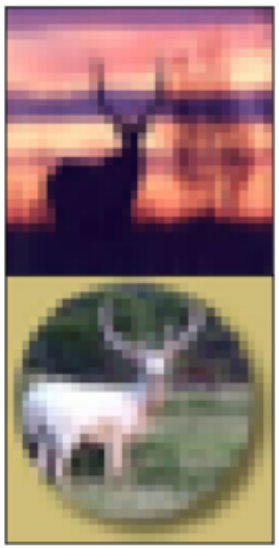


Dogs classified as trucks



Automobiles classified as horses

Deer classified as automobiles

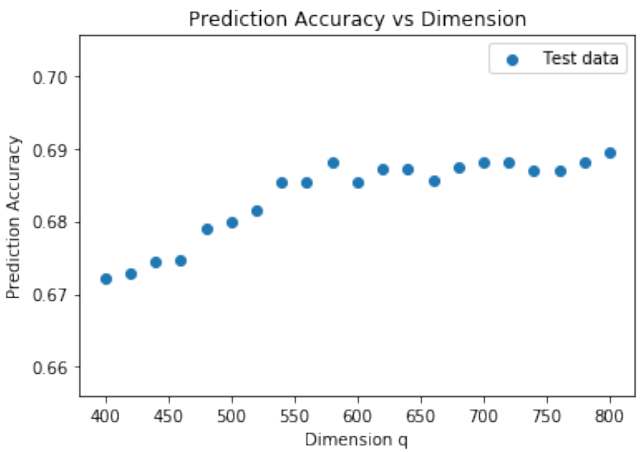


Neural Network and Support Vector Machine

- Using the outputs of the neural network after the flattening-layer as inputs
- 68.95% accuracy with 800 principal components
- Not as good as the neural network

Table: Confusion Matrix CNN + SVM:

	b'airplane'	b'automobile'	b'bird'	b'cat'	b'deer'	b'dog'	b'frog'	b'horse'	b'ship'	b'truck'
b'airplane'	748	18	36	17	15	11	11	17	75	52
b'automobile'	29	796	6	9	4	10	17	11	30	88
b'bird'	82	7	540	58	106	59	74	41	18	15
b'cat'	27	11	78	450	69	173	90	43	29	30
b'deer'	32	5	75	54	639	29	65	82	13	6
b'dog'	21	8	62	141	55	577	37	68	15	16
b'frog'	10	13	53	53	35	21	793	7	10	5
b'horse'	21	1	33	42	59	61	6	750	7	20
b'ship'	53	37	13	15	8	8	11	3	827	25
b'truck'	37	95	14	14	4	10	11	17	23	775



References

- Learning Multiple Layers of Features from Tiny Images, Alex Krizhevsky, 2009.
- Deep Neural Networks, Jochen Garcke, Bastian Bohn, Jannik Schürg, 2019.