Automatic Segmentation of Spontaneous Data using Dimensional Labels from Multiple Coders

Mihalis A. Nicolaou[*], **Hatice Gunes**[*,‡] **and Maja Pantic**[*,†]

[*]Department of Computing, Imperial College London, U.K.
(michael.nicolaou08, h.gunes,m.pantic)@imperial.ac.uk

[‡] School of Computing and Communications, University of Technology Sydney, Australia
[†]Faculty of EEMCS, University of Twente, The Netherlands

Abstract

This paper focuses on automatic segmentation of spontaneous data using continuous dimensional labels from multiple coders. It introduces efficient algorithms to the aim of (i) producing ground-truth by maximizing inter-coder agreement, (ii) eliciting the frames or samples that capture the transition *to* and *from* an emotional state, and (iii) automatic segmentation of spontaneous audio-visual data to be used by machine learning techniques that cannot handle unsegmented sequences. As a proof of concept, the algorithms introduced are tested using data annotated in arousal and valence space. However, they can be straightforwardly applied to data annotated in other continuous emotional spaces, such as power and expectation.

1. Introduction

In everyday interactions people exhibit non-basic, subtle and rather complex mental or affective states like thinking, embarrassment or depression (Baron-Cohen and Tead, 2003). Accordingly, a single label (or any small number of discrete classes) may not reflect the complexity of the affective state conveyed by such rich sources of information (Russell, 1980). Hence, a number of researchers advocate the use of dimensional description of human affect, where an affective state is characterized in terms of a number of (continuous) latent dimensions (Russell, 1980),(Scherer, 2000).

Spontaneous data and their dimensional annotations, provided by multiple coders, pose a number of challenges to the field of automatic affect sensing and recognition (Gunes and Pantic, 2010). The first challenge is known as *reliability of ground truth*. In other words, achieving agreement amongst the coders that provide annotations in a dimensional space is very challenging (Zeng et al., 2009). In order to make use of the manual annotations for automatic recognition, most researches take the mean of the coders ratings, or assess the annotations manually. How to best model inter-coder agreement levels for automatic affect analyzers remain mainly unexplored. The second challenge is known as *the baseline problem*: having "a condition to compare against" in order for the automatic recognizer to successfully learn the recognition problem at hand (Gunes and Pantic, 2010). Automatic affect analyzers relying on audio modality obtain such a baseline by segmenting their data based on speaker turns (e.g., (Wollmer, M. and Eyben, F. and Reiter, S. and Schuller, B. and Cox, C. and Douglas-Cowie, E. and Cowie, R., 2008)). For the visual modality the aim is to find a frame in which the subject is expressionless and against which changes in subject's motion, pose, and appearance can be compared. This is usually achieved by constraining the recordings to have the first frame containing a neutral expression. Although expecting expressionless state in spontaneous multicue or multimodal data is a strong and unrealistic constrain, automatic affect analysers *depend* on the existence of such a baseline state (e.g., (Petridis et al., 2009; Gunes and Piccardi, 2009)). Moreover, a number of machine learning techniques such as (coupled) Hidden Markov Models and Hidden-state Conditional Random Fields cannot handle unsegmented sequences, they require the data to have a class label for the entire sequence. To date, many automatic affect recognizers using audio-visual data and utilizing the aforementioned techniques segment their data manually (e.g., (Petridis et al., 2009)).

This paper provides solutions to all of the aforementioned issues. It (i) produces ground-truth by maximizing inter-coder agreement, (ii) elicits the frames or samples that capture the transition *to* and *from* an emotional state (a baseline condition to compare against), and (iii) automatically segments long sequences of spontaneous audio-visual data to be used by machine learning techniques that cannot handle unsegmented sequences.

2. Data

As a proof of concept, the algorithms introduced are tested using data annotated in arousal (how excited or apathetic the emotion is) and valence (how positive or negative the emotion is) space to obtain sequences that contain either positive or negative emotional displays. We use the Sensitive Artificial Listener Database (SAL-DB) (Cowie et al., 2005; Douglas-Cowie et al., 2007) and the SEMAINE Database (SEMAINE-DB) [1] that contain audio-visual spontaneous expressions.

2.1. Data Sets and Annotations

Both for the SAL-DB and the SEMAINE-DB, spontaneous data was collected to the aim of capturing the audio-visual

[1]The Semaine Database: http://semaine-db.eu/

interaction between a human and an avatar with four personalities: Poppy (happy), Obadiah (gloomy), Spike (angry) and Prudence (pragmatic).

The SAL data has been annotated by a set of coders who provided continuous annotations with respect to valence and arousal dimensions using the FeelTrace annotation tool (Cowie et al., 2000; Cowie et al., 2005). Feeltrace allows coders to watch the audio-visual recordings and move their cursor, within the 2-dimensional emotion space (valence and arousal) confined to $[-1, 1]$, to rate their impression about the emotional state of the subject.

For SAL-DB, 27 sessions (audio-visual recordings) from 4 subjects have been annotated. 23 of these sessions were annotated by 4 coders, while the remaining 3 sessions were annotated by 3 coders. The SEMAINE-DB has also been annotated using FeelTrace along five emotional dimensions (valence, arousal, power, expectation and intensity) separately, by (up to) 4 coders.

2.2. Challenges

The time-based operation of Feeltrace presents us with the following challenges: (i) for the sessions coded, there is no one-to-one correspondence between the timestamps of each coder, (ii) throughout the annotation files, there are time intervals where annotations are not available, and (iii) annotations are not (always) synchronized with the audio-visual data stream.

We tackle the first issue by binning the annotations: annotations that correspond to one video frame are grouped together. The second point refers to missing annotations for some sets of frames. This could potentially be due to the following reasons: (i) the coder might not be certain about the annotation for that particular interval, (ii) the coder might release the mouse button for some other reason, (iii) the coders appear to stop annotating when the avatar is talking, and (iv) the CPU load may have an effect on the frequency of measurements being recorded. Finally, the third issue could possibly be due to the following: (i) the response time is expression dependent, i.e., positive expressions are perceived faster and more accurately than negative ones (Alves et al., 2008), and (ii) the lag caused by the CPU load may have an effect on the synchronization between the actual video played and the recording of the annotations.

Table 1: The inter-coder MSE after applying local normalisation procedures: normalizing to a standard deviation of one and a zero mean (GD), normalizing to zero mean (ZA) and no normalisation (NN).

	ZA$_{MSE}$	**GD**$_{MSE}$	**NN**$_{MSE}$
Valence	0.046	0.93	0.072
Arousal	0.0551	0.9873	0.0829

3. Methodology

In this section we address the challenges identified when working with databases annotated in continuous dimensional spaces.

Algorithm 1: Binning the annotations of the coders {set of bins, b} $\leftarrow Binning()$

```
1  //all members of any structures are considered to be zero
2  for each coder file c in the annotation files set do
3      for each annotation a in a coder file c with a timestamp of t do
4          Determine bin b where t ∈ b
5          b.val ← b.val + a.val
6          b.arsl ← b.arsl + a.arsl
7          b.annotCount ← b.annotCount + 1
8      end
9      for all bins b in the set of bins do
10         Average b.val and b.arsl by dividing with b.annotCount
11     end
12 end
```

Algorithm 2: Detecting crossovers in coder annotations: $\{PosCrossOver, NegCrossOver\} \leftarrow$ DetectCrossovers(coder c)

```
1  //bstr is the binned structure, every member is an annotation of A-V values at that
   frame by the specific coder
2  for each f in bstr do
3      if sign(bstr(f).val) ≠ sign(bstr(f − 1).val) then
4          if sign(bstr(f).val) > 0 then
5              Add f to PosCrossOver structure
6          end
7          else
8              if sign(bstr(f).val) < 0 then
9                  Add f to NegCrossOver structure
10             end
11         end
12     end
13 end
```

3.1. Annotation Pre-processing

This process involves determining normalisation procedures and extracting statistics from the data in order to obtain segments with a baseline and high inter-coder agreement.

Binning. Binning refers to grouping and storing the annotations together. As a first step the measurements of each coder c are binned separately. Since we aim at segmenting video files, we generate bins which are equivalent to one video frame f. This is equivalent to a bin of 0.04 seconds (SAL-DB was recorded at a rate of 25 frames/s). The basic binning procedure is illustrated in Algorithm 1. The fields with no annotation are assigned a "not a number" (NaN) identifier.

Normalisation. The arousal and valence (A-V) measurements for each coder are not in total agreement, mostly due to the variance in human coders' perception and interpretation of emotional expressions. Thus, in order to deem the annotations comparable, we need to normalize the data. Similar procedures have been adopted by other works using SAL-DB (e.g. (Wollmer, M. and Eyben, F. and Reiter, S. and Schuller, B. and Cox, C. and Douglas-Cowie, E. and Cowie, R., 2008)).

We experimented with various normalisation techniques. After extracting the videos and inspecting the superimposed ground truth plots, we opted for local normalisation (normalizing each coder file for each session). This helps us avoid propagating noise in cases where one of the coders is in large disagreement with the rest (where a coder has a very low correlation with respect to the rest of the coders). As can be seen from Table 1, locally normalizing to zero mean produces the smallest mean squared error (MSE) both

for valence (0.046) and arousal (0.0551) dimensions. Varying the standard deviation results in values which are outside the range of $[-1, 1]$ and generates more disagreement between coders.

Statistics and Metrics. We extract two useful statistics from the annotations, with a motivation of using them as measures of agreement amongst the annotations provided: correlation (COR) and sign-agreement (SAGR). We start the analysis by constructing vectors of pairs of coders that correspond to each video session, e.g., when we have a video session where four coders have provided annotations, this gives rise to six pairs. For each of these pairs we extract the correlation coefficient between the valence (val) values of each pair, as well as the percentage of sign-agreement in the valence values, which stands for the level of agreement in emotion classification in terms of positive or negative:

$$SAGR(c_i, c_j) = \frac{\sum_{f=0}^{|frames|} e(c_i(f).val, c_j(f).val)}{|frames|} \quad (1)$$

where c_i and c_j represent the pair of coders the sign-agreement metric is calculated for, and $c_i(f).val$ stands for the valence value annotated by coder c_i at frame f. Function e is defined as:

$$e(i, j) = \begin{cases} 1 & if(sign(i) = sign(j)) \\ 0 & else \end{cases}$$

The sign-agreement metric is of high importance for the valence dimension as it determines whether the coders agree on the classification of the emotional state as positive or negative. More specifically, such metrics provide information regarding the perception and annotation behaviour of the coders (i.e., to what degree data is annotated similarly by different coders). In these calculations we do not consider the NaN values to avoid negatively affecting the results.

After these metrics (agreement, correlation) are calculated for each pair, each coder is assigned the average of the results of all pairs that the coder has participated in. In other words, the averaged metric m'_{S,c_j} with respect to coder c_j for a specific metric m (i.e., correlation or agreement) is defined as follows:

$$m'_{S,c_j} = \frac{1}{|S| - 1} \sum_{i \in S, c_i \neq c_j} m(c_i, c_j) \quad (2)$$

where S is the relevant session annotated by $|S|$ number of coders, and each coder annotating S is defined as $c_i \in S$. Essentially, we calculate the averaged level of agreement of coder c_j with respect to the rest, by using the metric m. This is somewhat equivalent to the enumerator of the modified Williams Index, which would be obtained by dividing this enumerator by the averaged level of agreement of all the coders except c_j (Alberola-Lopez et al., 2004). Instead, we obtain the weighted average by using the m' as weights, as shown in line 28 of Algorithm 4. The automatic segmentation process is based on the correlation metric (cor') alone as correlation experimentally proved stricter than sign-agreement in providing better comparison between the coders.

Interpolation. In order to deal with the issue of missing values, similar to other works reporting on data annotated in continuous dimensional spaces (e.g., (Wollmer, M. and Eyben, F. and Reiter, S. and Schuller, B. and Cox, C. and Douglas-Cowie, E. and Cowie, R., 2008)), we interpolated the actual annotations at hand. We used piecewise cubic interpolation as it preserves the monotonicity and the shape of the data.

Algorithm 3: Match crossovers across coders for each session, maximizing the number of coders participating: $\{MatchedCO\} \leftarrow$ MatchCrossOvers($CrossOvers$)

```
1  for Each session s do
2      for i=4 to 2 do
3          //get as many coders as possible to agree (max. 4 and min. 2)
4          for Each crossover co in CrossOvers belonging to s do
5              currentlyMatched ← {co}
6              Find all crossovers co2 in CrossOvers which:
7              - Belong to s
8              - Are from different coders
9              - co2 ≠ co ∧ abs(co2.time − co.time) ≤ 0.5 seconds
10             Add the co2 to currentlyMatched
11             if length(currentlyMatched) = i then
12                 mark all crossovers in currentlyMatched as seen
13                 add currentlyMatched to MatchedCO
14                 remove currentlyMatched from CrossOvers
                   belonging to s
15             end
16         end
17     end
18 end
```

3.2. Automatic Segmentation

The automatic segmentation stage consists of producing negative and positive audio-visual segments with a temporal window that contains an offset before and after (i.e., the baseline) the displayed expression. This process is presented in Algorithm 4 that makes use of Algorithms 2 and 3.

Firstly, we describe the actual time window that the audio-visual segment is supposed to capture. For instance, for capturing negative emotional states, if we assume that the transition *from* non-negative *to* a negative emotional state occurs at time t (in seconds), we then have a window of $[t - 1, t, t', t' + 1]$ where t' seconds is when the emotional state of the subject returns to non-negative. The procedure is analogous for positive emotional states.

Detecting and Matching Crossovers. In Algorithm 2, for an input coder c, the crossing over from one emotional state to the other is detected by examining the valence values and identifying the points where the sign changes. Here a modified version of the sign function is used which returns 1 for values ≥ 0 (a value of 0 valence is never encountered in the annotations), -1 for negative, and 0 for NaN values. Algorithm 2 accumulates all crossover points for each coder, and returns the set of crossovers *to-a-positive* ($PosCrossOver$) and *to-a-negative* ($NegCrossOver$) emotional state. The output is then passed to Algorithm 3.

The goal of Algorithm 3 is to match crossovers across coders. For instance, if a session has annotations from 4 coders, due to synchronization issues discussed previously, the frame (f) where each coder detects the crossover is not the same for all coders (for the session in question). Thus, we have to allow an offset for the matching process.

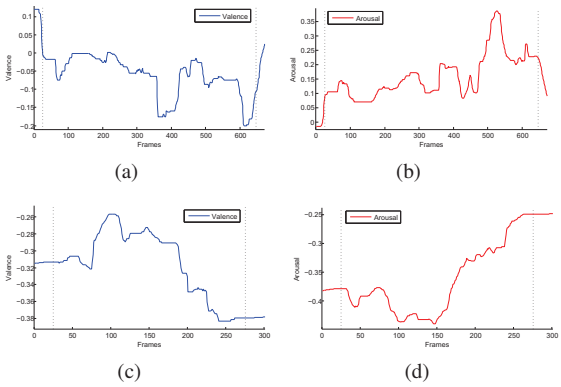

Figure 1: Two examples of interpolated valence ((a),(c)) and arousal ((b),(d)) plots from two individual segments produced by the segmentation procedure.

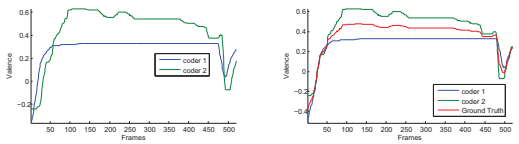

Figure 2: Valence annotations from two coders in SEMAINE-DB before and after applying pre-processing operations.

This procedure searches the crossovers detected by the coders and then accepts the matches where there is less than the pre-defined offset (time) difference between them. When a match is found, we remove the matched crossovers and continue with the rest. The existence of different combinations of crossovers which may match using the predefined offset poses an issue. By examining the available datasets, we decided to maximize the number of coders participating in a matched crossover set rather than minimizing the temporal distances between the participating coders. The motivations for this decision are as follows: (i) if more coders agree on the crossover, the reliability of the ground truth produced will be higher, and (ii) the offset amongst the resulting matches is on average quite small (less than 0.5 secs) when considering only the number of participating coders. Maximising the number of participating coders can simply be achieved by iterating over the entire set of crossovers. This is expressed by the loop beginning in line 2 of Algorithm 3. We disregard cases where only one coder detects a crossover due to lack of agreement between coders.

Segmentation Driven by Matched Crossovers : This procedure (illustrated in Algorithm 4) takes the output of Algorithm 3, and attains the sets of matched crossovers (Algorithm 3, lines 6-7). An iteration for all sets of matched crossovers for *to-Negative* transition is shown starting in line 7. $mcos$, $mcos(i).f$ and $mcos(i).c$ represent the current matched crossover, the frame where the i-th crossover (of the matched crossover) occurred, and the coder who detected the i-th crossover of $mcos$, respectively. $mcos(i).val$ is the vector of valence measurements for coder i participating in $mcos$. The crossover frame decision (for each member of the set) is made in lines 10:17, and the *start frame* of the video segment is decided. In order to capture 1 second before the transition window, the number of frames corresponding to the pre-defined offset are subtracted from the *start frame*. The ground truth values for valence are retrieved in lines 19:30 by incrementing the initial frame number where each crossover was detected by the coders. The procedure of determining combined average values continues until the valence value crosses again to a *non-negative* valence value. The endpoint of the audio-visual segment is then set to the frame including the offset after crossing back to a *non-negative* valence value.

The ground truth of the audio-visual segment consists of the arousal and valence (A-V) values described in lines 24 and 28 of the algorithm. If only two coders agree in the detection of crossovers, their contribution is weighted by using the correlation metric (cor', calculated as described in Equation 2).

Figure 3: Example frames from an automatically extracted segment from SEMAINE-DB capturing the transition from a negative to a positive emotional state and back.

4. Experiments and Results

As a proof of concept, the algorithms introduced have been extensively tested on SAL-DB.

We first present in Fig. 1 two segments extracted by using Algorithm 4, for a transition *to* a negative emotional state. The first dashed vertical line represents the transition *to* that state, and the second one *out of* that state. In the plots, we present the A-V values after the interpolation. Thus, at times no crossover may be observed in the valence values.

As performance evaluation is a significant issue for any automatic system, in Table 2 we attempt to provide meaningful performance results of the introduced algorithms on SAL-DB. The table presents the performance of the automatic audio-visual segmentation procedure in terms of: (i) how well it is able to utilise the actual number of frames (*# of frames*), (ii) using the given data, how many audio-visual segments it is able to produce (*# of segments*), and (iii) how much overlap there is (*overlap*) between the segments, and between the positive and negative classes. The goal of the automatic segmentation procedure is then to utilise as many frames as possible from the given data to produce a high number of meaningful segments. Too much overlap between the segments or between the classes is un-

Table 2: Evaluation of the introduced segmentation algorithms using SAL-DB. The table presents the actual number of frames together with the utilised number of frames (*# of frames*), the number of audio-visual segments produced (*# of segments*) using the data at hand, and the intra-class (percentage of frames included in more than one segment within the same class) and inter-class (percentage of frames included in both classes) overlap.

	subject #	1	2	3	4
total	**# of frames**	56162	80553	28583	88199
negative	**# of frames**	27389	46056	14554	43353
	# of segments	110	170	99	166
	intra-class overlap	6.42%	8.33%	4.53%	7.70%
positive	**# of frames**	23831	36034	13584	38589
	# of segments	110	149	91	174
	intra-class overlap	18.90%	14.18%	10.22%	11.60%
	inter-class overlap	6.16%	7.39%	14.37%	9.92%

Algorithm 4: Segment and produce ground truth: Segmentation()

1 **for** *each coder annotation file c* **do**
2 //capture a transition to and from a neg. state to a non-neg.
3 // use the correlation (cor') for weighting when match has 2 coders
4 $\{PosCrossOver, NegCrossOver\} \leftarrow DetectCrossovers(c)$
5 $MatchedPos \leftarrow MatchCrossOvers(PosCrossOver)$
6 $MatchedNeg \leftarrow MatchCrossOvers(NegCrossOver)$
7 **for** *each matched set of crossovers* $mcos$ *in* $MatchedNeg$ **do**
8 //average time (frame) of crossing over to negative valence
9 //0.5 second offset has been used
10 **if** *length(*$mcos$*)* $\geq$ *3* **then**
11 //agreement in 3 or 4 coders
avgFrm = $int\left(\frac{\sum_{i=0}^{|mcos|} mcos(i).f}{length(mcos)}\right)$
12
13 **end**
14 **else**
15 //2 coders agree, weight using correlation (cor')
avgFrm = $int\left(\frac{\sum_{i=0}^{|mcos|}(mcos(i).f * cor'(mcos(i).c))}{\sum_{i=0}^{|mcos|} cor'(mcos(i).c)}\right)$
16
17 **end**
18 $startFrm = avgFrm - 25$
19 incFrm $\leftarrow$ 0
20 **repeat**
21 incFrm $\leftarrow$ incFrm + 1
22 **if** *length(*$mcos$*)* $\geq$ *3* **then**
23 //agreement in 3 or 4 coders
24 avgValence =
$\frac{\sum_{i=0}^{|mcos|} mcos(i).val(mcos(i).f+incFrm)}{length(mcos)}$
25 **end**
26 **else**
27 //2 coders agree, weight using cor'
28 avgValence =
$\frac{\sum_{i=0}^{|mcos|}(mcos(i).val(mcos(i).f+incFrm) * cor'(mcos(i).c))}{\sum_{i=0}^{|mcos|} cor'(mcos(i).c)}$
29 **end**
30 **until** *sign(avgValence)=1 or avgValence is* NaN ;
31 //add offset after crossing back to non-negative (or NaN)
32 endFrm = (avgFrm + incFrm) + 25
33 //Video is segmented in the range [startFrm,endFrm]
34 //Ground truth (valence/arousal) is averaged
35 **end**
36 //the process is repeated analogously for "to-Positive" crossovers ($MatchedPos$) - line 7
37 **end**

intended and undesirable, but expected to some degree due to the offsets before and after the transitions. By observing Table 2 we conclude that the algorithm fulfills its goal.

As a final step we test the developed algorithms on the recently released SEMAINE-DB. Although the arousal and valence annotations of SEMAINE-DB do not contain NaN values, the steps to be followed for segmentation are similar.

Finally, a qualitative assessment of the proposed algorithms is provided by Fig. 2 and Fig. 3. Fig. 2 illustrates valence annotations from two coders in SEMAINE-DB before and after applying pre-processing operations (for synchronization). Fig. 3 shows example frames from an automatically extracted segment from SEMAINE-DB using the presented algorithms. Overall, the produced segment appears to well capture the transition from a negative emotional state to a positive one, and back.

5. Conclusion

This paper introduced efficient algorithms to the aim of (i) producing ground-truth by maximizing inter-coder agreement, (ii) eliciting the frames that capture the transition *to* and *from* an emotional state, and (iii) automatic segmentation of spontaneous multimodal data to be used by machine learning techniques that cannot handle unsegmented sequences. As a proof of concept, the algorithms introduced have been tested using SAL and SEMAINE data annotated in arousal and valence spaces. Overall, the automatic segmentation procedures introduced appear to work as desired and output segments that well capture the targeted emotional transitions.

6. Acknowledgments

The work of Mihalis A. Nicolaou and Maja Pantic is funded by the European Research Council under the ERC Starting Grant agreement no. ERC-2007-StG-203143 (MAHNOB). The work of Hatice Gunes is funded by the European Community's 7th Framework Programme [FP7/2007-2013] under the grant agreement no 211486 (SEMAINE).

7. References

C. Alberola-Lopez, M. Martin-Fernandez, and J. Ruiz-Alzola. 2004. Comments on: A methodology for evaluation of boundary detection algorithms on medical images. *IEEE Transactions on Medical Imaging*, 23(5):658–660, May.

N. T. Alves, J. A. Aznar-Casanova, and S. S. Fukusima. 2008. Patterns of brain asymmetry in the perception of positive and negative facial expressions. *Laterality: Asymmetries of Body, Brain and Cognition*, 14:256–272.

S. Baron-Cohen and T. H. E. Tead. 2003. *Mind reading: The interactive guide to emotion*. Jessica Kingsley Publishers Ltd.

R. Cowie, E. Douglas-Cowie, S. Savvidou, E. McMahon, M. Sawey, and M. Schroder. 2000. Feeltrace: An instrument for recording perceived emotion in real time. In *Proc. of ISCA Workshop on Speech and Emotion*, pages 19–24.

R. Cowie, E. Douglas-Cowie, and C. Cox. 2005. Beyond emotion archetypes: Databases for emotion modelling using neural networks. *Neural Networks*, 18:371–388.

E. Douglas-Cowie, R. Cowie, I. Sneddon, C. Cox, L. Lowry, M. McRorie, L. Jean-Claude Martin, J.-C. Devillers, A. Abrilian, S. Batliner, A. Noam, and K. Karpouzis. 2007. The humaine database: addressing the needs of the affective computing community. In *Proc. of Int'l Conf. on Affective Computing and Intelligent Interaction*, pages 488–500.

H. Gunes and M. Pantic. 2010. Automatic, dimensional and continuous emotion recognition. *International Journal of Synthetic Emotions*, 1(1):68–99.

H. Gunes and M. Piccardi. 2009. Automatic temporal segment detection and affect recognition from face and body display. *IEEE Tran. on Systems, Man, and Cybernetics-Part B*, 39(1):64–84.

S. Petridis, H. Gunes, S. Kaltwang, and M. Pantic. 2009. Static vs. dynamic modeling of human nonverbal behavior from multiple cues and modalities. In *Proc. of ACM Int'l Conf. on Multimodal Interfaces*, pages 23–30.

J. A. Russell. 1980. A circumplex model of affect. *Journal of Personality and Social Psychology*, 39:1161–1178.

K.R. Scherer, 2000. *The Neuropsychology of Emotion*, chapter Psychological models of emotion, pages 137–162. Oxford University Press.

Wollmer, M. and Eyben, F. and Reiter, S. and Schuller, B. and Cox, C. and Douglas-Cowie, E. and Cowie, R. 2008. Abandoning emotion classes - towards continuous emotion recognition with modelling of long-range dependencies. In *Proc. of 9th Interspeech Conf.*, pages 597–600.

Z. Zeng, M. Pantic, G.I. Roisman, and T.S. Huang. 2009. A survey of affect recognition methods: Audio, visual, and spontaneous expressions. *IEEE Tran. on Pattern Analysis and Machine Intelligence*, 31:39–58.