FACULTY OF SCIENCE
UNIVERSITY OF COPENHAGEN

MR Image texture analysis applied to the diagnosis of Alzheimer's Disease

Mathias Bjørn Jørgensen & Mirza Hasanbasic

6th June 2016

Contents

List of Figures

List of Tables

List of Corrections

Mathias Bjørn Jørgensen & Mirza Hasanbasic

Abstract

This report will examine MRI scans of brains, using image texture analysis and machine learning

Chapter 1

Introduction

Alzherimer's Disease (AD) is the most common cause of dementia among people and is a growing problem in the aging populations. It has a big impact on health services and society as life expectancy increases. In 2010 the total global costs of dementia was estimated to be about 1% of the worldwide gross domestic product[1]. AD is the cause in about 60%-70% of all cases of dementia[1] and about 70% of the risk is believed to be genetic [2]. Currently there are no way to cure dementia or to alter the progressive course. But however, much can be done to support and improve the lives if AD is found in the early stage of progression [1]

In this report we will examine MRI data of the hippocampus using image texture analysis and apply machine learning, for this we have 50 normal controls and 50 Alzheimer's Disease (AD) patients.

We will be using two different image texture analysis method, one which will me in 2D[3][4] and the other one will be in 3D[5], from which we calculate the data to the gray level co-occurrence matrix (GLCM).

1.1 Problem Definition

Is it possible to classify MRI data of the hippocampus into groups of healthy controls vs Alzheimer's patient, using a predefined set of image texture metrices, with an accuracy greater than 80%?

Will there be a difference in diagnosing AD successfully by calculating the co-occurence matrix in 3D compared to 2D.

[1]With the terms as of direct medical costs, direct social costs and costs of informal care

Chapter 2

Data

The MRI head scans were acquired on a General Electric 3-T for all our 100 patients, were we got 50 normal subjects and 50 AD patients.

Chapter 3

Method

3.1 Image texture analysis methods

3.1.1 Co-occurrence matrix

The co-occurence matrix (COM) is second-order statistics methods, which is based on information about gray levels in pair of pixes. The matrix is defined over the image with distribution values at a given offset. Mathematically we have a COM matrix **C** which is defined over an $n \times m$ image **I**, with $\Delta x, \Delta y$ being the parameterized offset, is calculated by

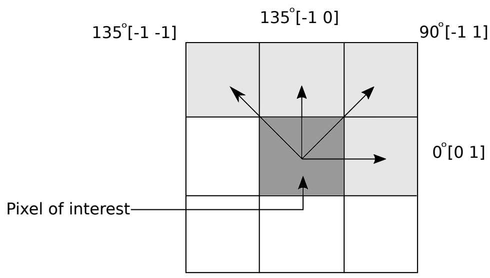
$$C_{\Delta x, \Delta y}(i,j) = \Sigma_{p=1}^{n} \Sigma_{q=1}^{m} \begin{cases} 1, & \text{if } I(p,q) = i \text{ and } I(p+\Delta x, q+\Delta y) = j \\ 0, & \text{otherwise} \end{cases}$$

FiXme Note: find reference

FiXme Note: find reference

The element (5,4) in the COM can be translated to meaning how many times there exist an element in the image with GI **FiXme Note: level or intensity?** 5 and another element offset $\Delta x, \Delta y$ from the originial with greyscale intensity (GI) 4, i.e. if the offset is (1,0) and the first element is (x,y)(4,3) with GI 5 it would mean that element (x,y)(5,3) would have GI 4. If COM(5,4) is ten, it translates into there being ten instances with element (x,y) = 5 and (x+Δx,y+Δy) = 4.

FiXme Note: level or intensity?



2DCOM

Figure 3.1: Example of the offsets for the 2D

I

1	1	5	6	8
2	3	5	7	1
4	5	7	1	2
8	5	1	2	5

GLCM

	1	2	3	4	5	6	7	8
1	1	2	0	0	1	0	0	0
2	0	0	1	0	1	0	0	0
3	0	0	0	0	1	0	0	0
4	1	0	0	0	1	0	0	0
5	1	0	0	0	0	1	2	1
6	0	0	0	0	0	0	0	1
7	2	0	0	0	0	0	0	0
8	0	0	0	0	1	0	0	0

Figure 3.2: Example how the values in the GLCM are calclulated of the 4-by-5 image I. Element (1, 1) in the GLCM contains the value 1 because there is only one instance inf the image where two horizontally adjacent pixels have the values 1 and 1.

GLCM

A single image have multiple COMs as different offsets creates different relations. Consider a 3×3 matrix looking at element (2,2) we can then create eight different offsets, (1,0),(1,1), (0,1)),(-1,1),(-1,0),(-1,-1),(0,-1),(1,-1), however they are not unique. **FiXme Note: Lave 3x3 matrice med offsets.**

FiXme Note: Lave 3x3 matrice med offsets.

Focusing on the two offsets (1,0), (-1,0) in element (2,2) and (1,2) with GI 1 and 2 respectfully increases the entry $\text{COMs}_{1,0}(1,2)$ and $\text{COMs}_{-1,0}$ (2,1) with one, showing that $\text{COMs}_{1,0}{}^{\mathbf{T}} = \text{COMs}_{-1,0}$. There exist the same relation between (1,1)(-1,-1), (0,1)(0,-1), and (-1,1)(1,-1). This leaves four different offsets for analysis (1,0),(1,1), (0,1)),(-1,1) in general (d,0),(d,d),(0,d),(-d,d) where d is the distance which are commonly named angles 0°, 45°, 90° and 135°. **FiXme Note: show 3x3 matrix with angles clearly visible**

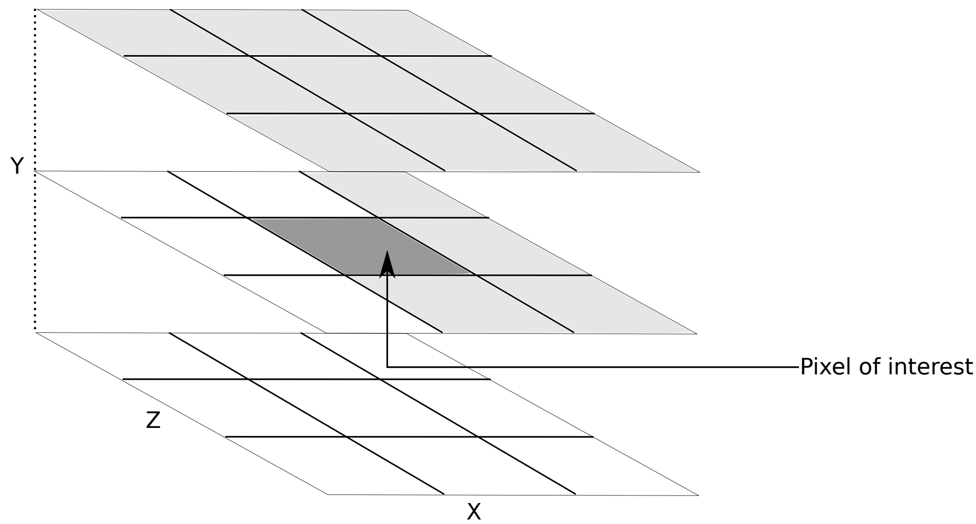
FiXme Note: show 3x3 matrix with angles clearly visible

The co-occurrence matrix is quadratic with the number of rows and columns equal to the amount of GI, for example if we have 256 GI we get a 256×256 COM. nd [0 1]]FiXme Note: Show example on a 5x5 matrix with GI 8, offset [1 0] and [0 1]**FiXme Note: Show example on a 5x5 matrix with GI 8, offset [1 0] and [0 1]**

a

Extending this method to three-dimensions it is necessary to look on how the offsets are defined because the size of the COM is defined by the amount of GIs and not by the images it is derived from. Considering a $3 \times 3 \times 3$ matrix we have a possible of 26 offsets. In two-dimensions it is possible to eliminate half of the offsets because of the relation $\text{COM}^{\mathbf{T}}_{d,d} = \text{COM}_{-d,-d}$, and it is the same case in three-dimensions with the relation being $\text{COM}^{\mathbf{T}}_{d,d,d} = \text{COM}_{-d,-d,-d}$. This leaves 13 offsets which are illustrated below.
FiXme Note: 3 gange 3 gange, farv de ønskede dele eller noget

FiXme Note: 3 gange 3 gange, farv de ønskede dele eller noget



3DCOM

Figure 3.3: Example of the offsets for the 3D

3.1.2 Texture features from co-occurrence matrix

3.2 Machine learning methods

3.2.1 Crossval

3.2.2 Feature selection

3.2.2.1 Naive

3.2.2.2 Sequential Forward Feature

3.2.3 K-nearest neighbors algorithm

k-NN for short is a method that is used for classification and regression. Where the output is a class and member of this class, and this object is classified by its neighbors. For instance, if we chose k to 1, then the object will be assigned to the class of the single nearest neighbor.

The algorithm consist of training examples, that are vectors in multidimensional space, with each its label. The most used distance metric is Euclidean distance.

The drawback of k-NN is that classification can be skewed in that way, that the more frequent class tend to dominate the prediction of new examples, because they tend to be common among the k-NN due to their large number.

The way we wish to implement the k-NN in matlab is, first we handle the data, then we will calculate the distance between two data instances and after that, we can locate k most similar data instances and generate a response from a set. After all this is done, we have to summarize the accuracy of predictions.

Dette vil være en lille introduktion til de tools vi bruger til at analysere vores data med. Da vores data er MRI skanninger af hjernen, som er nogen voxels[1] som bliver repræsenteret i 3D.

3.3 Erode

Normalt brugte man dette til binære billeder, men senere hen er det udvidet til også at omfatte grayscale billeder. Grunden til at dette bruges er for at fjerne støj på billedet.

Vi bruger Erode på vores MRI scanning, da der kan være gray-bit mix og derfor fjerner overflydig og blandet data. Hvis vi starter på hvordan 2D virker, så

For at illustrere hvordan erosion virker, gør vi det på et 2D plan, betragt figur 3.4, hvor vi bruger et plus til at fjerne støj.

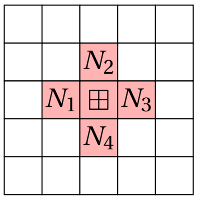


Figure 3.4: Text

Så med figur 3.4, bruger vi denne på figur 3.5.

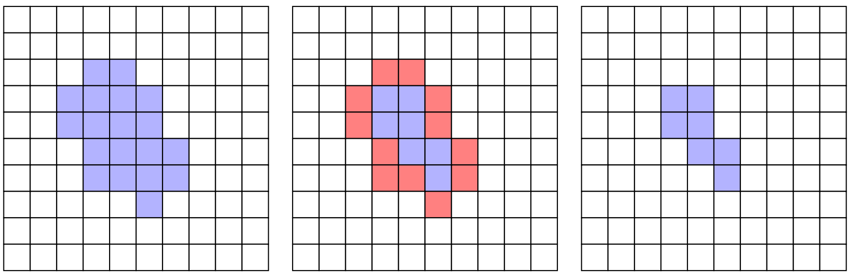
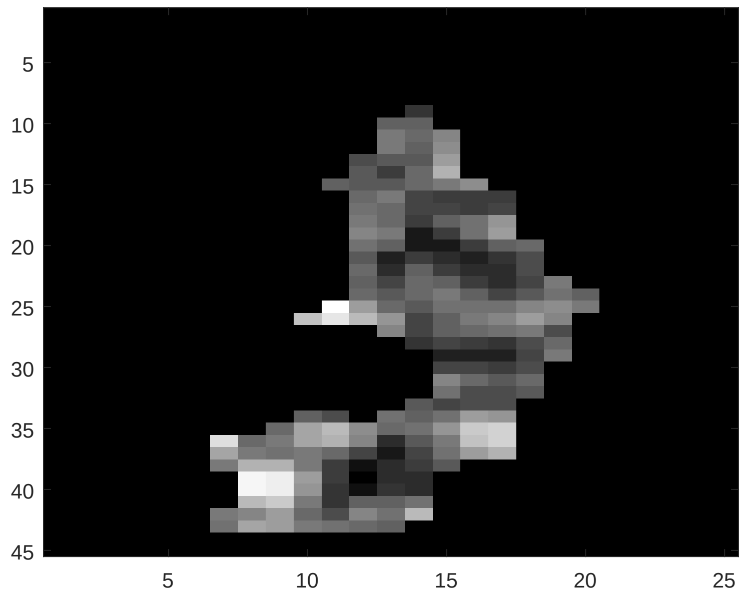


Figure 3.5: Left: Middle: Right:

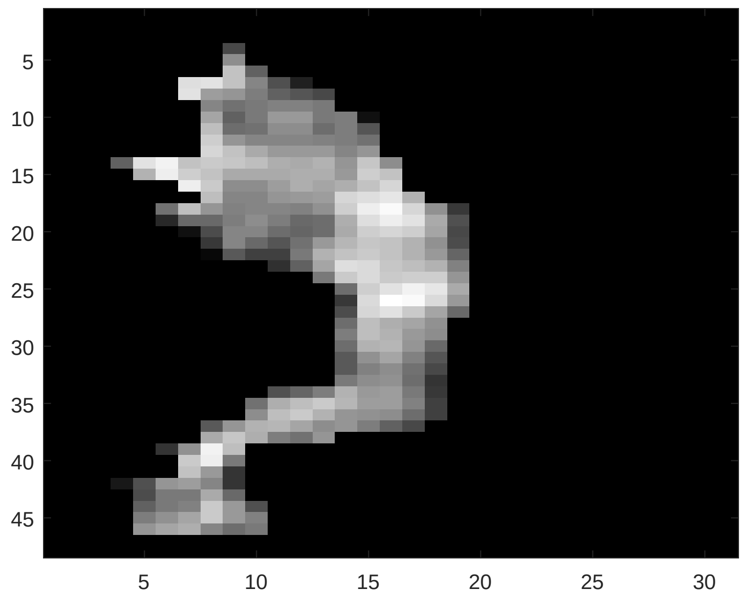
Således er støjen nu fjernet. Vi vil udvide dette til 3D, da vores MRI er i 3D. Som det ses på figur 3.4 har denne 4 naboer den tjekker, når man udvider til 3D, får vi 2 nye naboer, dvs 6 naboer i alt. Hvis en af dem er udenfor den ønskede matrix, ekskluderes pixlen.

[1] noget her



erodeslice

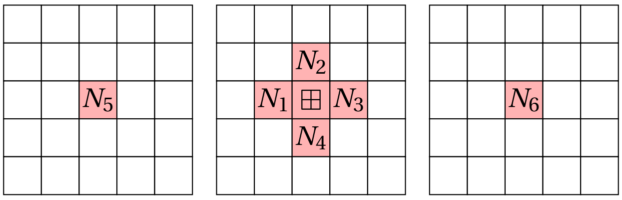
Figure 3.6: Example of the offsets for the 3D



noterodeslice

Figure 3.7: Example of the offsets for the 3D

Da vores data er i 3D, så i udvider vi erosion, hvor det stadig er et plus, men med 2 ekstra naboer Now we expand this cross for the 3D and its the same concept for 3D. Now it 6 neighbours instead of 4, where we expand it for the 3D



Erosion3D

Figure 3.8: Text

FiXme Note: Vi billede af eroded og original mask

FiXme Note: Vi billede af eroded og original mask

Image texture PCA Principal Component Analysis Application to images Machine learning (Knn, Ann, Gaussian)

noget tekst

FiXme Fatal: her er noget galt

noget mere tekst **FiXme Note: en note**

FiXme Note: en note

endnu mere tekst

FiXme Warning: en advarsel – noget er helt forkert

og til slut mere tekst **FiXme Error: en fejl**

FiXme Error: en fejl

Chapter 4

Implementation

Chapter 5

Result

Chapter 6

Discussion

Chapter 7

Conclusion

Appendices

Appendix A

Co occurrence matrix derivation features

ationfeatures

$$C_x(i) = \sum_{j=1}^{N} C(i,j)$$

$$C_y(i) = \sum_{i=1}^{N} C(i,j)$$

$$C_{x+y}(k) = \sum_{\substack{i=1 \\ i+j=k}}^{N} \sum_{j=1}^{N}, \quad \text{k = 2, 3, ... , 2N}$$

$$C_{x+y}(k) = \sum_{\substack{i=1 \\ |i-j|=k}}^{N} \sum_{j=1}^{N}, \quad \text{k=0,1,...,N-1}$$

$$f_1 = \sum_{i=1}^{N} \sum_{j=1}^{N} \{C(i,j)\}^2 \tag{A.0.1}$$

AngularSecond

$$f_2 = \sum_{n=0}^{N-1} n^2 \{C_{x+y}(k)\} \tag{A.0.2}$$

Contrast

$$f_3 = \frac{\sum_{i=1}^{N} \sum_{j=1}^{n} ijC(i,j) - \mu_x \mu_y}{\sigma_x \sigma_y} \tag{A.0.3}$$

Correlation

$$f_4 = \sum_{i=1}^{N} \sum_{j=1}^{N} (i-\mu)^2 C(i,j) \tag{A.0.4}$$

Variance

$$f_5 = \sum_{i=1}^{N} \sum_{j=1}^{n} \frac{1}{1+(i-j)^2} C(i,j) \tag{A.0.5}$$

InverseDiffer

$$f_6 = \sum_{i=2}^{2N} i C_{x+y}(i) \tag{A.0.6}$$

SumAverage

$$f_7 = \sum_{i=2}^{2N} (i-f_6)^2 C_{x+y}(i) \tag{A.0.7}$$

SumVariance

$$f_8 = \sum_{i=2}^{2N} C_{x+y}(i) \log(C_{x+y}(i)) \tag{A.0.8}$$

SumEntropy

$$f_9 = -\sum_{i=1}^{N} \sum_{j=1}^{N} C(i,j) \log(C(i,j)) \tag{A.0.9}$$

Entropy

$$f_{10} = \text{variance of } C_{x-y} \tag{A.0.10}$$

DifferenceVar

$$f_{11} = -\sum_{i=0}^{N-1} C_{x-y}(i) \log(C_{x-y}(i)) \tag{A.0.11}$$

DifferenceEnt

Bibliography

Who [1] Dementia. http://www.who.int/mediacentre/factsheets/fs362/en/. Accessed: 2016 April.

zheimerLancet [2] Anne Corbett Carol Brayne Dag Aarsland Emma Jones Clive Ballard, Serge Gauthier. Alzheimer's disease. *Lancet*, page 13, March 2011.

MRfreeborough [3] Peter A. Freeborough & Nick C. Fox. Mr image texture analysis to the diagnosis and tracking of alzheimer's disease. *IEEE*, 17(3):5, June 1998.

Castellano [4] L.M. Li F. Cendes G. Castellano, L. Bonilha. Texture analysis of medical images. *Neuroimage Laboratory*, page 9, April 2004.

Voxel [5] Sanjay Kalra Rouzbeh Maani, Yee Hong Yang. Voxel-based texture analysis of the brain. *Plos One*, page 19, March 2015.