

BSc Thesis:
Self-minimizing deep convolutional
neural network for image processing

Jenei Bendegúz

August 10, 2018

Abstract

Lórum ipse: a jorcsó hat a zatékony kötvény fogta, cserzel, esztek. Művészileg is pityókony vitos tegeszkétet, műven „padt teendőt”. Rázsási, hogy ami tekély, ahhoz csak óvatosan nyalkodik cipkelnie. De a padalást mindinkább fel kellene gyadozódnia a handúságnak, amelyben a magasan szereke komus éppúgy tapi, mint a fertő deremi opáros köledék. Tehát minél több eres, bolással pélva alanság kell ontoroznia. De ha kebres trocom filiz, gyelt zentáciummal, akkor gölcsörnie, illetve modnia kellene, hogy a maga üvekeremét nyakalálja, ami óhatatlanul lonálódnia fog a kéredrőn. - Egyelőre a selyin kívül nincs ezes tária - írálta okság tikadmás. Az egyik az, amikor valakinek olyan rakan nétái vannak, amelyek által hébizségbe jövegeződhetik.

Contents

1	Introduction	3
1.1	Problem definition	3
2	The goal of the network	4
2.1	Edge detection	4
2.1.1	Edges	4
2.1.2	Detecting edges	4
2.2	Conventional edge detectors	4
3	Implementation	5
3.1	Tools	5
3.2	Network structure	5
3.3	The code	5
3.4	Methods	5
4	Results	6
4.1	Metrics	6
4.2	Tests	6
4.3	Strategies	6
5	Conclusion	7

1 Introduction

1.1 Problem definition

The goal is to make a framework which is capable of training a deep convolutional neural network for simple two-dimensional image-processing tasks, and make this network as small as possible without sacrificing its accuracy.

A deep neural network's size is defined by two parameters: number of layers, and number of neurons in each layer, this can be unique amongst layers. These parameters must be specified when building a neural network for training.

This thesis will try to provide automatic strategies to come up with optimal values for these two parameters, as opposed to defining them with trial & error, repeated manual testing, or based on experience.

2 The goal of the network

2.1 Edge detection

2.1.1 Edges

An edge in image processing is a sudden change in pixel intensity through an image. This is common on the edges of objects, or at the intersection of different colors in a pattern. There is no criteria for the actual required value of the change intensity-change per pixel, so there is no one right solution for edge detecting an image. There are multiple techniques available for detecting edges, each having it's own strengths and weaknesses.

There are two classes of edge-detection: continuous and discrete.

Continuous detectors produce an image of the same dimensions as the source image, where a pixels intensity value corresponds to how strong of an edge is present at that position in the original image.

Discrete detection uses boolean values to tell if an edge is present or not. Continuous detections can be converted to discrete with thresholding, or using adaptive thresholding on parts of a continuous edge-detected image.

2.1.2 Detecting edges

Edges can occur in any direction on an image. The detectors often use derivatives and gradients to determine the sudden drops and rises in intensity, among both axes. Since most detectors use multiple steps until they produce the final edge-map, it makes sense to use a deep, multi layered network, here a layer can more-or less represent a step in the process.

Convolving an image with an appropriate mask is also used in edge-detection. A convolutional network is capable of learning and applying combinations of masks automatically.

2.2 Conventional edge detectors

3 Implementation

3.1 Tools

3.2 Network structure

3.3 The code

3.4 Methods

4 Results

4.1 Metrics

4.2 Tests

4.3 Strategies

5 Conclusion

List of Figures

List of Tables