Bless You*!*

- a CBR based sneeze detector
- DVA406 Intelligent Systems Seminar 2015-03-19
  - Simon Palmér
  - Niclas Säll
  - Göran Forsström

Background

- Early indications of events in the society.
- One such event is the outbreak of an influenza.
- Sneeze detectors can early detect such an outbreak.

Use Case

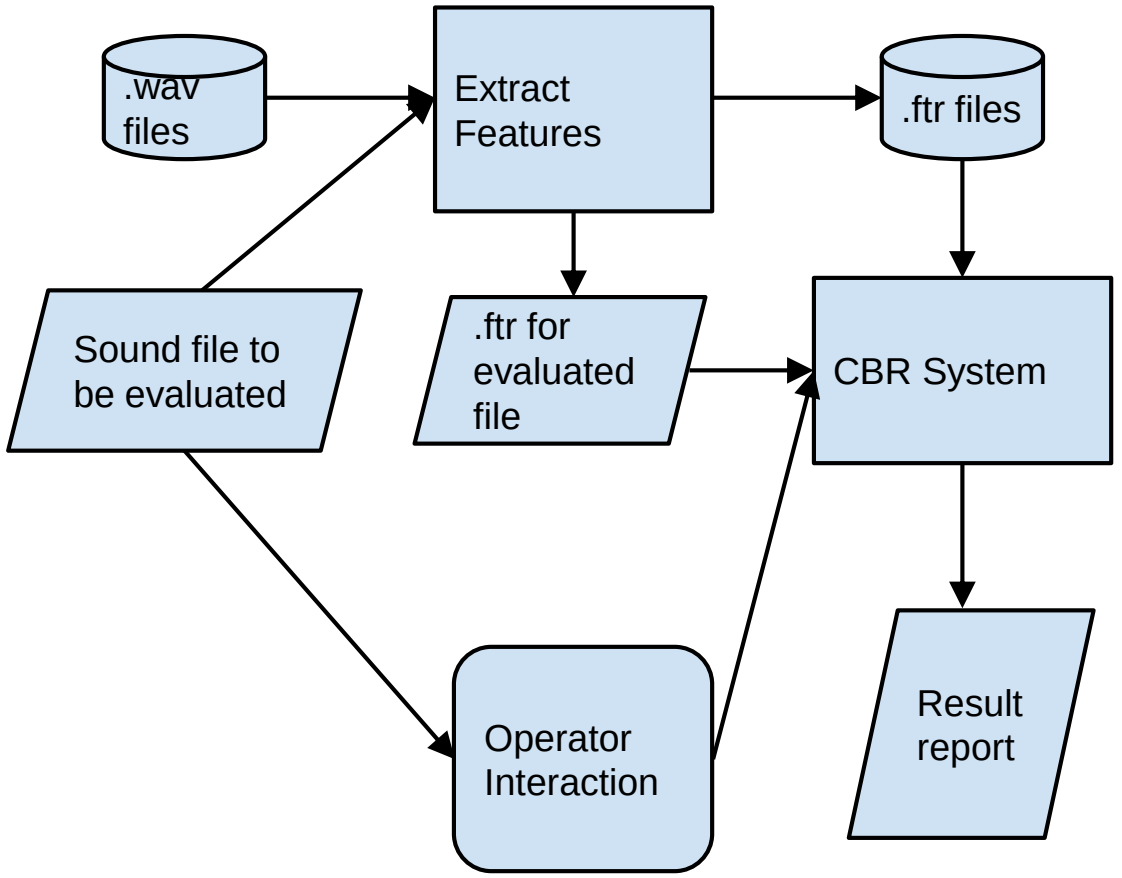
- A microphone listens to the sound in a public place.
- When it detects that someone sneezes a counter is incremented.
- A supervisory system reads the sneeze count at cyclic intervals.
- The read counter values can be used to detect if a flu is in progress.

Our Program

- Limitations
  - No continous evaluation
  - Uses sound samples
  - Console interface

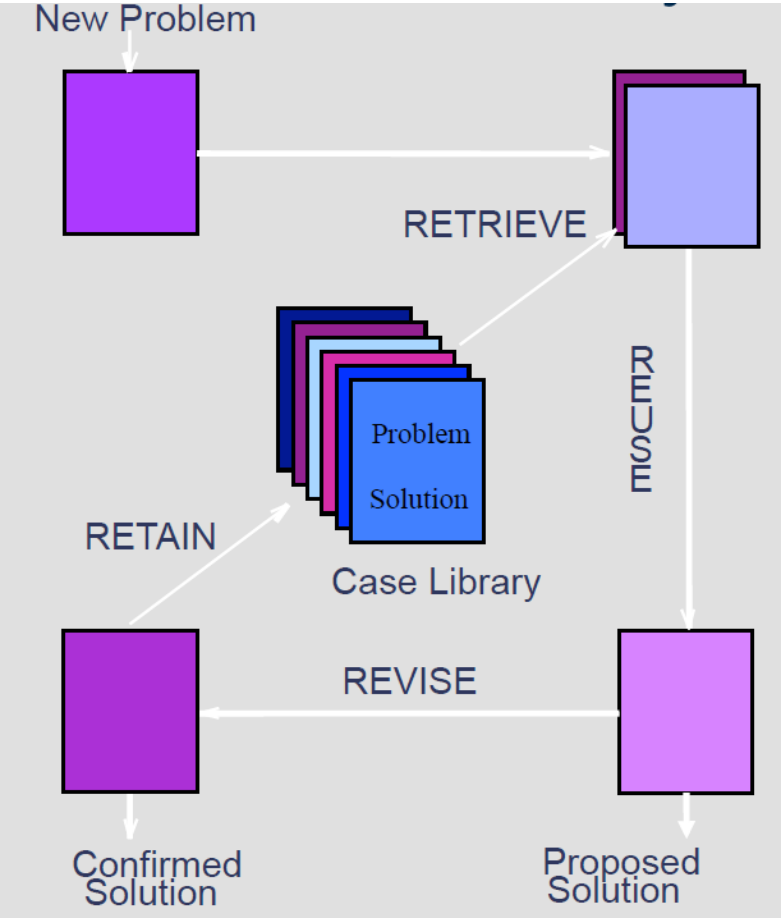
System Overview

- Command Line Program
  - Batch mode
  - Interactive mode
- About 160 sound samples
  - 100 sneezes
  - 60 none-sneezes
- Text report:
  - Proposed sound type
  - Statistics



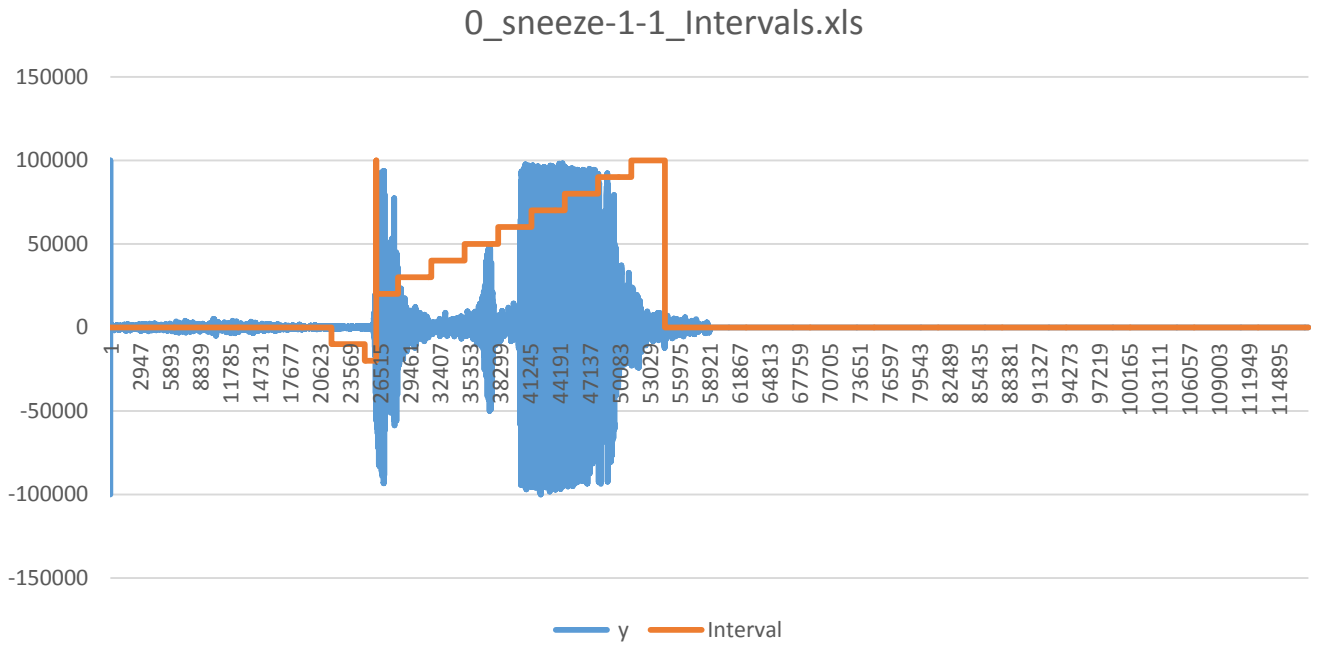
Classic CBR Overview

- New Problem: sound sample
- Case Library:
  - Confirmed sneeses (50)
  - Confirmed none-sneezes (50)
- Proposed Solution:
  - Proposed Sneeze
  - Proposed None-Sneeze



Extract Features from Sound Sample

- .wav-files PCM, 44.1 kHz, 16 bits, 1 or 2 channels (mono/stereo)
- Stereo -> Mono
- Normalize to 100000
- Analyse
  - Trigger On
  - Trigger Off
  - Split into intervals



Feature types per Interval

- Peak values
- Average values
- RMS values (effect)
- Peak to peak values
- CF - Crest Factor values
- PZ - Passage through Zero values
- FFT values (Fast Fourier Transform)
  - Sample length 2^{12}
  - Sample length 2^{14}
  - Sample length 2^{16}

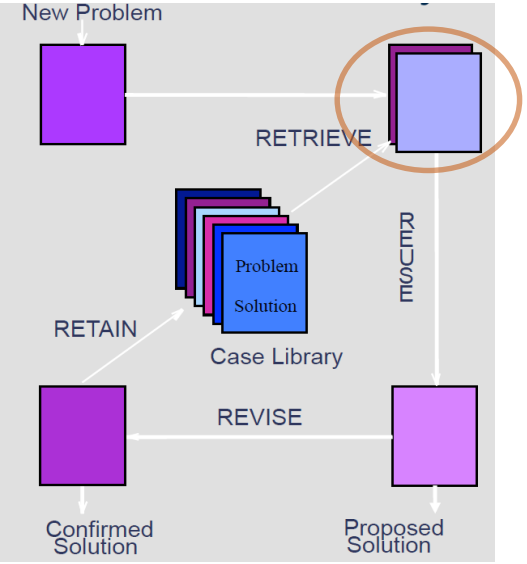
$$x_{\mathrm{rms}} = \sqrt{\frac{1}{n}\left(x_1^2 + x_2^2 + \cdots + x_n^2\right)}.$$

$$C = \frac{|x|_{\mathrm{peak}}}{x_{\mathrm{rms}}}.$$

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i 2\pi k \frac{n}{N}} \qquad k = 0, \ldots, N-1.$$

Feature values normalized within intervals (0.0 … 1.0)

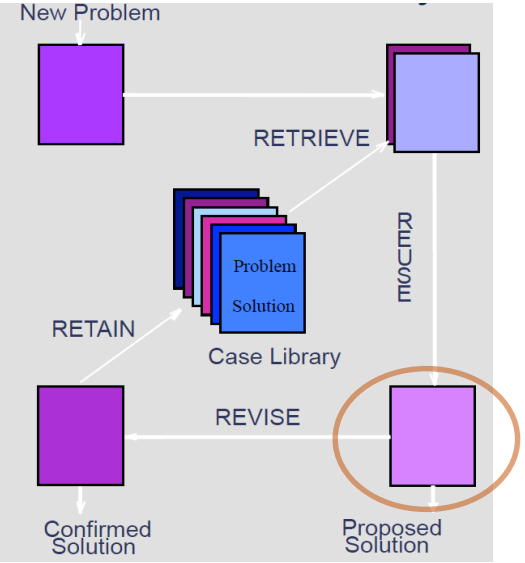
CBR System



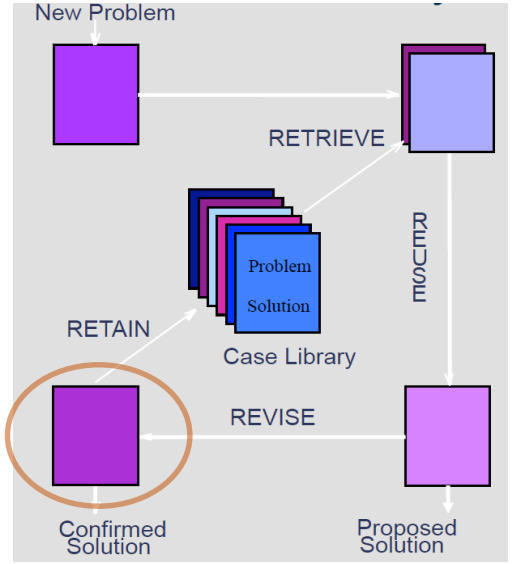
- Retrieve (find best SF value cases)
  - Compare new problem (*N*) to case library cases (*R*).
  - Calculate Similarity Function (*SF*) over all features and intervals.
    - $d_k(N,R) = \sum_{i=0}^{n} | n_i - r_i |$ *i* = interval
    - $f_k(N,R) = \frac{1}{1+d_k(N,R)}$ (0.0 … 1.0)
    - $SF(N,R) = \sum_{k=0}^{n} w_k * f_k(N,R)$ *k* = feature type, w_k = feature type weight
    - $\sum_{k=0}^{n} w_k = 1.0$

CBR System

- Reuse (generate proposed solution)
  - Pick 5 best cases, highest Similarity Function
  - Use majority vote to decide if proposed sneeze/none-sneeze



CBR System - Maintenance



- Revise
  - Add a new known case => 101 cases in library
  - Iterate each of 101 cases in case library while calculating SF.
    - Obtain worst case
      - Prio 1: case that have participated in voting and voted wrong every time
      - Prio 2: case that never participated in voting and has lowest SF value
      - Prio 3: case that has the lowest SF value
- Retain
  - Retain the best cases while preserving sneeze/none-sneeze ratio, i.e. remove the worst => back to 100 cases in library.

Build Environment

- Visual Studio
- C#

Demonstration

Ta-da!

What we learned

- Github (Conflicts!)
- Sound file structure
- FFT (Fast Fourier transform)
- CBR Systems
  - Distance calculation
  - Similary functions

The End

Thanks for listening!