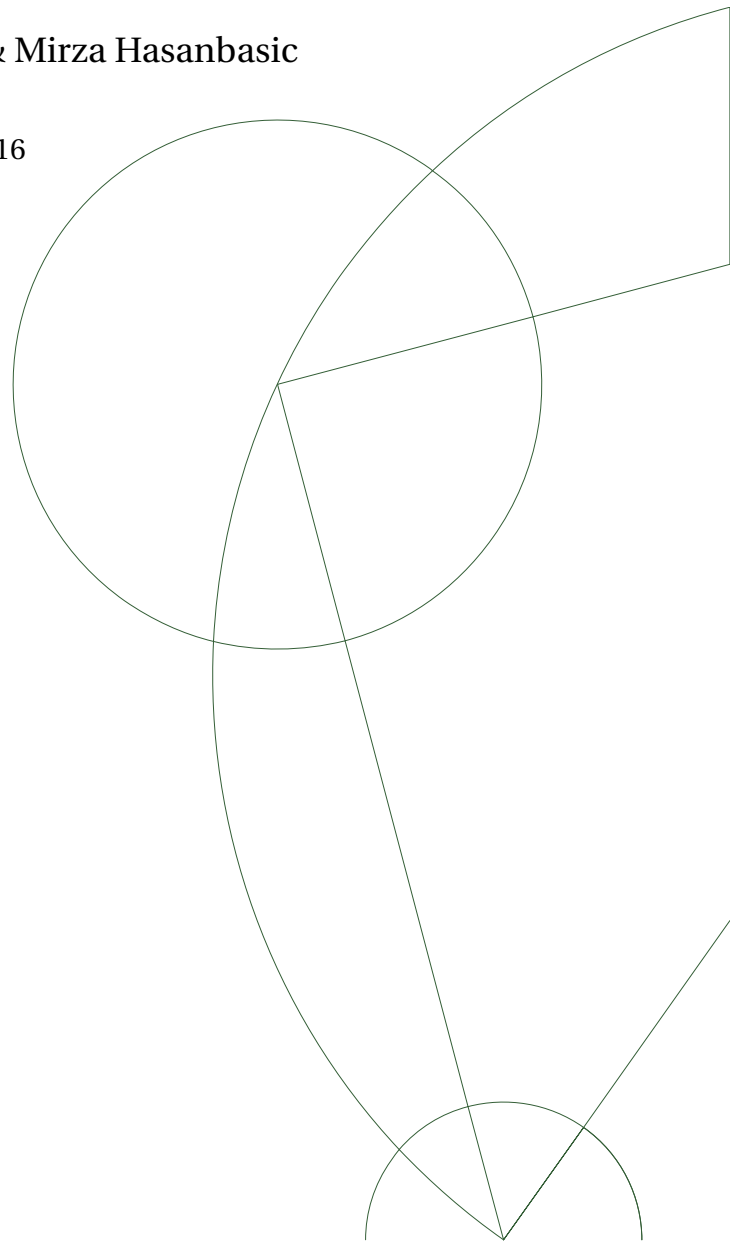




MR Image texture analysis applied to the diagnosis of Alzheimer's Disease

Mathias Bjørn Jørgensen & Mirza Hasanbasic

8th June 2016



Contents

1	Introduction	3
1.1	Problem Definition	3
2	Data	4
3	Method	5
3.1	Image texture analysis methods	5
3.1.1	Co-occurrence matrix	5
3.1.2	Texture features from co-occurrence matrix	7
3.2	Machine learning	7
3.2.1	10-fold cross-validation	7
3.2.2	Feature selection	7
3.2.2.1	Sequential Forward Feature	8
3.2.2.2	Naive	8
3.2.3	K Nearest Neighbors algorithm	8
3.3	Erode	9
4	Implementation	11
5	Result	12
6	Discussion	13
7	Conclusion	14
	Appendices	15
A	Co occurrence matrix derivation features	16
	Litteratur	18

List of Figures

3.1	Example of the offsets for the 2D	6
3.2	Example how the values in the GLCM are calculated of the 4-by-5 image I. Element (1, 1) in the GLCM contains the value 1 because there is only one instance in the image where two horizontally adjacent pixels have the values 1 and 1. . .	6
3.3	Example of the offsets for the 3D	7
3.5	Text	9
3.7	Text	10

List of Tables

List of Corrections

Note: find reference	5
Note: level or intensity?	5
Note: how does this remove overfitting?	7

Mathias Bjørn Jørgensen & Mirza Hasanbasic

Abstract

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

Chapter 1

Introduction

Alzheimer'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 product¹. 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 diagnosed 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 in order to diagnose AD in patients. Our dataset contain 100 patients 50 control and 50 with AD.

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 metrics, with an accuracy greater than 80%?

Is there a difference in diagnosing AD successfully by calculating the co-occurrence matrix in 3D compared to 2D.

¹With 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, where we got 50 normal subjects and 50 AD patients.

Chapter 3

Method

3.1 Image texture analysis methods

Short intro to what is image texture

3.1.1 Co-occurrence matrix

The co-occurrence 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 \mathbf{C} which is defined over an $n \times m$ image \mathbf{I} , with $\Delta x, \Delta y$ being the parameterized offset, is calculated by

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

FiXme Note: find reference

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 original with greyscale intensity (GI) 4, i.e. if the offset is (,1) 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(4,4) is ten, it translates into there being ten instances with element (x,y) = 5 and (x+ Δx ,y+ Δy) = 4.

FiXme Note:
find
reference
FiXme Note:
level or
intensity?

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.

Focusing on the two offsets (0,1), (0,-1) in element (2,2) and (1,2) with GI 1 and 2 respectively increases the entry $\text{COM}_{s_{1,0}}(1,2)$ and $\text{COM}_{s_{-1,0}}(1,1)$ with one, showing that $\text{COM}_{s_{0,1}}^T = \text{COM}_{s_{0,-1}}$. 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 (0,1),(-1,1), (-1,0),(-1,-1) in general (0,d),(-d,d),(-d,0),(-d,-d) where d is the distance which are commonly named angles 0° , 45° , 90° and 135° .

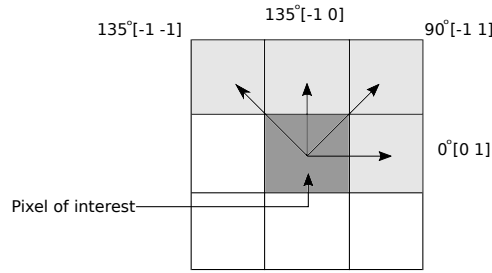


Figure 3.1: Example of the offsets for the 2D

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.

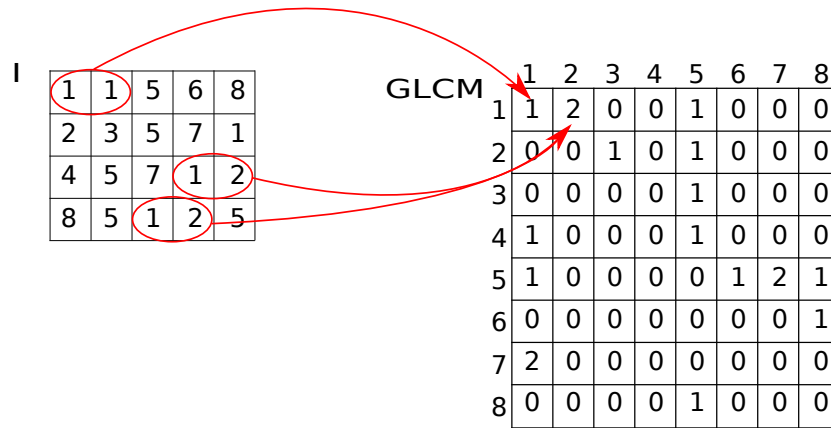


Figure 3.2: Example how the values in the GLCM are calculated of the 4-by-5 image I. Element (1, 1) in the GLCM contains the value 1 because there is only one instance in the image where two horizontally adjacent pixels have the values 1 and 1.

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 $\text{COM}_{d,d}^T = \text{COM}_{-d,-d}$, and it is the same case in three-dimensions with the relation being $\text{COM}_{d,d,d}^T = \text{COM}_{-d,-d,-d}$. This leaves 13 offsets which are illustrated below.

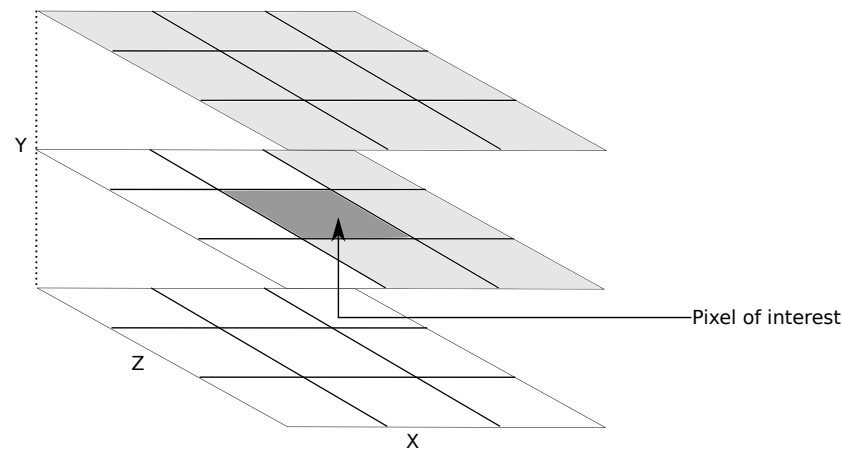


Figure 3.3: Example of the offsets for the 3D

3.1.2 Texture features from co-occurrence matrix

3.2 Machine learning

Short intro to Machine learning

3.2.1 10-fold cross-validation

Given a model with unknown parameters and a training set to which the model can be fit then the fitting process optimizes the model's parameters to fit the training data. Validating this model against independent data (test data) from the same data pool as the training, it will generally turn out that the model does not fit the test data as well as the training data. It is known as overfitting and is a problem when the size of the training data set is small. Cross-validation is used to counteract overfitting.

Dividing the entire data set into 10 groups at random, one subsample is saved for testing and the remaining nine are used to fit the model. The procedure is done so all subsamples get to be used to validate exactly one time and the validation result can then be averaged to produce a single estimation. This solves the problem of overfitting, as the validation data set is never used in to fit the model. **FiXme Note: how does this remove overfitting?**

FiXme Note:
how does
this remove
overfitting?

3.2.2 Feature selection

Feature selection is the process of selection of a subset of relevant features for use in model construction. The main goal of feature selection is to choose a subset of the entire set of input features so the subset can predict the output with an accuracy comparable to using the entire set to predict the output, but with a large reduction in the computational cost. When a data set contains many features that are either redundant or irrelevant it is possible to remove them without incurring much loss of information. Features may be redundant due to the

presence of another feature with which it is strongly correlated, while they may be very informative for the model their information have already been provided by a different feature. Only choosing a subset of the possible features have the added advantage of decreasing the complexity of the model.

3.2.2.1 Sequential Forward Feature

Sequential forward feature selection(SFS) is a greedy algorithm to choose features. It starts off with finding the best possible single feature to describe the model. Given the feature set of that single feature, the next step is to find which other feature would improve the predictiveness of the model and then add that to the set of selected features. It continues to grow the set of selected features, until the goal have been reached. The goal can either be a specific amount of features, a specific accuracy for the model or it can stop if all choices of a new feature would decrease the accuracy. A problem with SFS is due to its greedy nature, there is no guarantee that the first feature selected is part of the optimal solution. Given three features X_1 , X_2 and X_3 where X_1 is the best single feature, X_2 is second best and X_3 the worst, it does not necessitate that pairing X_1, X_2 is better than X_2, X_3 , nor does it secure any other relation, meaning that the SFS does not guarantee the optimal solution.

3.2.2.2 Naive

3.2.3 K Nearest Neighbors algorithm

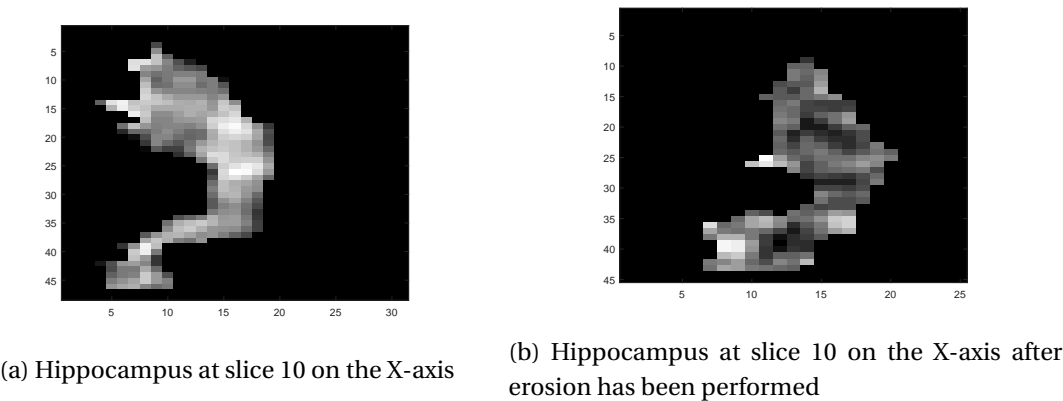
The K Nearest Neighbor (KNN) is a method that is used for classification and regression. It should be noted, that KNN is a non parametric lazy learning algorithm, which means that it does not make any assumptions on the underlying data distribution. In other words, it means that the training phase is fast and KNN keeps all the training data. It should be noted that KNN makes decision based on the entire training data set and in the KNN an object is classified by majority vote of its neighbors, with the object being assigned to the class most common amongst its k_{nn} 's. Since the training phase is minimal, then it should be noted that the testing phase is very costly for KNN in both memory and time and often it can be a worst case for time needed, since all points might take part in the decision making.

Because the KNN classifier predicts the class of a given test observation by identifying the observations that are nearest to it, the scale of the variables matters. Imagine that we have a large scale variables, they will have a much larger effect on the distance between observations and hence the KNN classifier, than variables on a small scale. Often a good way to handle this problem is to standardize the data.

The disadvantage of KNN is that choosing a k may be tricky, so we are left to test the algorithm on multiple k 's and often it needs a large number of samples for better accuracy.

3.3 Erode

Each patients hippocampus has been segmented in the MRI scan. The problems we can run into are that the background will blur with the segmentation i.e. the hippocampus. Performing a erosion can solve this problem and we can focus on the hippocampus, with the maximum number of details. As seen in figure 3.4a the erosion has not been performed yet, but we might have some problems with data surrounding the hippocampus is blurring out the edges of the hippocampus. To solve this, we create a mask to separate the hippocampus from the background data and end up with what is left in figure 3.4b



The erosion we have used is called a city-block metric and can be seen in figure 3.5.

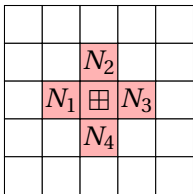
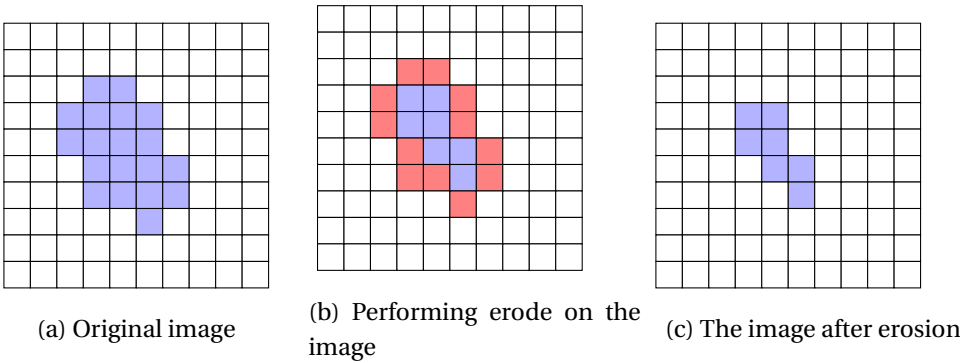


Figure 3.5: Text

To give an example of how the erosion works, it will be illustrated and we will use the city-block metric for this purpose. So we will use figure 3.5 and erode the image in figure ??.



As seen in figure ?? the noise (background) have been removed. This is an example in 2D. Now we wish to extend the erosion city-block to 3D. As seen in figure 3.5 it have 4 neighbours and when we extend this to 3D we will end up with 6 neighbours instead as seen in figure 3.7 and the concept is still the same as in 2D

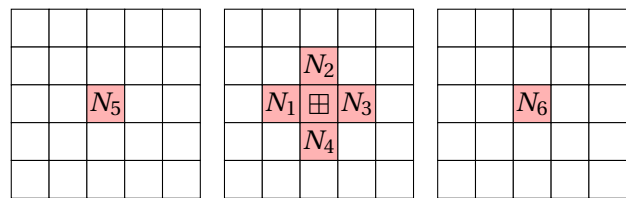


Figure 3.7: Text

Chapter 4

Implementation

Chapter 5

Result

Chapter 6

Discussion

Chapter 7

Conclusion

Appendices

Appendix A

Co occurrence matrix derivation features

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Bibliography

- [1] Dementia. <http://www.who.int/mediacentre/factsheets/fs362/en/>. Accessed: 2016 April.
- [2] Anne Corbett Carol Brayne Dag Aarsland Emma Jones Clive Ballard, Serge Gauthier. Alzheimer's disease. *Lancet*, page 13, March 2011.
- [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.
- [4] L.M. Li F. Cendes G. Castellano, L. Bonilha. Texture analysis of medical images. *Neuroimage Laboratory*, page 9, April 2004.
- [5] Sanjay Kalra Rouzbeh Maani, Yee Hong Yang. Voxel-based texture analysis of the brain. *Plos One*, page 19, March 2015.