

Zagreb, 9. ožujka 2018.

DIPLOMSKI ZADATAK br. 1699

Pristupnik: **Denis Malić (0036472502)**
Studij: Elektrotehnika i informacijska tehnologija
Profil: Elektroničko i računalno inženjerstvo

Zadatak: **Afektivno računarstvo na ugradbenim računalima**

Opis zadatka:

U okviru diplomskog rada potrebno je istražiti mogućnost izvedbe sustava afektivnog računarstva pomoću ugradbenog računala. Afektivno računarstvo obuhvaća raznovrsne postupke umjetne inteligencije ali u kontekstu ljudskih emocija, njihovog prepoznavanja, interpretacije, obrade i simulacije. Kao ulazni modalitet potrebno je koristiti govorni signal te na pogodno odabranim zadacima afektivnog računarstva, demonstrirati mogućnost realizacije sustava ugradbenim računalom. Zadaci mogu biti estimacija emocionalnog stanja u ravni ugodna-pobuđenost, estimacija primarnih emocija (neutralno, sreća, tuga, strah, ljutnja), estimacija razine stresa, ili estimacija kognitivnog napora. Dio kojeg je potrebno implementirati i evaluirati na ugradbenom računalu, odnosi se samo na izvedbu klasifikatora, ili estimatora, dok se gradnja sustava (estimacija parametara statističkih modela) može provesti unaprijed korištenjem standardiziranih baza prilagođenih pojedinom zadatku. Posebnu pažnju posvetiti na mogućnost rada u stvarnom vremenu, na memorijske i procesne zahtjeve algoritma, te diskutirati arhitekture klasifikatora i njihove parametre koji to omogućuju. Kao platformu za demonstraciju i evaluaciju brzine rada i točnosti, potrebno je koristiti ugradbeno računalo Raspberry PI.

Zadatak uručen pristupniku: 16. ožujka 2018.

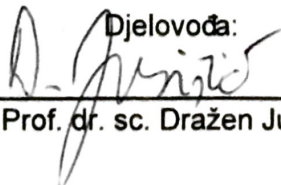
Rok za predaju rada: 29. lipnja 2018.

Mentor:




Prof. dr. sc. Davor Petrinović

Djelovođa:



Prof. dr. sc. Dražen Jurišić

Predsjednik odbora za
diplomski rad profila:



Prof. dr. sc. Mladen Vučić

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 4531

**AFEKTIVNO RAČUNARSTVO NA
UGRADBENIM RAČUNALIMA**

Denis Malić

Zagreb, lipanj 2018.

UNIVERSITY OF ZAGREB
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

MASTER'S THESIS no. 4531

**AFFECTIVE COMPUTING ON EMBEDDED
PLATFORMS**

Denis Malić

Zagreb, June 2018.

ACKNOWLEDGMENTS:

Puno hvala mentoru prof.dr.sc. Davoru Petrinoviću na pomoći i uloženom vremenu. Također, hvala asistentu Igoru Mijiću koji mi je puno pomogao kroz razgovor i savijete.

A massive thank you to my mentor, Professor Davor Petrinović, for his guidance, advice and willingness to help me. Also, many thanks to Igor Mijić, an assistant without whom I would not have grasped machine learning mechanisms.

Contents

1.	Introduction	1
1.1.	Ambiguity of emotional expression.....	1
1.2.	History of machine learning	2
2.	Emotion annotation schemes	4
2.1.	Categorical labelling.....	4
2.2.	Dimensional approach	4
2.3.	Approach used in this project	5
3.	Project outline	7
4.	Features	10
4.1.	Functionals	15
5.	Database	17
6.	Python functions and libraries	21
6.1	Mathematical background.....	23
7.	Database preparation	31
8.	Accuracy.....	35
9.	Execution time comparison on Raspberry Pi vs. PC	39
8.1.	Raspberry Pi implementation	40
10.	Raspberry PI.....	41
11.	Conclusion and final remarks	43
12.	Summary	44
13.	Sažetak	45
	Bibliography	46

TABLE OF FIGURES:

FIGURE 1 THE 2 PI RADIAN RANGE SEGMENTED INTO EIGHT OCTANTS. DIGITAL IMAGE.
APPROXIMATIONS FOR THE ARCTANGENT FUNCTION IN EFFICIENT FRINGE PATTERN
ANALYSIS, 2007, [HTTPS://WWW.OSAPUBLISHING.ORG/OE/FULLTEXT.CFM?URI=OE-15-6-3053&ID=131164](https://www.osapublishing.org/oe/fulltext.cfm?uri=oe-15-6-3053&id=131164) 6

FIGURE 2 ILLUSTRATION OF A TWO-DIMENSIONAL PLANE TOGETHER WITH THE
ALTERNATIVE THREE-DIMENSIONAL REPRESENTATION. ON THE RIGHT IT IS VISIBLE
THAT NOT ALL QUADRANTS ARE EQUALLY DOTTED WITH TYPICAL EMOTIONS. THIS
FIGURE IS TAKEN FROM [1]. 6

FIGURE 3 THE RECORDINGS WERE ANNOTATED USING THE ANNEMO WEB-BASE
ANNOTATION TOOL. THIS FIGURE IS TAKEN FROM [12]. 18

FIGURE 4 THIS FIGURE REPRESENTS VALENCE ANNOTATIONS WITH ZERO MEAN
NORMALISATION FROM SEVEN ANNOTATORS; THREE MALE, THREE FEMALE AND
ONE NON-NATIVE ANNOTATOR. THIS FIGURE IS TAKEN FROM [12]. 19

FIGURE 5 USED PYTHON PACKAGES 22

FIGURE 6 USED PYTHON PACKAGES ON RASPBERRY PI 23

FIGURE 7 THIS FIGURE REPRESENTS THE MARGIN ZONE WITH DATAPOINTS BELONGING TO
TWO CLASSES THAT ON ITS BOUNDARIES HAVE SUPPORT VECTORS (CIRCLED). 25

FIGURE 8 THIS FIGURE SURMISES WHAT IS WRITTEN IN THE TEXT REGARDING THE
DEFINITIONS OF THE SEPARATING, POSITIVE AND NEGATIVE (MARGIN)
HYPERPLANES. 26

FIGURE 9 29

FIGURE 10 THE RASPBERRY PI 3 MODEL B+ 41

1. INTRODUCTION

Machine learning was defined as “the field of study that gives computers the ability to learn without being explicitly programmed” by Arthur Samuel who also coined its name. Algorithms able to overcome following strictly static program instructions by making data-driven predictions have its roots in pattern recognition and computational learning theory in artificial intelligence. Those algorithms are commonly associated with machine learning. The main characteristic of the algorithms is building a model from sample inputs. Machine learning is, thus, a subset of artificial intelligence in the field of computer science that aims to progressively improve performance on a specific task with data, without being explicitly programmed. Division by which problems in the field are separated infers two major tasks: supervised and unsupervised learning. The former taking as input annotations together with training data, and the latter that focuses on exploratory data analysis and does not take annotations as input.

Affective computing research has gained momentum in the past few years, and although what seemed intractable mostly due to the amount of data associated with related tasks, has become an actuality. Boosted computing power of today’s computers has been one of the triggers that helped to shift theoretical knowledge to colossally important every-day applications. Another basis upon which the interest in the field has been sparked in many researchers are the practical implications, for example, in human-computer interactions, customer service furthering, and emergency call-centres prioritisation among callers that makes the service more effective and prompt. When interacting with a machine, humans often get impatient or frustrated if the machine responds inappropriately to how someone feels, for example, brief answers are not welcome when we are in doubt, and long-winded answers are irritating in other situations.

1.1. AMBIGUITY OF EMOTIONAL EXPRESSION

At the beginning it is important to point out to the complexity and multimodality of day-to-day human interactions where emotions are conveyed by means of language, vocal intonations, facial expressions, hand gesture, head movement, body movement and posture.

Despite of abundance of cues in human-human interaction, the mainstream thus far divulged course of research in the affective computing field has focused on vocal and facial expressions. Background research in various scientific fields such as neuroscience, psychology and linguistics is used as an affective computing development basis. Thus, progress in emotion sensing and recognition is closely related to the study of the aforementioned disciplines. In readers interest would be to consult materials that are part of the course called Digital Speech Processing conducted at Faculty of Electrical Engineering and Computing in Zagreb, which are helpful to understand speech related measurable features commonly exploited in human affect modelling. Familiarizing oneself with principles of speech formation is beneficial for selecting audio features used for a specific application. Further expansion in that regard will be provided in the following chapters.

Despite major advances in the field of affective computing research, modelling, analysing, interpreting and responding to naturalistic human affective behaviour remains a challenge for automated systems as emotions are ambiguous constructs to define even for humans. Even though all the so far proposed approaches provide an insight into the underlying principles of affect conveyance, all of them beg the question of real-life relevance, as different tactics of affect modelling are constantly pitched against each other in inconclusive papers from various journals. That is a proof of doubtfulness that researchers are still facing with, and it is a remainder of the pitfalls accompanying the subject. Emotional boundaries are in psychological circles still vague or ill-defined with uncertainty in individual expression and non-uniform perception among general population. Individual variations are substantial traits of affective displays, and with subtlety and complexity of emotional states conveyed in human interactions, cause emotive computing to be challenging.

1.2. HISTORY OF MACHINE LEARNING

It would be to readers disservice not to mention the history of machine learning and the course of the events that led to where we are today in 2018. Therefore, it will be flashed out how and why can we with such confidence claim to deal with, what is certainly not a trivial matter, with arguable success, and how we got there. It all started in the fifties when computers were still in little supply and rather weak which remained the case for half a

century making the reasons of marginalisation of the matter understandable. Throughout that era the subject was relegated to being mainly theoretical and rarely employed. It was in 1963 that the Support Vector Machine was created by Vladimir Vapnik in Soviet Union, only for it to be under the radar for three more decades until he was scooped by Americans to the Bell Labs in the nineties. The neural network was devised in the 1940's, but again the computers of that time were nowhere near powerful enough to run them well. More than half a century on has the shift happened in the approach to the mathematical theorems. Odd though it may sound to some, against the backdrop of archived knowledge, computers of the twenty-first century have spurred the significant progress in the field. As the subject was stirred in the wake of technological advancements, new techniques were conceived that made machine learning seem mature, nevertheless it is very useful to learn it today in 2018, since it can now be evaluated rather than just studied superficially.

2. EMOTION ANNOTATION SCHEMES

2.1. CATEGORICAL LABELLING

In this subsection, so far proposed psychological methodologies linked to the subject are going to be outlined. Seven discrete emotions: sadness, surprise, fear, anger, neutral, happiness and disgust form the mainstream classes of the categorical method. Categorical labelling stems from the theory of pure emotions in which a person is characterised by the ability to express one emotion at a time. Apart from the conventional categorical method, there are other classification methods whose purpose will be additionally discussed once the drawbacks of the categorical method are explained. As it happens, while expressing emotions we as humans exhibit non-basic, subtle and rather complex mental states like thinking, embarrassment or depression which cannot be expressed using the categorial method, instead they require alternative treatment. For example, varying intensities of emotions, presence of several emotions at a time, and restriction in terms of the number of descriptive classes are the downsides of the categorical approach.

2.2. DIMENSIONAL APPROACH

Having in mind the fuzziness of emotional boundaries, some researches have conceived that affect intensity evaluation necessitates continuous scale used for measurement. To that end, psychologists have come up with a technique that accounts for the categorical approach disadvantages. Emotions can also be represented on a continuous scale, or to be precise, on multiple scales in a two-dimensional plane or a three-dimensional space. Usually, emotions are two-dimensional objects in a plane of valence and arousal. Usage of the third dimension can be encountered while browsing through myriad of published papers, with its name not being equal throughout. In line with that and having contemplated possible advances of using the third dimension for the purposes of automatic affect sensing and recognition, it was concluded that striking improvements would not be obtained, if the third dimension was employed. Furthermore, usage of a database with annotations of that kind has not been granted to the author, so it remains open for investigation to determine what would be the gains of using the third dimension. In **Figure 2** commonly declared emotions are mapped

into a two-dimensional plane with the subfigure on the left representing an emotional space with three dimensions.

2.3. APPROACH USED IN THIS PROJECT

In this project dimensional emotional labels were mapped into a two-dimensional radially divided space separated to eight octants. Humans can only appreciate emotional labelling to a certain level, therefore, the procedure did not jeopardize the perceptual emotional categorisation. On balance, the minimal risks of simplifying emotion annotations by mapping them into eight octants is outweighed by significant computation time improvements. The improvements are obtained as a consequence of employing classification instead of regression for training and testing a classifier. Moreover, regression used in conjunction with dimensionally annotated data causes the need for implementation of two separate classifiers, hence, doubling the running time in the prediction step on an embedded platform.

There is yet another remark to make, it could be beneficial to introduce the neutral class upon further statistical analysis. The question that the analysis shall answer by examining the annotations from the Recola database is what would be the radius of the centrally positioned neutral area. The reason why averaging the annotations across time frames was not sufficient are their resulting near-zero values that ensue from annotations located near the centre in a particular time frame (other functionals were computed on top of the mean value as a solution). Accuracy, which is already satisfactory for emotion prediction, could be enhanced by accounting for all near-the-centre datapoints with little or no significance in emotion class prediction.

To recap, the approach used in this project is a mixture of the categorical and the dimensional approaches that aims to make the implemented algorithms suitable for computation time constrained applications without degrading perceptual emotion categorisation. In **Figure 1** 360 degrees range is segmented into eight octants; the octants have dimension signs written inside.

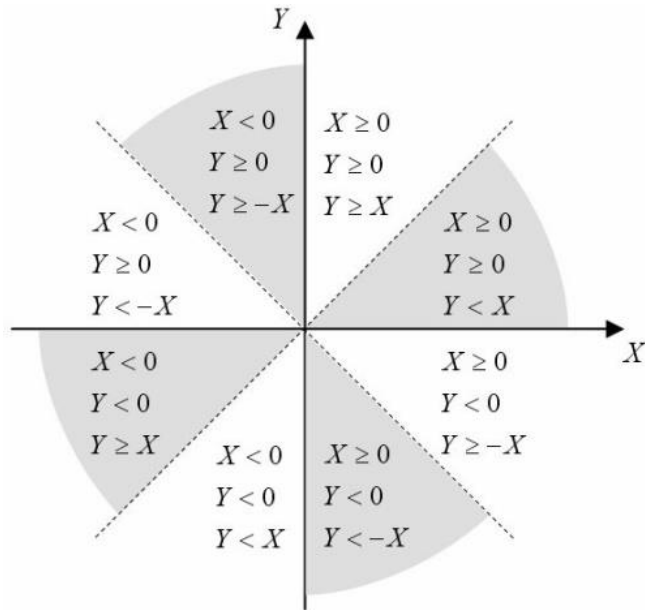


FIGURE 1 THE 2 PI RADIAN RANGE SEGMENTED INTO EIGHT OCTANTS. DIGITAL IMAGE. APPROXIMATIONS FOR THE ARCTANGENT FUNCTION IN EFFICIENT FRINGE PATTERN ANALYSIS, 2007, [HTTPS://WWW.OSAPUBLISHING.ORG/OE/FULLTEXT.CFM?URI=OE-15-6-3053&ID=131164](https://www.osapublishing.org/OE/FULLTEXT.CFM?URI=OE-15-6-3053&ID=131164)

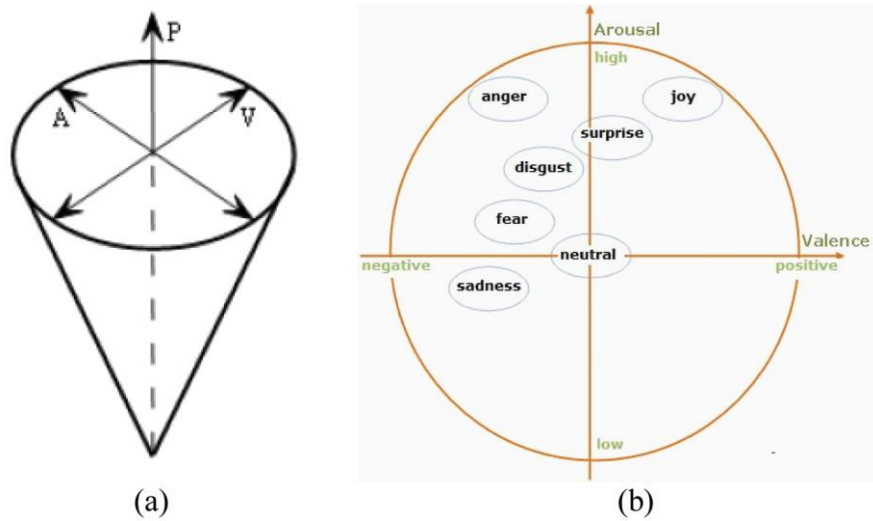


FIGURE 2 ILLUSTRATION OF A TWO-DIMENSIONAL PLANE TOGETHER WITH THE ALTERNATIVE THREE-DIMENSIONAL REPRESENTATION. ON THE RIGHT IT IS VISIBLE THAT NOT ALL QUADRANTS ARE EQUALLY DOTTED WITH TYPICAL EMOTIONS. THIS FIGURE IS TAKEN FROM [1].

3. PROJECT OUTLINE

In this project typical assignments such as supervised classifier learning, classifier testing and implementation on an embedded platform have been decided on as a domain of study. Noteworthy, related to the topic attainments will be hereby laid out. Since graduate projects which show grasp and technical literacy of a participant with the ability to deliver palpable results are considered successful, this thesis is aimed towards that goal. In other words, focus is not on outscoring current state-of-the-art achievements, but on illustrating a typical emotion recognition machine learning task. This project has its specificities on top of ordinarily exhibited machine learning problems, for instance, selection of optimal features in the feature extraction step, the annotations mapping, and the database preparation. Following steps intend to systematically explain machine learning mechanisms in a comprehensive manner.

It is of utmost importance to adopt the proper method for challenging peculiarities of emotion recognition, since approaching the problem varies greatly depending on the nature of the task. Innate differences of tasks are contingent on the type of data that is classified, i.e., whether data are emotions, share prices or something else. Accordingly, training and testing of data of dissimilar nature against the same classifier is not advisable. First decision pertaining emotion recognition should cope with whether to use regression or classification. The former would imply having discrete annotations, and the latter having continuously annotated data, usually in multiple dimensions. Furthermore, any experience could potentially save us a great deal of time spent on classifier's parameters tuning in case it has been proven that data is not annotated properly or if the wrong classifier is chosen for the task in question. Approach used for tackling the problem could be called prospective hindsight. To reiterate, when handling data and its annotations it is vital to be able to answer questions that aim to explain will the implemented procedure be useful for differentiating between emotions or is such a procedure solely satisfiable for the sake of obtaining seemingly content accuracy with no real-world applicability. Another way of looking at it is making sure that predicted data manifests what is easily and unambiguously perceived by humans. If predicted data does not put across emotional states the way humans can recognise them, then there is no point in trying hard to achieve an unperceivable differentiation, or a distinction that is not agreed upon among psychologists. Additional explanations of the

approaches (the categorical and the dimensional approach) can be found in chapter Emotion annotation schemes.

On the back of this, discussion is piloted to the method that was used in this project. Conclusions hereby presented were not entirely made in a chronological order – as a consequence of foreseeing the faults of the techniques used – but rather as a result of learning on one’s own mistakes. The main issues of interpreting affective displays are the choice of using discrete or continuous emotional labels, and cherry-picking features from profusion to choose from. The discrete emotional categories were here obtained by mapping the continuously annotated recordings into eight octants. The stated approach is underpinned by the fact that by mapping emotions to eight or so sections of a two-dimensional plane no perceivable loss in clarity in predicted emotions occurs in comparison to emotions represented as objects in a two-dimensional continuous space. Thereupon making this approach a mixture of the discrete and the continuous approach with little or no notable loss in usefulness. It has been established that by mapping data into a two-dimensional plane where emotions are characterised by their octant position does not make the scheme lesser in merit. Furthermore, to better apprehend how the project was organised, here is the list of carried out steps:

1. Given database’s features were evaluated
2. New features were proposed (not invented, but selected)
3. Functionals were chosen
4. The functionals were computed once the feature files were created using the openSMILE toolkit
5. Suitable classifier was chosen
6. Classifier’s parameters were adjusted
7. Equally distributed data sets were created (the training and the testing data set)
8. Accuracy and computation time were noted
9. The code was implemented on RPi
10. The computation times were comparatively assessed
11. The testing data set was decimated to attain better RPi performance

Given database’s features were deduced as superfluous for use by virtue of their inadequacy when it comes to the score obtained from the classifier trained on the initial data, without computing new features or functionals. The classifier was trained against the data

with original features computed by database's creators, which did not produce satisfying results. What is more, in different papers, [2], alike conclusions were drawn which indicate to the usage of groups of features that are evidently similar. The recommended features presented in [3] come from an interdisciplinary meeting of voice and speech scientists in Geneva and they were further developed at Technical University Munchen (TUM). Criteria used in choosing the parameters were as follows:

- 1) The potential of a parameter to index psychological changes in voice during affective processes
- 2) The frequency and success of certain parameter deduced from previous work (literature)
- 3) Parameter's theoretical significance

Two versions of acoustic parameter sets are advocated in [3]. Researches from [3] claim to have chosen the parameters wisely according to the extensive research carried on the part of several independent researchers. First of the two feature sets is the minimalistic set (GEMAPS) and the second is the extended parameter set (eGEMAPS). The minimalistic set owes its convenience to eighteen low-level descriptors typically divided into three groups: frequency related parameters, energy (amplitude) related parameters and spectral parameters. The GEMAPS parameter set is an interdisciplinary attempt to agree on a minimalistic parameter set, based on multiple sources, interdisciplinary evidence and theoretical significance.

4. FEATURES

An early survey from the literature, [4], summarises a few decades of affective speech research, and concludes from the empirical data presented, that loudness, fundamental frequency mean, variability, range and energy of a speech signal show correlation with prototypical vocal affective expressions such as stress (intensity, F0 mean), anger and sadness (all parameters) and boredom (F0 and range). Same is overviewed in [5] . Furthermore, speech and articulation pace were also found to be important for setting emotions apart. Particularly, work concerning an automatic unsupervised arousal framework in which the pace features are brought up is [6]. In other work, [7], acoustic analysis of fundamental frequency and harmonics related parameters was performed. Previous paper, [7], explains the experiments done on a small set of emotional speech utterances, where the authors have confirmed the importance of F0 and a spectral distribution as cues to affective speech content. Some other features were also mentioned in the previously cited papers which are not going to be examined because of their limited applicability to this project.

Following **Table 1** lists features used in this thesis. The below listed features are representative of a good feature-set for acoustic affect modelling. In the column on the right clarification of the features on the left is given.

TABLE 1 TABLE OF USED PARAMETERS TAKEN FROM THE EGMAPS AND THE GMAPS

Frequency related parameters:	
Pitch , logarithmic F0 on a semitone frequency scale, starting at 27.5 Hz (semitone 0)	Pitch is often described as a relative scale in which 1 octave corresponds to a doubling in fundamental frequency, and an octave is divided into 12 semitones. This musical scale is effectively log F0. Pitch is a perceptual consequence of F0. Pitch is qualitative and F0 is quantitative (i.e., we can objectively measure it from a signal). It is sufficient to state that our perception of pitch depends only on F0.
Jitter , deviations in individual consecutive F0 period lengths	Jitter (absolute) is a cycle-to-cycle variation of the fundamental frequency, i.e., the average absolute difference between consecutive periods, expressed as: $J = \frac{1}{N-1} \sum_{i=1}^{N-1} T_i - T_{i+1} \quad (1)$
Formant 1, 2, and 3 frequency, centre frequency of first, second, and third formant	A formant is a concentration of acoustic energy around a particular frequency in the speech wave. There are several formants, each at a different frequency, roughly one in each 1000 Hz band. Or, to put differently, formants occur at roughly 1000 Hz intervals. Each formant corresponds to a resonance in the vocal tract. Additionally, one can recall the source-filter theory of speech production.
Formant 1, bandwidth of first formant	Formant bandwidth is known to have little effect on vowel quality. It has a strong effect on mutual masking between vowels. A narrow-bandwidth voice is thus more resistant to masking, and a stronger masker than a wide formant vowel [8]. This paper considers the relation between the vocal tract characteristics (bandwidth of formants etc.) under the personality of speech [9].
Energy/Amplitude related parameters:	

<p>Shimmer, difference of the peak amplitudes of consecutive F0 periods</p>	<p>Shimmer (dB) is expressed as the variability of the peak-to-peak amplitude in decibels, i.e., the average absolute base-10 logarithm of the difference between the amplitudes of consecutive periods, multiplied by 20:</p> $S(dB) = \frac{1}{N-1} \sum_{i=1}^{N-1} \left 20 \log \left(\frac{A_{i+1}}{A_i} \right) \right , \quad (2)$ <p>where A_i are the extracted peak-to-peak amplitude data and N is the number of extracted fundamental frequency periods.</p>
<p>Loudness, an estimate of the perceived signal intensity from an auditory spectrum</p>	<p>Loudness is an attribute of auditory sensation in terms of which sounds can be ordered on a scale.</p>
<p>Harmonics-to-noise ratio (HNR)</p>	<p>The HNR is the relation of energy in harmonic components to energy in noise-like components</p>
<p>Spectral (balance/shape/dynamics) parameters:</p>	
<p>Alpha Ratio, ratio of the summed energy from 50-1000 Hz and 1-5 kHz</p>	<p>Alfa ratio is defined as the ratio between the energy in the low frequency region and the high frequency region. More specifically, it is the ratio between summed energy from 50 to 1000 Hz and 1 to 5 kHz, expressed as:</p> $\rho_\alpha = \frac{\sum_{m=1}^{m_k} X(m)}{\sum_{m=m_k+1}^M X(m)} \quad (3)$ <p>Where m_{1k} is the highest spectral bin index where $f \leq 1$ kHz is still true.</p>
<p>Hammarberg Index, ratio of the strongest energy peak in the 0-2 kHz region to the strongest peak in the 2–5 kHz region</p>	<p>The measure was defined by Hammaberg in 1980 as the ratio of the strongest energy peak in the 0-2 kHz region to that of the strongest peak in the 2-5 kHz region. Hammaberg defined fixed static pivot point of 2 kHz where the low and high frequency regions are separated. Symbolically the index is defined as:</p>

	$\eta = \frac{\max_{m=1}^{m_{2k}} X(m)}{\max_{m=m_{2k}+1}^M X(m)}, \quad (4)$ <p>Where m_{2k} is the highest spectral bin index where $f \leq 2$ kHz is still true.</p>
Spectral Slope 0-500 Hz and 500-1500 Hz	Spectral slope is a linear regression slope of the logarithmic power spectrum within the two given bands.
MFCC 1-4 Mel-Frequency Cepstral Coefficients 1-4	Mel-frequency cepstrum (MFC) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency.
Spectral flux , difference of the spectra of two consecutive frames	The spectral flux of restricted frequency bands, or sub-band flux, is a computational measure indicating the extent to which the spectrum changes over time.
Harmonic difference H1-H2	The H1-H2 harmonic difference is the ratio of energy of the first F0 harmonic (H1) to the energy of the second F0 harmonic (H2)
Harmonic difference H1-A3	The H1-A3 harmonic difference is the ratio of energy of the first F0 harmonic (H1) to the energy of the highest harmonic in the third formant range (A3)

Throughout machine learning related papers handcrafted feature sets are used to simplify the task of working with complex data. Similarly, for recognition of affective displays handcrafted features are advocated instead of making use of large feature sets, however, optimal compact feature set has not been agreed upon. There exists copious of related work wherein massive feature sets are used after which dimensionality reduction is done to save only the features with at least some relevance to the specified task. Popular emotion recognition challenges prove the aforesaid. For example, the INTERSPEECH compare paralinguistic challenge, [10], makes the trend of using large feature sets obvious. However, in AVEC 2016 challenge, [2], efforts are aimed towards making use of a reduced feature set, specifically the eGeMAPS. Interesting assessment of performance of the

eGEMAPS was undertaken in [3], where comparative evaluation against the feature set used by the INTERSPEECH compare challenge is performed. The benefits of the eGEMAPS have been demonstrated in [3] for as much as to the fact that the eGEMAPS only amounted to 1.4% of the size of the best performing feature set from the INTERSPEECH compare challenge, while being competitive in performance. Moreover, since MFCC coefficients have proven to be highly successful, [11], they are included in the set used in this thesis on top of the features from the GEMPAS, also the Spectral flux feature was added to the used set. Both the Spectral flux and the MFCCs are included in the extended eGEMAPS feature set from [3], not the GEMPAS. Conversely, some features such as the formant 1, 2, and 3 relative energy, as well as the ratio of the energy of the spectral harmonic peak at the first, second and third formant's centre frequency to the energy of the spectral peak at F0 were taken out from the minimalistic GEMAPS feature set. Because of that and in contribution to proven successfulness of the eGEMAPS, feature set used in this thesis was assembled of mostly the features from the GEMPAS with some additional, valuable for this task features from the extended set. The feature set is detailed in **Table 1**. Minimalistic feature sets analogously as large-scale, brute-force feature sets compute LLDs on a time window. An optimal time window's length is considered generally disputable, yet for specific applications it is not coincidentally chosen. Time window's length ensures that LLDs computed meet the condition of temporal stationarity. Given the circumstances of herein described task, the openSMILE toolkit in conjunction with the GEMAPS corresponding configuration file was utilised to compute the LLDs with the window that is 20 milliseconds long for energy, spectral and formant features, and 60 milliseconds long for the remaining features. The openSMILE toolkit is a publicly available, free toolkit, with some functionalities not freely obtainable, most commonly used in alike cases. The time window mentioned here is not the time window used for computing functionals.

A more complex MFCC coefficients computation is going to be clarified here to provide an insight to coefficients' remarkable ability to extract characteristics entrenched in vast majority of emotive speech signals. Even though there exists ample of other alike features, MFCCs are popular, and that is for a reason. MFCCs are decorrelated Mel-filter bank coefficients (MFBs). Triangular filters scaled in accordance to human perception are used to produce MFBs; each filter bank represents the contribution of a band of frequencies similarly perceived by humans. The discrete Cosine Transform (DCT) is then applied to

filtered data to convert MFBs into the time domain, producing MFCCs. The DCT has by consequence of that the role of decorrelation of the features.

Not all features are as complex in their computation, although they require attention. Averaging over the waveform for computing loudness, for example, is not how humans would observe it, hence the method which consists of using 26 triangular filters is employed for calculating MFBs. Upon MSBs computation each filter bank is weighted and scaled to create an auditory spectrum that is then used for loudness extraction by summing over the spectrum.

4.1. FUNCTIONALS

Functionals are a measure that accounts for any sudden spikes in a speech utterance by smoothing the utterance; all functionals are computed on a fixed-length window applied to an utterance. The fixed-length windows (not to be confused with the time windows from Features) are overlapped and shifted 40 milliseconds forward in each step until the end of the utterance is reached. The shift length (or an overlapping step) is determined by consulting literature and taking into consideration characteristics of emotive speech signals. At the very beginning of the project only mean was computed with successive windows without overlapping. Poor results of that approach ensued from the negligence of the characteristics of the data that was annotated continuously in time by movements of the sliders across the screen. The major shortcoming of doing described is rapid database size shrinking caused by averaging or representing neighbouring samples in chunks. Another drawback is the undermining of the process of annotating the data set, i.e., neglecting the insufficient smoothing of the annotations. After having examined the influence of a few approaches to the resulting size of the data set and considering the sampling rate of the openSMILE toolkit, it was decided that the fixed-length windows should be 2.5 seconds long with the overlapping step of 40 milliseconds. There is yet another reason why such overlapping length was chosen – to be consistent with the annotation sampling rate of 40 milliseconds. To achieve accordance between the annotation files and the feature files the beginning and ending part of the annotation files were removed to compensate for the usage of the fixed-length windows. The centre of each window is the point in time with inclined significance over side parts of the windows, which is why the functionals (or statistical measures) are

computed over the whole fixed-length window. Furthermore, the decision to compute functionals on a 2.5 second window with an overlapping step of 40 milliseconds is in line with the paper from literature [2]. To make this chapter wholesome there follows a table of used functionals. In **Table 2** used functionals are listed.

TABLE 2

Used functionals:	
Percentile 25, 75	Percentile is the value below which given percentage of observations falls. Taking a distribution for example, if we say that the percentile 25 equals to 0.01, it means that 25% of the data from the distribution falls below 0.01 in the distribution sample range. Using the limit for example, as samples approach infinity, the percentile approximates the inverse of the CDF function.
Coefficient of variation (COV)	The COV, also known as relative standard deviation, is a standardised measure of dispersion of a distribution. It is defined as the ratio of the standard deviation and the mean, i.e., it denotes the variability of the distribution.
Standard deviation (SD)	Standard deviation quantifies the amount of variation of a set of data values. The formula of the SD is: $s = \sqrt{\frac{\sum_{i=1}^N (x_i - \bar{x})^2}{N - 1}}. \quad (5)$ Where $x_1, x_2, x_3 \dots$ are the observed values, and N is the number of samples.
Mean	
Min	
Max	

5. DATABASE

A new multimodal corpus of spontaneous interactions in French called RECOLA, for Remote Collaborative and Affective interactions, was recently introduced by Ringeval in [12]. Participants were indulged in a survival collaborative task that was performed in dyads, i.e., in groups of two people, and remotely by video conference. The RECOLA database is a multimodal database that includes 9.5 h of recordings; precisely it consists of audio, video, electro-cardiogram (ECG) and electro-dermal activity (EDA) modalities, all of which were synchronously and continuously recorded for 46 participants. The database is annotated by six French-speaking annotators who used the ANNEMO web-based tool. Not all participants were native French speakers, even though all of them spoke French during the experiment; there were 17 native French, three native German and three native Italian speakers. Annotators were, on the other hand, all native French speakers. Notwithstanding the original number of participants, only a fraction thereof agreed for their recordings to be publicly available, as a result of which the database has not got 46, but 23 recordings. At the website where the database was downloaded there is another folder with 27 recordings prepared for and used in AVEC2016 challenge [2]. Initial recordings' names are, therefore, not consecutively numbered; their names stem from the full database's names, prior to its reduction. To address mentioned, the reading from the folders was performed once the files were renamed. Furthermore, the database comes with dimensional annotations, since they have been proven to be successful in an affective content analysis.

In this database multimodal information sources serve the purpose of studying communication modalities, for instance gestural and facial expressions, however, only audio recordings were used in this project.



FIGURE 3 THE RECORDINGS WERE ANNOTATED USING THE ANNEMO WEB-BASE ANNOTATION TOOL. THIS FIGURE IS TAKEN FROM [12].

In the process of constituting the corpus, mood induction techniques were introduced to broaden the valence dimension range. Facilitators of the experiment decided on mood to be induced in one of the participants based on a previous participants inquiry. The survival task employed [12], originally designed by NASA, involved high stakes in resolving the dilemmas participants were faced with, which made it suitable for artificial surroundings because it managed to provoke real-life emotions in a laboratory environment.

Figure 3 depicts the annotation procedure that included two steps – annotating valence and arousal by moving the sliders across the screen. Final annotation files consist of six annotations, since the one from the non-native annotator was removed.

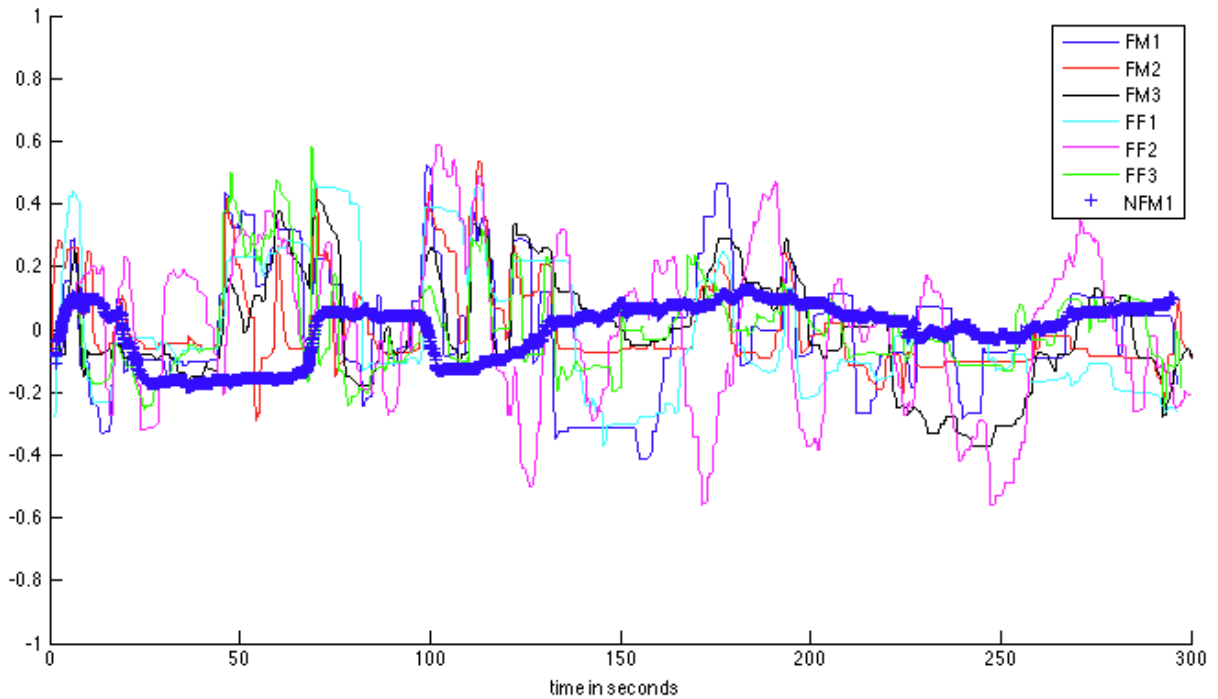


FIGURE 4 THIS FIGURE REPRESENTS VALENCE ANNOTATIONS WITH ZERO MEAN NORMALISATION FROM SEVEN ANNOTATORS; THREE MALE, THREE FEMALE AND ONE NON-NATIVE ANNOTATOR. THIS FIGURE IS TAKEN FROM [12].

From **Figure 4** it is obvious that native annotators did assess valence in a similar manner despite the non-negligible inter-rater disagreements. However, the non-native annotator (the blue line) did not rely on linguistic information, just the non-verbal cues, and that is why his latent and attenuated annotations were excluded from the annotation files. On that note, it is often highlighted how important it is to have reliable ground-truth annotations for dimension-based affect modelling and analysis [1]. To this date, the conundrum of whether to use self-assessed annotations or the ones coming from observers is still something researchers are working on. Some works infer to the profits of a subjective assessment while the others are preoccupied in stressing the objectiveness and the unbiasedness achieved by applying statistical metrics on a range of multiple annotators' assessments. In this work, seven annotators had been given the assignment of assessing valence and arousal separately by moving the slider, however, there exists models where greater accuracy was facilitated using self-assessment labels, in defiance of the intuitive ambiguity associated with assessing arousal while being emotionally overwhelmed. Arousal was the dimension whose self-assessed annotations were discussed in [1]. Whether the same holds independently of the utilised modalities and cues, or can the analogy be drawn for valence, is open for discussion. Further, cleverer approach, according to the authors, to achieve a more reliable ground-truth is laid out in [2], where the golden standard is addressed.

Additionally, accuracy evaluation could be done on one annotator at a time for all six annotators. Such evaluation would provide an insight in inter-rater differences and it would free the performance assessment of a burden caused by averaging annotations across six annotators. Specifically, if a person who annotated recordings unusually high or low can be spotted, that annotator would not worsen accuracy in a practical application in which there are no labels (annotations) in the prediction step. The only thing that would matter, though, is how well can an emotion be classified based on a classifier trained with hopefully properly labelled training data set, and not how well can an emotion be predicted compared with labels coming from the same annotators who labelled the training data set. In this project labels from six annotators were averaged and used in the assessment of the classifier's performance in the prediction step.

Authors of the database said that they had used different normalization techniques to improve the inter-rater discrepancies. Further, they describe how they mitigated the problem of missing values in the annotation files that had only occurred for no more than 20 seconds at a time, but still required their attention. To alleviate the problem, a piecewise cubic interpolation was used.

Since this database is a conflation of multi-modal sources, to achieve compatibility with the rest of the information modalities, the audio recordings' annotation files were binned to resulting, here used files, with the frame rate of 40 milliseconds. The sample rate of 40 milliseconds because of that going to have to be deliberated when the openSMILE toolkit gets utilised for extracting the project specific features and functionals. While extracting the features, decimation from 10 milliseconds to 40 milliseconds is ought to be performed to achieve congruency between the annotation and the feature files.

6. PYTHON FUNCTIONS AND LIBRARIES

One of the extenuating circumstances of the task of this project was the availability of free and open-source Python libraries used for technical computing. For example, SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering. SciPy builds on the NumPy array object and is part of the NumPy stack which includes tools like Matplotlib, pandas and SymPy, and an expanding set of scientific computing libraries. The basic data structure used by SciPy is a multidimensional array provided by the NumPy module. NumPy provides some functions for linear algebra, Fourier transforms, and random number generation, but not with the generality of the equivalent functions in SciPy. Refer to [13] for more on the subject.

Utilisation of available functions, particularly machine learning based functions, is a convenient way of taking advantage of decades of condensed mathematical knowledge. If machine learning functions were rewritten every time a task of a similar type shall be delivered, that would make the exploration of applicability of different classifiers to specific tasks virtually impossible. Classification algorithms implemented in machine learning functions are complex enough to require investigation in their own right. Reinventing the aforementioned algorithms that are based on well-established theorems and corollas from the 20th century would mean spending a great deal of time and effort every time a project is being carried out. In that respect, the algorithms in the form of functions are going to be tapped into for the purposes of this project. The functions cull their embedded knowledge from various theoretical sources; also, they were written by different authors. The tactic is, thence, to deploy tools that were meant to make future relating work more effective by switching focus from the parallelism and futility of reinventing what exists to the joint efforts of researches around the world in achieving task specific attainments. All the potential downsides that coding one's own classifier entails are bypassed by making use of the Python functions which come in easily obtainable packages. None the less, the convenience of the pre-coded machine learning functions does not ensue their instant applicability to a wide range of tasks; the functions need to be interfered with in terms of adjusting their parameters.

Ubiquitous practice of dealing with data sets is splitting the whole data set into three parts: a training, a development and a testing part. Mentioned splitting scheme is an

assurance that the classifiers' parameters will not be optimised on the same data set on which testing is performed. In other words, an evaluation (development) data set exists so that classifier's parameters can be evaluated by testing the choice thereof on a separate data set from a testing data set (later in text referred to as a split). If the evaluation is performed on a split that is afterwards used as a testing split, then such an optimisation is not valid, for it overlooks the overfitting that happens in that case. Say we take a data set and intend to optimise classifier's parameters on a split created as a fraction of the original data set, and then if we use the same split for testing the classifier's performance, resulting accuracy will not be unbiased nor will it be repeatable.

Moreover, there exists even more statistically content solution for optimising classifier's parameters, which was employed in this project. It is called cross-validation, a method that improves a standard technique by performing the splitting arbitrary amount of times. In this project, the independency of the testing split is ensured by splitting the data set into three parts. Along with the splitting, the evaluation is performed in each iteration of the algorithm. Hence, the optimisation is not only performed once, but numerous times, each of which is characterised by a different evaluation split. The tactic implies training the classifier several times, which is done at the expense of the increased computation time. The technique here described is encouraged throughout the literature for its notable benefits for parameters tuning, despite that it comes at a price.

In **Figure 5** all used Python packages are listed.

```
1 import pickle
2 from os import path
3 from os import listdir
4 from os import getcwd
5 import pandas as pd
6 import subprocess
7 from os import chdir
8 from scipy.stats import variation
9 import numpy as np
10 from sklearn import svm
11 from sklearn.svm import LinearSVR, SVC
12 from sklearn.decomposition import PCA
13 from sklearn.preprocessing import StandardScaler
14 #from extraction_utils import get_sync_delay, pad_sync_wav_file
15 from sklearn.model_selection import train_test_split
16 from sklearn.model_selection import GridSearchCV
17 from sklearn.metrics import classification_report
18 from sklearn.utils import shuffle
--
```

FIGURE 5 USED PYTHON PACKAGES

On the Raspberry version of the code only the testing of the classifier and the features and functionals computation are implemented, thus it requires fewer packages. In Figure 6 all packages used on the Raspberry Pi are listed.

```
1 import pandas as pd
2 from sklearn.decomposition import PCA
3 from sklearn.preprocessing import StandardScaler
4 import numpy as np
5 import pickle
6 import time
```

FIGURE 6 USED PYTHON PACKAGES ON RASPBERRY PI

The two most commonly utilised classifier functions are the `svm.SVC` and the `svm.SVR`. Letter C stands for classification, while letter R stands for regression. Classification is to be used with categorically annotated data, whereas regression is to be used with continuously annotated data. Apart from the `svm.SVC`, there are other alternatives to choose from, for instance, the `neighbors.KNeighborsClassifier()`. The choice comes down to a particular application and hinges upon the annotation method of a testing file. Due to everything so far mentioned in this chapter such as splitting a data set and emphasising the gains of using cross-validation for tweaking classifier's parameters, the `svm.SVC` classifier is chosen and optimised via cross-validation. Cross-validation is implemented using the `GridSearchCV()` function. The chosen kernel function is `rbf`. Kernel functions are highly influential on the outcome of classification. To better comprehend its meaning, an overview of the Support Vector Classifier with mathematical insights of the theorems that were taken advantage of in the classifier construction follows.

6.1 MATHEMATICAL BACKGROUND

The Support Vector Machine's history and the course of events that led to its widely accepted role in up-to-date machine learning tasks is discussed in History of machine learning, where reader's attention is directed towards its dependency on computing power of today's computers. In this chapter, the objective of the SVM and its inner workings are elucidated. If we take two-dimensional case as an example, finding the best splitting boundary between data is akin of finding the best splitting line, however, if we extend the analogy to N-dimensional data, we are talking about the separating hyperplane. The best separating hyperplane is the hyperplane that contains the widest margin between supporting vectors.

Another name sometimes used as a synonym for the separating hyperplane is the decision boundary. Intuition would imply that to classify a sample, proximity from the sample and all the remaining samples should be calculated, which is what the `neighbors.KNeighborsClassifier()` classifier does, although that is not necessary when the SVM is utilised. The biggest disadvantage of employing the KNN is poor scalability which is the corollary of computing the distances per every datapoint. Fairly comparable results of the KNN and the SVM accuracy-wise aside, the SVM aims to mitigate the problem of lousy scalability by implementation of the separating hyperplane. The hyperplane is computed only once when the classifier is trained, thus notably improving classification time even for small databases. On an intuitive level it is not unconceivable how the separating hyperplane is being positioned in a space by iteratively maximizing the margin between support vectors until the final decision is made upon convergence. The total width between support vectors is, therefore, the margin. Moreover, literature suggests that not only is the SVM superior in terms of pace at which classification is performed, but it is a more reliable algorithm in comparison with the KNN in handling outliers. The aforesaid undermines the non-linearity typically inseparable from machine learning tasks associated data sets and the principles of handling thereof. Bearing in mind that data sets (or sometimes referred to as feature-sets) are made up from features whose number depends on the task, invoking linear algebra procedures will not be sufficient. Indeed, other assertions or constraints made upon the SVM when dealing with multi-dimensional data shall be pondered that will prove to be integral for both undertaking the math behind the algorithm and optimising SVM's parameters.

Let us define classification function:

$$\mathbf{sgn}(\mathbf{w} \cdot \mathbf{x}_i + \mathbf{b}). \quad (6)$$

By observing the function value, we can determine to which class testing sample, \mathbf{x}_i , belongs. In above equation \mathbf{w} is weight, a perpendicular vector to the separating hyperplane, and \mathbf{b} is bias. From this point on \mathbf{x}_i will be called a datapoint. Therefore, the sign is the key for differentiating between two classes. Further, let us define support vectors:

$$\mathbf{w}_0^T \mathbf{X} + \mathbf{b}_0 = 1 \quad (7)$$

Support vectors are x from above. We divide support vectors into negative and positive depending whether the above equation equals -1 or 1 respectively. If we write the above equation and the one that equals to -1, it corresponds to negative support vectors. By multiplying both equations with the class value (or annotation label) y_i , we get:

$$y_i(x_i \cdot w + b) - 1 = 0. \quad (8)$$

Where y_i equals -1 and 1 for the equations defining negative and positive support vectors respectively. The above equation is the joint definition of support vectors from two classes. Next, we define the margin hyperplanes (later called the positive and negative hyperplanes) as the hyperplanes that include support vectors, and are a boundary separating the training data set from the margin zone on both sides.

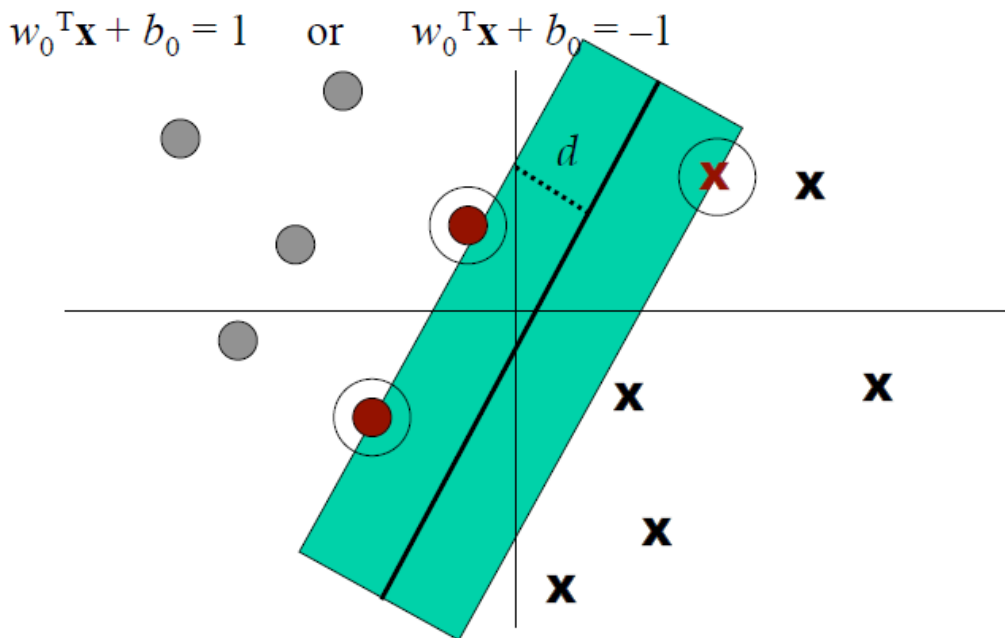


FIGURE 7 THIS FIGURE REPRESENTS THE MARGIN ZONE WITH DATAPOINTS BELONGING TO TWO CLASSES THAT ON ITS BOUNDARIES HAVE SUPPORT VECTORS (CIRCLED).

On the top of **Figure 7** defining equations for negative and positive support vectors are given. For the training data set we impose a constraint on their proximity from the separating hyperplane:

$$\begin{aligned} w \cdot x_i + b &\geq +1 \text{ when } y_i = +1 \\ w \cdot x_i + b &\leq -1 \text{ when } y_i = -1 \end{aligned} \quad (9)$$

The positive and negative hyperplanes are defined as:

$$\begin{aligned} H_1: w \cdot x_i + b &= +1 \\ H_2: w \cdot x_i + b &= -1 \end{aligned} \quad (10)$$

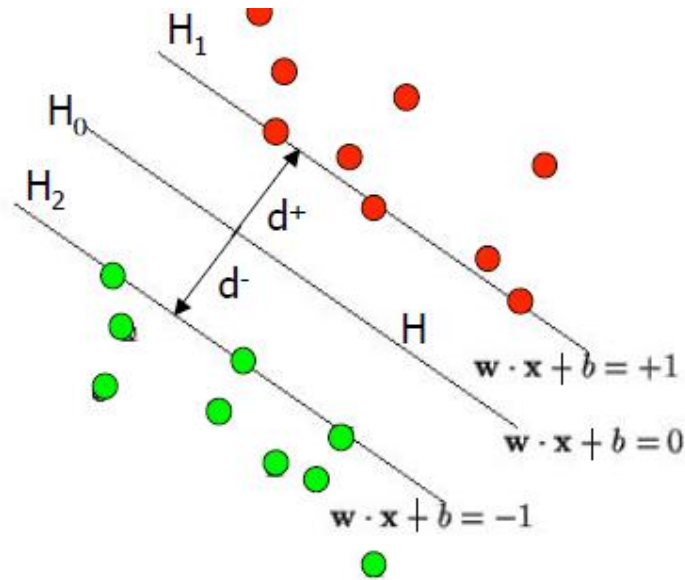


FIGURE 8 THIS FIGURE SURMISES WHAT IS WRITTEN IN THE TEXT REGARDING THE DEFINITIONS OF THE SEPARATING, POSITIVE AND NEGATIVE (MARGIN) HYPERPLANES.

In **Figure 8** the positive and negative hyperplanes are visually represented in a two-dimensional plane. Also, in **Figure 8** the separating hyperplane is noticeable; it is defined as:

$$w \cdot x_i + b = 0 \quad (11)$$

The constraint on a training data set from (9) is not applied on datapoints used for testing; datapoints used for testing can be anywhere between the separating hyperplane and the positive and negative hyperplanes. Next, constraint optimisation implies minimizing $\|w\|$ and maximising b . We are trying to maximise the width between the positive and negative hyperplanes while obeying the discrimination boundary. To confirm stated, we denote the width as:

$$W = (x_+ - x_-) \cdot \frac{w}{\|w\|}, \quad (12)$$

where x_+ and x_- are support vectors. After plugging support vector definition from (8) in equation (12), we get:

$$W = \frac{2}{\|w\|}. \quad (13)$$

Hence, to maximise the with, $\|w\|$ is minimised; or $\frac{1}{2}\|w\|^2$ is minimised.

As a result of what was written in chapter 6.1 Mathematical background, quadratic programming problem has arisen:

Min: $f: \frac{1}{2}\|w\|^2$ (this is a quadratic function), such that $g: y_i(x_i \cdot w + b) - 1 = 0$.

This a constrained optimisation problem and it can be solved by the Lagrangian multiplier method. The fact that it is a quadratic problem signals that the surface is paraboloid with just a single global minimum, thus avoiding problem associated with neural networks. Next, an intersection of f and g needs to be found at a tangent point (tangent means that the derivative equals zero). An intersection is min (or max) for f , such that the constraint is satisfied. Same thing can be recast by defining two constraints with new variables – a and λ .

$$\begin{aligned} \nabla f(p) &= \nabla \lambda g(p) \\ g(x) &= 0 \end{aligned} \quad (14)$$

In (14) we are looking for a solution point p . If we introduce the Langrangian L and require the derivative of L to be zero:

$$\begin{aligned} L(x, a) &= f(x) - ag(x) \text{ where} \\ \nabla(x, a) &= 0 \end{aligned} \quad (15)$$

Partial derivatives with respect to x will recover the parallel normal constraint. Partial derivatives with respect to λ will recover second constraint from (14). In general, first row of (15) becomes:

$$L(x, a) = f(x) + \sum_i a_i g_i(x) \quad (16)$$

In our case:

$$f(x): \frac{1}{2}\|w\|^2 ; g(x): y_i(w \cdot x_i + b) - 1 = 0, \quad (17)$$

so Lagrangian is:

$$\min L = \frac{1}{2} \|\mathbf{w}\|^2 - \sum a_i y_i (\mathbf{w} \cdot \mathbf{x}_i + b) + \sum a_i \text{ wrt } \mathbf{w}, b \quad (18)$$

From the property that derivatives must equal zero at min:

$$\mathbf{w} = \sum_{i=1}^l a_i y_i \mathbf{x}_i, \quad \sum_{i=1}^l a_i y_i = 0 \quad (19)$$

To solve the dual of this problem, w and b from (18) need to be substituted with (19). In the dual formulation maximization is necessary:

$$\max L_D(a_i) = \sum_{i=1}^l a_i - \frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l a_i a_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \quad (20)$$

To solve (20) for a , only inner products are needed. After solving (20) for a , w can be found. Now that w is computed (once the training finishes), we can classify an unknown point by looking at the sign of

$$f(x) = \mathbf{w} \cdot \mathbf{u} + b = \left(\sum_{i=1}^l a_i y_i \mathbf{x}_i \cdot \mathbf{u} \right) + b \quad (21)$$

To surmise, in this chapter a quadratic programming problem has been introduced by means of which the training of the SVM classifier is performed. The problem is based on the maximization of the width, and it progresses into a quadratic optimisation problem, which is solved by the Lagrangian multiplier method.

In contribution to the two-dimensional theory, its N-dimensional non-linear extension is analogously conceivable. In a general case non-linear functions which map the original feature space into a higher dimensional space are utilised. Non-linear functions or kernels guarantee the existence of a higher dimensional space and make the mapping possible without stringent constraints.

As for the multi-class implementation of the SVM, it can be obtained by training several classifiers and combining their results. There are numerous strategies for combining SVMs; two methods that can be chosen by adjusting Python parameters are called “one per class” and “pairwise coupling”. The “one per class” method includes training one classifier per class that discriminates between that class and the rest of the classes. Each class’ corresponding classifier is assuming a positive value belongs to that class, and a negative value to any of the other classes. Classifier’s output is the index i of the largest discriminating

function f_i . The most commonly used discrimination function is the signed distance between a datapoint and the separating hyperplane:

$$f_i(\mathbf{x}) = \frac{\mathbf{w}_i \cdot \mathbf{x} + b_i}{\|\mathbf{w}_i\|}, \quad (22)$$

where w_i and b_i are the normal and the bias of the separating hyperplane of the i -th SVM classifier respectively. Final decision is made by finding the maximal discriminating function:

$$c(\mathbf{x}) = \arg \max_{i \in \{1, \dots, K\}} f_i(\mathbf{x}), \quad (23)$$

where K is the number of classes.

The pairwise coupling method trains $K(K-1)/2$ binary SVM classifiers. Each of these classifiers is trained to discriminate between two classes. To classify a datapoint, the pairwise coupling method combines the discrimination functions of $K(K-1)/2$ classifiers by using a voting scheme. When the number of classes is high the pairwise coupling method requires training with a large number of SVM classifiers.

In Figure 9 is the code wherein cross-validation is implemented:

```

4 parameters = [{'kernel': ['rbf']},
5               'gamma': [0.01, 0.1, 0.5],
6               'C': [1, 10]]
7
8 clf = GridSearchCV(svm.SVC(kernel='rbf'), parameters, verbose=4) #decision_function_shape='ovr'
9 clf.fit(X_train, y_train)
10
11 with open('RBFwithGridSearch', 'wb') as f:
12     pickle.dump(clf, f)
13
14 print("Best parameters set found on development set:")
15 print()
16 print(clf.best_params_)
17 print(clf.best_score_)
18 print()
19 print("Grid scores on training set:")
20 print()
21 means = clf.cv_results_['mean_test_score']
22 stds = clf.cv_results_['std_test_score']
23 for mean, std, params in zip(means, stds, clf.cv_results_['params']):
24     print("%0.3f (+/-%0.03f) for %r"
25           % (mean, std * 2, params))
26

```

FIGURE 9

Gamma value with the greatest accuracy equals 0.01; and C value equals 10 for the data used in the project. Performance of each option can be evaluated by examining means

and standard deviations of the scores, since the classifier is trained several times for each parameters combination. The number of times the classifier is trained for each parameters permutation is determined by the number of splits of the data set used for training and evaluation, i.e., by signifying the number of splits to the `GridSearchCV()` function via changing the CV parameter. By default, the CV parameter is set to 3, thus, indicating to three-fold cross-validation. The estimator that is chosen by the search gets stored in the `best_estimator_` attribute; the `best_score_` attribute bears the information of the mean cross-validated score of the best estimator; and the `best_params_` attribute stores the parameter setting that gave the best score for the best estimator. In this case an estimator choice was not evaluated in the `GridSearchCV()` function, only the other two attributes were examined, for the estimator choice was decided ahead of its evaluation through the use of cross-validation. To clarify, an estimator choice denotes used classifier together with its parameters such as the kernel function parameter. The classifier used is the `svm.SVC` and the kernel of choice is `rbf`. The range of parameters is not wider because it would be too demanding in the sense of computation time. The CV parameter was not interfered with; it was set on default value. Cross-validation splits a testing data set into N parts, each of which is once used as an evaluation split for parameter tuning. After the decision on the best parameters is made, the cross-validation function then fits the whole testing split to a classifier with respect to the best parameters. More about splitting data sets can be found in Python functions and libraries.

7. DATABASE PREPARATION

The database used consists of 23 audio recordings, some of which were originally assigned to the testing data set while others were allocated to the training data set. However, that approach was not applicable to the database because of the unevenly distributed affective expressions. The reason being that the database was not properly balanced during its creation, making for the emotionally biased recordings that cannot be a realistic representation of what might be expected in a speaker independent classification case. Mainly, the size of a database and even distribution of affective states determine if a database can be utilised for modelling a speaker independent recognition scheme. In that regard, the size of the database was not sufficient to represent an unbiased set of recordings from which a random recording could be taken out as an adequate candidate for testing the classifier. Described obstacle was at first assumed to be true after the results of the testing turned out to be unsatisfactory in the sense that they were not accurate unless the database's samples were not shuffled. Which brings us back to chapter Functionals where the functionals computation on a fixed-length window was explained, underlining the statistical dependency of neighbouring samples (or datapoints). Hence the reason why were the results seemingly convincing when the database's samples had been shuffled. Notwithstanding the potential usefulness of shuffling data for other applications, doing it would not even remotely be similar to the situations that portray practical implications. Thereby, random segments were pulled out from all 23 recordings; from each recording two of the segments were conflated into one testing file that should have served as a proof of concept, had the results been acceptable. But that premise proved to be flawed as well; it comes out to be that it was not enough to extract whichever 30 seconds segments, two per 5-minute recording to ensure for the statistical balance of emotive states across the two splits.

This chapter is focused on the database and its specificities, and it aims to lay out the strategies associated with databases with uneven distribution of classes, in this case affective states. Though the problem may be alleviated when dealing with enormous databases in which there are enough recordings so that splits contain classes from across the spectrum due to the sheer size of data put into them, it was not the case in this project. The unfavourable characteristics of this database are attributable to the database's size, the scheme used for exerting emotive states, and the pitfalls that entail from not concentrating enough on deliberate allocation of the data throughout the recordings. To that end, a new

scheme was proposed and tested, whereby satisfying results were attainable due to the apparent uniformity of the data across the splits from the perspective of the classifier. Put differently, thanks to the unbiasedness, as the classifier interprets it, a scheme that transforms the database, giving it the most sought-after properties, allows for the classification to be successful, while at the same time not making allocation of the data to the splits unrealistic. For such a scheme to work, data annotations should be examined, taking all recordings into consideration, and only datapoints that have counterparts in the training split should be assigned to the testing split. If annotations are thought of as attributes of emotive states, then we are addressing equal distribution of affects regarding their attribute dispersion across all recordings. To clarify, it would not make sense to check the annotation of each datapoint at a time because it would make the process unreasonably demanding in terms of computation time, with no tangible improvements obtained in return. In light of this, the algorithm used to account for the database inauspiciousness is going to be succinctly explained here:

- 1) All recordings were divided into segments of 30 seconds.
- 2) To every segment of each recording predominant octant was assigned by inspecting each annotation included in the segment.
- 3) For every segment of each recording percentage of the predominant octant was computed.
- 4) Based on the two variables, the percentage and the predominant octant, the list of segments was sorted, such that the beginning of the list starts with the chunk of first octant segments, following with the chunk of second octant segments and so on. Within every chunk of a certain octant segments list was sorted by the percentage variable.
- 5) From the sorted list first segment from each chunk of segments was assigned to the testing file. Not all segments from the top of every chunk were assigned to the testing files; some of them which were not accompanied with at least one more segment from the same chunk were earmarked for the training split only. One testing file, created by continuously appending all recording's testing segments, and others, containing one recording's testing segments only, were separately saved to memory. This procedure made it possible to test the classifier for every speaker (recording) individually and against the testing file that contains segments from all speakers.
- 6) The training file was filled with corresponding segments parallelly with the testing files.

- 7) The normalized extracted features for training were fed as input to the PCA function with desired retained accuracy of 95%.
- 8) The chosen classifier (the svm.SVC) was trained on the training split using cross-validation technique that further divided the training split into three parts and searched for the best parameters permutation producing cross-validated mean accuracy for every choice of parameters. Upon finding the optimal parameters choice, the same training split was reused in its entirety to refit it to the classifier with the optimal parameters, thus enlarging the split used for training in cross-validation, which was only two-thirds as large as the whole training split.
- 9) The classifier was tested separately for each speaker and jointly for all datapoints from the training split. Several metrics were used for accuracy representation.

The steps described are balancing the distribution of datapoints across the two splits. Reason for not including the segments which do not have at least one other segment below them in a chunk of segments belonging to the same octant to the testing split, lies with the fact that by doing so some octants would not be present in both splits. As the loop that iterates through all recordings was being executed, percentages and predominant octants per segments were printed out, which proved that the database is not only unevenly filled with segments of different kind per recording, but also how diverse the annotations across recordings were.

Consequently, in spite of a trend in the research community to make contributions in speaker-independent systems, considering its complexity and the need for a database of greater size, it does not come as a surprise that there is a limited number of speaker-independent related papers available. In some papers authors refer to speaker-dependent schemes as speaker-independent systems with a limited number of speakers, which is contradictory. In this project the database consisting of 23 speakers was separated into two splits, each of which contains segments deliberately picked so that they model a real-life case of equally distributed data for testing and training. Used scheme, thus, makes certain that the database has sought-after properties, and it is an example of a robust and stable system that demonstrates good generalization thanks to the database preparation. In that context, to this artificially modelled speaker-dependent system the Support Vector Machine as well as K nearest neighbour classifier were applied and the efficiency is presented by means of emotion recognition accuracy. Upon completion of evaluating the classifiers, the

SVM classifier's performance was analysed with different function kernels under diverse parametrisation.

It has been suggested in [14] that the research community is endeavouring to make use of contextual information, such as gender, to achieve speaker-independency. In line with experimental results of this work, the SVM classifier appears to possess global optimality as a training algorithm as well as high performance [15]. Some other authors assert how it is possible to implement a combination of a speaker identification system followed by a speaker-dependent emotion recognition system. To be concise, source [14] provides a definition of a speaker-independent system, although, here used scheme is speaker-dependent. Citation from [14]:

“By the term speaker-independent we mean that the utterances that are included in the test set come from one specific speaker, whose utterances are not included in the training set. In other words, it is not possible for the classifier to be tested on utterances derived from the same speaker whose utterances belong to the training set.”

In all the works so far examined, statistical values of features are computed, and feature selection is applied to retain a small number of features. Frequently mentioned methods are the Sequential Floating Forward algorithm and the Principal component analysis. The latter was applied on the database's extracted features and functionals.

The accuracy will be discussed in chapter Accuracy, where comparison with previous work will offer qualitative conclusions.

8. ACCURACY

TABLE 3

speaker	precision	recall	f1-score	support
1	0.74	0.83	0.78	8759
2	0.67	0.81	0.73	9800
3	0.76	0.70	0.73	4577
4	0.73	0.68	0.70	7830
5	0.74	0.52	0.61	5433
6	0.59	0.38	0.46	604
7	0.55	0.51	0.53	297
8	0.81	0.59	0.68	1926
average / total	0.72	0.72	0.71	39226

Results from **Table 3** were obtained with the `svm.SVC` classifier with `rbf` kernel function, `C=10`, and `gamma=0.01`. Datapoints that comprise both the training and the testing splits are made up from the features computed for this project with the aid of the `openSMILE` toolkit; the features given with the database were discarded. Before the classifier had been trained, database's biasedness was accounted for by purposefully breaking it apart so that the appearance of different octant segments is balanced. Several kernel function options were implemented to monitor their influence on accuracy, which was notably dissimilar. Starting from the linear kernel, whose evaluation never went further from altering the code, because the code never terminated; to the poly kernel which did show relatively poor performance; and the sigmoid kernel which failed miserably. In the end, the `rbf` kernel proved to be both fast and accurate.

In **Table 3** rows represent eight classes to which datapoints were mapped from their original two-dimensional representation in a plane of valence and arousal. Reader shall recall explanation of **Figure 2**, where the dimensionally annotated data is depicted. The dimensional approach requires regression instead of classification, making the computation time longer in comparison, however, attempt to implement two classifiers for valence and arousal has been experimented with. It did not result in expected accuracy nor acceptable computation time. Furthermore, unexplained bias has arisen and was observed after calling the score function, whose meaning is not equal to the meaning of the score function used in

conjunction with the `svm.SVC`. For all the addressed reasons, a mixture of the dimensional and the categorical approach has been developed – mapping the data to eight octants. Labels were not attached to the octants, for it remains an arbitrary problem. This way, interpreting or naming the octants in terms of affective jargon is left to whomever thinks they have the appropriate names. Results are obtained on the testing split that consists of purposely picked and conflated segments.

The precision is the ratio $tp / (tp + fp)$, where tp is the number of true positives, and fp is the number of false positives. The precision is intuitively the ability of the classifier not to label a sample as positive if it is negative.

The recall is the ratio $tp / (tp + fn)$, where tp is the number of true positives, and fn is the number of false negatives. The recall is intuitively the ability of the classifier to find all the positive samples.

The F-1 score can be interpreted as a weighted harmonic mean of the precision and recall, where an F-1 score reaches its best value at 1 and worst score at 0.

The support is the number of occurrences of each class.

Additionally, each speaker's testing file was tested with the `clf.score` function. Results are very similar to the ones obtained with the precision function. The score function computes the percentage of correctly predicted datapoints out of all datapoints used for testing.

Upon features and functionals computation completion some rows contained zeros or NaNs. That is because a recording had unvoiced regions at certain times, which shortly thereafter influenced functionals computation making for gaps (zeros or NaNs) in the final features files. The gaps problem was cured by simply omitting affected rows from the files.

TABLE 4

Speaker <i>i</i>	Duration of testing split	Prediction computation time, tested on PC, without decimation (40 milliseconds)	Score, (clf.score() function)	predominant octants across segments for speaker <i>i</i> :
1	60s	5.49s	.68	5 3 1 0 1 0 0 0
2	57.48s	5.06s	.59	0 1 0 0 5 1 0 3
3	57.48s	5.09s	.7	4 5 0 0 1 0 0 0
4	57.44s	4.97s	.6	3 5 0 0 1 0 0 1
5	87.48s	7.7s	.77	3 3 0 1 3 0 0 0
6	117.44s	10.52s	.86	2 4 2 2 0 0 0 0
7	87.4s	7.73s	.74	4 3 0 3 0 0 0 0
8	57.44s	5.05s	.68	0 3 1 6 0 0 0 0
9	57.44s	5.09s	.71	1 4 0 0 5 0 0 0
10	27.44s	2.09s	.33	0 0 0 0 9 0 1 0
11	117.44s	10.45s	.81	0 3 2 3 2 0 0 0
12	117.44s	9.8s	.75	2 2 0 4 2 0 0 0
13	57.48s	5.09s	.65	6 4 0 0 0 0 0 0
14	57.44s	4.99s	.63	0 3 1 5 1 0 0 0
15	87.44s	5.41s	.79	2 5 0 0 3 0 0 0
16	57.36s	5.08s	.65	5 0 1 3 1 0 0 0
17	57.4s	4.94s	.62	5 1 0 2 1 0 0 1
18	87.48s	7.94s	.75	1 4 0 2 3 0 0 0
19	57.48s	5.09s	.82	6 2 1 1 0 0 0 0
20	87.48s	8.22s	.77	2 4 0 4 0 0 0 0
21	57.48s	5.05s	.68	1 2 0 5 1 0 0 1
22	57.48s	5.31s	.64	2 7 0 1 0 0 0 0
23	57.48s	5.05s	.68	0 7 0 3 0 0 0 0

In **Table 4** accuracy for every testing file, belonging to each speaker, is shown. Fifth column signals to everything written in chapter 7. Database preparation, where it is explained how chunks of different octants are not equally split per recording nor across recordings (speakers). Now it is clear that pervasive unbalance on relatively short recordings could have only been rectified by interfering in the database prior to fitting it to the classifier and testing accuracy. Second column shows that, for most recordings, two segments were assigned to the testing split. These segments are together around 60 seconds long. Results from third column were computed with the sampling period of 40 milliseconds; with a higher period (160 milliseconds) computation times were roughly half as long as the ones from **Table 4**.

As for the time consumed for training the classifier, it took around 170 minutes until all the data was fitted (with cross-validation). The training was conducted at University by courtesy of an assistant who enabled the access to a more powerful computer than one would usually have in personal possession.

There is still features and functionals computation step that has not been discussed. Since the first three recordings are approximately the same duration, their computation times were averaged. It took 7.48 seconds for the openSMILE features computation and 4.02 seconds for the functionals computation in the case with down-sampled files with a sampling period of 160 milliseconds. Original sampling period was 40 milliseconds; it was selected to be congruent with the sampling period from other modalities. (With the original sampling rate, it took 21 sec, on average, to compute the functionals).

To sum up, overall it takes on average 11.5 seconds to compute features and functionals with reduced sampling rate on a computer with 8 GB installed RAM and 3 GHz processor. After we account for the average testing prediction time with the reduced sampling frequency, it adds up to 14.2 seconds. The timing was performed on three one-minute long recordings, for first, second and third speaker, whose duration was determined by the database preparation scheme (See the first three rows in the second column from **Table 4** to learn about their duration). Reader is instructed to consult [14] for comparison with state-of-the-art accuracy achievements for both speaker-dependent and speaker-independent schemes.

9. EXECUTION TIME COMPARISON ON RASPBERRY PI VS. PC

On the Raspberry Pi, it took on average 45.1 seconds for the features computation, and 29.8 seconds for the functionals computation, which together with the average testing time comes to 86 seconds. The evaluation was performed on three one-minute long recordings, corresponding to first, second and third speaker, whose duration was established after the database preparation scheme. For the functionals computation and the classifier testing, reduced sampling rate with a period of 160 milliseconds was used. The features computation was facilitated by the openSMILE toolkit, which extracted them at a rate of 10 milliseconds. In comparison with the performance on the PC mentioned in chapter Accuracy, the difference in the execution times is 71.8 seconds. Put differently, the execution time on PC is almost one-sixth as long as the running time on the Raspberry Pi. Further reduction of the running time could be achieved by additionally lowering the sampling rate or with computation of a fewer number of functionals, which are listed in chapter Functionals. Also, putting less demand on accuracy by calling the PCA function with a smaller input argument, which was in this case set to 90% accuracy, would surely improve computation time. Constraint on the number of computed features is controlled by the PCA function's input argument, therefore, the running time could be enhanced by allowing for the number of features and functionals to shrink below 42 (the dimension of the feature files with PCA(0.90)). The PCA function was also saved to pickle after it was fitted to the training data set, and that is why it reduced the testing data set to the same dimension; it was a precaution that guaranteed the reduced testing data set is not going to have different number or choice of features and functionals. Performance of the two implementations is detailed in **Table 5**.

TABLE 5 THE RESULTS IN THIS TABLE ARE OBTAINED BY AVERAGING COMPUTATION TIMES FOR THE FIRST THREE TESTING SPLITS, EACH OF WHICH ARE AROUND 60 SECONDS LONG.

	Features computation time	Functionals computation time	Average testing prediction time	Sum
PC	7.48	4.02	2.7	14.2
Raspberry Pi	45.1	29.8	11.1	86

9.1. RASPBERRY PI IMPLEMENTATION

To make use of the packages mentioned in Python functions and libraries, they had to be installed with either `pip` or `apt-get` commands in a terminal. The `pip` prefix commands store installation files in `/user/local/pyhon3.5/dist-package/`, and the `apt-get` prefix commands store installation files in `/user/lib/python3/dist-packages/`, unless otherwise instructed. The `pip` commands are used to download and install packages of different kinds and versions, whereas the `apt-get` commands do not provide the option of choosing the version of a package and are considered generally easier to use in the sense that the packages can be updated and upgraded with a simple command that does not require for a programmer to know the up-to-date version number of the package. To install `pip`, this command should be executed: `sudo apt-get install python3-pip`.

The installation of the openSMILE toolkit did not finish quickly and apart from the installation instructions provided in the official manual, some other commands had to be executed. It is important to remark that prior to the openSMILE execution, the path must be changed with the `PATH` command to wherever it was installed. The execution times were measured separately for the functionals computation, the features computation and the testing part of the code. For the testing part of the code, pickled data including the fitted classifier data and the fitted PCA function data were loaded.

10. RASPBERRY PI

The Raspberry Pi is a low cost, credit-card sized computer that has ports for monitor or TV (HDMI), mouse (USB), keyboard (USB), SD card and LAN access. What is more, it has audio, display and camera input, and GPIO bus. It has the ability to interact with the outside world and has been used in a wide array of digital maker projects [16]. Some of which were brought up in [16] are: music machines, parent detectors, weather stations and tweeting birdhouses with infra-red cameras.

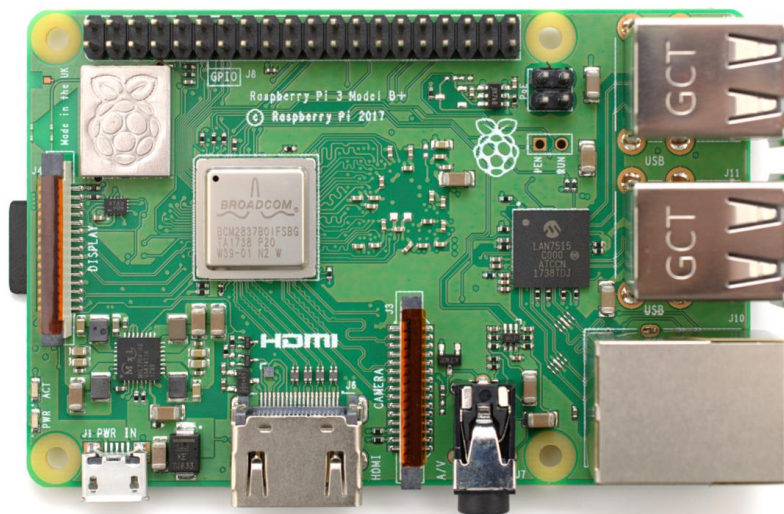


FIGURE 10 THE RASPBERRY PI 3 MODEL B+

TABLE 6

Release date	14 March 2018; 3 months ago
Introductory price	US\$35
<u>Operating system</u>	Raspbian
<u>System-on-chip</u> used	<u>Broadcom</u> BCM2837B0
<u>CPU</u>	1.4 GHz 64/32-bit quad-core <u>ARM Cortex-A53</u>
Memory	1 GB <u>LPDDR2 RAM</u> at 900 MHz
Storage	<u>MicroSDHC</u> slot
Power	1.5 W (average when idle) to 6.7 W (maximum under stress)
Website	raspberrypi.org
Graphics	<u>Broadcom VideoCore</u> IV 300 MHz/400 MHz

Here follows a description from Wikipedia page accessed on 25th of June 2018:

“Raspberry Pi 3 Model B was released in February 2016 with a 64 bit quad core processor, and has on-board WiFi, Bluetooth and USB boot capabilities. On Pi Day 2018 model 3B+ appeared with a faster 1.4 GHz processor and a 3 times faster network based on gigabit ethernet (300 Mbit / s) or 2.4 / 5 GHz dual-band Wi-Fi (100 Mbit / s). Other options are: Power over Ethernet (PoE), USB boot and network boot (an SD card is no longer required). This allows the use of the Pi in hard-to-reach places (possibly without electricity).”

Additional details are available in [17]. In **Table 6** an overview of the Raspberry Pi hardware parts is presented; and **Figure 10** depicts the layout of the used components.

11. CONCLUSION AND FINAL REMARKS

Taking into consideration running time results from chapter Execution time comparison on Raspberry Pi vs. PC, arguing that this scheme could be used in real-time applications on the Raspberry Pi does not seem implausible, however, the running time is still above the real-time margin of 60 seconds. The timing results were obtained on averagely 60 seconds long recordings. There are options one should ponder that could make the algorithm faster, such as accepting lower accuracy, and sampling data with a lower rate. Demand on accuracy can be controlled by changing the input argument of the PCA function, which can lower dimensionality at the expense of poorer accuracy. Down-sampling by a factor of four was implemented in this project, making the sample period four times as long as before, i.e., it was increased from 40 milliseconds to 160 milliseconds. As a conclusion regarding the real-time applicability, the author is at liberty to say how it is not an unattainable goal, especially if a newer version of Raspberry embedded board comes out, or if lower accuracy or sampling rate is tolerable. Not only that, an option to write one's own classification related functions and optimise them so that they exceed performance of widely used ones could prove worthy of effort, but it would be beyond the scope of this thesis. Further, there are not many more Python implemented types of classifier functions that have not been utilised in this project. Let us conclude that on this board real-time implementation is a lofty goal, and how the option of picking another embedded platform should be contemplated. That kind of reasoning is solidified by the high computational demands systems like this one pose, thus requiring appropriately powerful embedded platforms.

On the other hand, accuracy here obtained is not unconvincing, quite the contrary, it indicates to a proper usage of available Python tools, a well-informed database preparation, and a correct way of tuning the parameters of the classifier. This project was constrained by the database's size that entailed a need for interference in its emotive content distribution. Furthermore, embedded platforms' computing power is a restraint for implementation of complex algorithms.

12. SUMMARY

Utilising emotion information through human-computer interactions is associated with practical applications that are beginning to pervade our every-day life. However, the challenges systems which aim to classify emotive states pose are many. Starting from the fuzziness of emotional boundaries to the computational demand of widely used algorithms and classifier functions. In this project an embedded platform was used for an affect recognition task which included computing features and functionals, database preparation, fitting data to a classifier, an evaluation of accuracy and timing the embedded board implemented algorithms. Since computation time is a critical factor in embedded platforms' performance evaluation, the importance of timing the algorithms was emphasised. The tasks of this project were facilitated by making use of the openSMILE toolkit and Python functions for computing the features and implementing the classifier respectively. Following the assessment of different classifiers under diverse parametrisation, the prepared training data set was fitted to the SVM classifier and prediction evaluation was performed in terms of accuracy and execution time, both of which were compared with embedded platform's results. Accuracy obtained is comparable to other works. The running time on the Raspberry Pi shows improvement as we decrease the sampling rate, but it is above the acceptable margin for real-time applications. The high accuracy is partially indebted to the meticulous database preparation, which ensured that the two data sets were evenly filled with emotive content from different octants. The used database contains 23 recordings, each from a different speaker; it is described in detail in chapter Database. Different categorisation approaches have been appraised in chapter Emotion annotation schemes. The annotated data was mapped into eight octants; each octant's accuracy expressed with various metrics is discussed in chapter Accuracy.

Key words: machine learning, Python, SVM, emotion recognition, affective computing, affect, emotive state, Raspberry Pi

13. SAŽETAK

Korištenje emocija u interakciji čovjeka sa računalom je povezano sa praktičnim aplikacijama koje su postale svakodnevnica. Unatoč tome, izazovi koje sustavi koji imaju cilj klasificirati emocionalna stanja predstavljaju su veliki. Počevši od neodređenosti emocionalnih granica do računske zahtjevnosti često upotrebljavanih algoritama i funkcija koje implementiraju klasifikatore. U ovom projektu je korišteno ugradbeno računalo za potrebe prepoznavanja afektivnih stanja koje uključuju: računanje značajki i statističkih mjera, prilagodbu baze, treniranje klasifikatora, evaluaciju točnosti i mjerenje vremena izvođenja implementiranih algoritama. Zadaće ovog projekta su izvršene korištenjem openSMILE alata za računanje značajki i Python funkcija za implementiranje klasifikatora. Nakon što je procijenjena upotrebljivost više klasifikatora uz razne parametre, SVM klasifikator je treniran sa skupom za treniranje i komparativna evaluacija u vidu vremena izvođenja i točnosti je izvršena. Točnost je usporediva sa ostalim radovima. Vrijeme izvođenja na Raspberry Pi pokazuje poboljšanja sa smanjenjem frekvencije uzorkovanja, ali ne zadovoljava marginu za izvođenje u stvarnom vremenu. Zadovoljavajuća točnost je djelom posljedica pažljive prilagodbe baze koja osigurava da su afektivna stanja približno jednako zastupljena u skupu za treniranje i testiranje.

strojno učenje, Python, SVM, prepoznavanje emocija, afektivno računarstvo, afekt, emocionalno stanje, Raspberry Pi

BIBLIOGRAPHY

- [1] H. Gunes and M. Pantic, "automatic, dimensional and continuous emotion recognition," *International Journal of Synthetic Emotions*, vol. 1, no. 1.
- [2] D. Lalanne, S. Scherer, G. Stratou, R. Cowie, M. Pantic and M. Valstar, "AVEC 2016 - Depression, Mood, and Emotion Recognition Workshop and Challenge," 2016. [Online]. Available: https://www.researchgate.net/publication/301896305_AVEC_2016_-_Depression_Mood_and_Emotion_Recognition_Workshop_and_Challenge?enrichId=rgreq-f781b1e53218457483d6dfeac7811416-XXX&enrichSource=Y292ZXJQYWdlOzMwMTg5NjMwNTtBUzozNjgxNDM5MTE0MDc2MTZAMTQ2NDc4. [Accessed 14 6 2018].
- [3] F. Eyben, K. R. Scherer, W. Schuller, J. Sundberg, E. Andre, C. Busso, L. Devillers, P. Laukka, S. Narayanan and K. Truong, "The Geneva Minimalistic Acoustic Parameter Set (GeMAPS) for Voice Research and Affective Computing," *IEEE TRANSACTIONS ON AFFECTIVE COMPUTING*, vol. 7, no. 2, 2016.
- [4] M. Goudbeek and K. R. Scherer, "Beyond arousal: Valence and potency/control cues in the vocal expression of emotion," *J. Acoust. Soc. Amer.*, vol. 128, pp. 1322-1366, 2010.
- [5] K. R. Scherer, "Vocal communication of emotion: A review of research paradigms," *Speech Commun.*, vol. 40, pp. 227-256, 2003.
- [6] D. Bone, C. C. Lee and S. Narayanan, "Robust unsupervised arousal rating: A rule based framework with knowledge-inspired vocal features," *IEEE Trans. Affective Comput.*, vol. 5, no. 2, pp. 201-213, 2014.
- [7] K. Hammerschmidt and U. Jurgens, "Acoustic correlates of affective prosody," *J. Voice*, vol. 21, pp. 531-540, 2007.
- [8] A. d. Cheveigné, "FORMANT BANDWIDTH AFFECTS THE IDENTIFICATION OF COMPETING VOWELS," *CNRS - IRCAM, France, and ATR-HIP, Japan*, 1999.
- [9] K. Hisao and O. Kengo, "Role of formant frequencies and bandwidths in speaker perception," *Electronics and Communications in Japan*, 1987.
- [10] B. Schuller, S. Steidl, A. Bataliner, F. Burkhardt, L. Devillers, C. Muller and S. Narayanan, "The INTERSPEECH 2010 Paralinguistic Challenge," 2010.
- [11] B. Schuller, A. Batliner, D. Seppi, S. Steidl, T. Vogt, J. Wagner, L. Devillers, L. Vidrascu, N. Amir, L. Kessous and V. Aharonson, "The relevance of feature type for the automatic classification of emotional user states: Low level descriptors and functionals," *Proc. 8th Annu. Conf. Int. Speech Commun. Assoc.*, pp. 2253-2256, Aug. 2007.
- [12] F. Ringeval, A. Sonderegger, J. Sauer and D. Lalanne, "Introducing the RECOLA multimodal corpus of remote collaborative and affective interactions," in *Proc. of Face & Gestures 2013, 2nd IEEE Inter. Workshop on Emotion Representation, Analysis and Synthesis in Continuous Time and Space (EmoSPACE)*, Shanghai, 2013.
- [13] "Scikit learn," [Online]. Available: scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html. [Accessed 27 6 2018].

- [14] M. Kotti and F. Paterno, "Speaker-independent emotion recognition exploiting a psychologically-inspired binary cascade classification schema," ISTI-CNR, Via G. Moruzzi, Pisa, Italy.
- [15] M. El Ayadi, M. Kamel and F. Karray, "Survey on speech emotion recognition: Features, classification schemes, and databases," *Pattern Recognition*, pp. 572-587, 2011.
- [16] "Raspberry Pi official web-site," [Online]. Available: <https://www.raspberrypi.org>. [Accessed 25 6 2018].
- [17] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Raspberry_Pi. [Accessed 25 6 2018].
- [18] B. Schuller and G. Rigoll, "Recognising Interest in Conversational Speech-Comparing bag of frames and supra-segmental features," *Proc. 10th Annu. Conf. Int. Speech Commun. Assoc.*, pp. 1999-2002, Sep. 2009.
- [19] E. Marchi, A. Batliner, B. Schuller, S. Fridenzon, S. Tal and O.Golan, "Speech, emotion, age, language, task, and typicality: Trying to disentangle performance and feature relevance," *Proc. 1st Int. Workshop Wide Spectrum Social Signal Process. ASE/IEEE Int. Conf. Social Comput.*, pp. 961-968, Sep. 2012.