| Mälardalens högskola (mdh) |
| --- |
| Intelligent Systems  DVA 406 |
| Project:  Bless You!   * a CBR based sneeze detector   DRAFT 0.1 |
| **Time: Spring 2015**  Göran Forsström  [gfm10001@student.mdh.se](mailto:gfm10001@student.mdh.se)  Simon Palmér  Spr10002@student.mdh.se  Niclas Säll  sns10001@student.mdh.se |
| Teacher: Sara Afshar  sara.afshar@mdh.se  **Date: 2015-03-18** |

## Abstract

This report includes a mini project performed within the course Intelligent Systems, DVA406.   
The project chosen is a Sneeze Detector.

BLA BLA BLA

Contents

1. Abstract 1

2. Introduction 3

3. Related work 3

4. Problem formulation 3

4.1. Background 3

4.2. Problem High Level Description 3

5. Approach 3

5.1. Block Diagram, CBR System 3

5.2. Block Diagram, Feature Detector 4

5.3. Block Diagram, CBR evaluation 5

5.4. Workflow 5

6. Method 6

6.1. Extract Features Design 6

6.2. CBR System Design 7

6.3. Feature Vector 9

6.4. Similarity Functions and Weight values 10

6.5. Case Base Library maintenance 11

6.6. System Performance Evaluation 11

6.7. Feature Vector Evaluation 11

7. Results and analysis (possibly with suggestion for improvement) 12

8. Conclusion 12

9. References 12

## Introduction

In the course DVA406 Intelligent System a mini-project is included as part of the examination. In the project you define a problem, find a solution for it and solve it.

The project chosen is: “Bless You” – a CBR-based Sneeze Detector.

## Related work

Här skriver vi individuellt 2-3 st summeringar var, från artiklar relevanta för kursinnehållet, addera referenser för artiklarna

## Problem formulation

### Background

The current trend to analyze big data is a way to get early indications of events in the society. One such event is the outbreak of an influenza. It is imaginable that sneeze detectors could be used to get an early indication of such an outbreak.

A microphone, placed in e.g. a public library, keeps listening to the sound in the library.

When it detects that someone sneezes a counter is incremented. A supervisory system is able to read the sneeze count at cyclic intervals. The read counter values can be used to detect if a flu is in progress.

### Problem High Level Description

Create a system that can:

Input sound input data.

Extract sound features and place them in a case library.

Compare a new sound with the cases in the library and evaluate if it is a match or not.

Maintain the library by updating it with new cases that gives better performance.

## Approach

The approach to analyze and solve the problem was to create an experimental “Bless You” system prototype that contains the basic CBR functions:

* As the intended system is a server function a simple command line program was suitable as an experimental platform.
* The program output is documented as report files and console printouts.
* A set of random sneezes were collected from the internet as well as sneeze-similar sounds such as coughs together with random sounds.
* The found sounds were captured and edited into standard type of .wav-files: PCM, 16bit, 44.1 KHz, 1 or 2 channels.
* As the time to analyze the sound files was thought to possible be quite long, a set of cached data files (.ftr-files) was introduced to optimize performance.
* The program is controlled by command line parameters and file lists in text files so that it was easy to experiment with different sets of sound files.
* The program is able to run in two modes depending of the parameter setup:
  + Build a case library, then in a loop: extract 1 file from the library and evaluate performance, repeat for each entry in the case library and calculate average performance.
  + Build a case library, then evaluate a single selected file.
* Learning: simulated maintenance phase where the case library is updated from the result of the analyze of a new set of sample files with known status.

### Block Diagram, CBR System

DELETE CHAPTER!

### Block Diagram, Feature Detector

DELETE CHAPTER!

### Block Diagram, CBR evaluation

DELETE CHAPTER!

### Workflow

DELETE CHAPTER!

## Method

The method used is implemented in a program structured as below:

.ftr file to be evaluated file

Extract Features

.wav files

.ftr files

CBR System

Result report

Sound file to be evaluated

Operator Interaction

Figure X: program structure

### Program parameters

Usage:

BlessYou P1 [P2]

where

P1 = name of text file with names of all .wav-files to be examined

P2 = path to directory for created .ftr-files (optional)

Format of list file used as P1: one line per .wav-file:

line = <sound type marker> TAB [<path>]<filename of .wav-file>

<path> = <absolute path> | <relative path to directory of the list file itself>

<sound type marker = ‘0’ if not a sneeze sound

‘1’ if a sneeze sound

‘?’ if unknown contents.

### Extract Features Design

Flow of operations for each case, i.e. .wav-file.

1. Read .wav file contents (16 bit PCM, 44 kHz)
2. If stereo: calculate sample as average of left and right sample.
3. Normalize: search for largest sample, scale all samples so that the largest sample is set to a predefined value, defined in C\_MAX\_POSSIBEL\_VALUE, e.g. 100000
4. Search for start of possible sneeze: search for a sample with an absolute magnitude of at least C\_TRIGGER\_LEVEL\_IN\_PERCENT, e.g. 50%.
5. Evaluate length of suspected sneeze, check for a low level defined in C\_TRIGGER\_OFF\_LEVEL\_IN\_PERCENT, e.g. 10% TBA after at least a time defined in C\_TRIGGER\_OFF\_DURATION\_IN\_MILLI\_SECS, e.g. 1000 ms.
6. Split into N = C\_NR\_OF\_INTERVALS, e.g. 10, equal time interval, indexed as t = [0, N-1]
7. Add a prefetch length, defined in C\_TRIGGER\_PREFETCH\_IN\_MILLI\_SECS of samples before the trigger.
8. Now the feature extraction can be made, the result is stored in a vector of float values, one value per interval. There is one vector per feature type, see table Feature Types! This is based on the suggested Feature Vector in : [EOlsson76 p.29, equation 2.33]  
   FV = [Peak(x), Mean(x), RMS(x), CF(x)]  
   where x is a vector of time-based sample intervals .

10. If there is performance issues in extracting the features, future optimization is possible by caching the vectors for each sound file in a .ftr-file with the same file name as the main file name of the .wav-file.

### CBR System Design

<not changed>

### Feature Vector

<not changed>

### Similarity Functions and Weight values

<not changed>

### Case Base Library maintenance

<not changed>

### System Performance Evaluation

<not changed>

## Results and analysis

The result is a detection rate of approximately XX % when analyzing a new sound file, with a case library of 50 sneeze sound files and 50 none-sneeze sound files randomly chosen among the total about 160 sound samples. However, after maintaining the library by running the maintenance function where remaining 60 sound files are used to optimize the case library the detection rate is increased to XX %.

(possibly with suggestion for improvement)

Suggested improvements:

1. The system can be optimized further by adjusting the weight values per feature type
2. Weights can be added for the intervals when calculating the individual feature type distance
3. The FFT feature can be change to more or less number of samples
4. The FFT frequency band can be adjusted.
5. Introduce noise. The new samples that are introduced for testing do not include any noise, which makes the evaluation simpler.
6. Use a microphone to continuously listen and evaluate if sneezes occur.

## Conclusion

The program manages to do a correct evaluation in about XX % of the cases which is better than

## References

Lägg till referenser