Person-Dependent and Person-Independent Analysis of Emotion Recognition using Facial Expressions

Enver Bashirov

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Applied Mathematics and Computer Science

Eastern Mediterranean University
July 2019
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Ali Hakan Ulusoy
Acting Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science in Applied Mathematics and Computer Science

Prof. Dr. Nazim Mahmudov
Chair, Department of Mathematics

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science in Applied Mathematics and Computer Science.

Prof. Dr. Hasan Demirel
Supervisor

Examining Committee

1. Prof. Dr. Hasan Demirel

2. Asst. Prof. Dr. Arif Akkeleş

3. Asst. Prof. Dr. Kamil Yurtkan

ABSTRACT

Facial emotion recognition is one of the prospective fields which can have various applications in many different areas. However, there is a huge difference between a personalized and non-personalized emotion recognition. In facial expression analysis, learning process starts with person's facial structure. A person-dependent system will receive person specific features during training which is advantageous compared to a person-independent system. Hence, with the addition of ethnicity, cultural background or gender differences, gathering results on non-personalized system of emotion recognition becomes a challenge.

In this thesis, models for person-dependent and person-independent emotion recognition are proposed. Experiments are carried out using SAVEE and RML facial video databases. Initially, frames and corresponding landmark features are extracted from the videos. *K*-means clustering algorithm is applied to the extracted landmark features in order to get the *k* most significant frames. After representing each video sequence with *k* keyframes, Support Vector Machine classifier is used for the training and testing of the proposed system. Experimental results show that recognition performance of person-dependent model is higher than person-independent model.

Keywords: Machine Learning; Image Analysis; Emotion Recognition; Facial Emotion Recognition; Support Vector Machine

ÖZ

Yüz duygularının tanınması, birçok farklı alanda çeşitli uygulamaları olan potansiyel alanlardan biridir. Bununla birlikte, kişiye bağımlı ve kişiden bağımsız duygu tanıma arasında büyük bir fark vardır. Yüz ifadesi analizinde, öğrenme süreci kişinin yüz yapısı ile başlar. Bireye bağımlı bir sistem, bireyden bağımsız bir sisteme kıyasla bireysel özellikleri eğitim aşamasında tanıyacabileceğinden dolayı daha avantajlı konumdadır. Bu nedenle, etnik köken, kültürel geçmiş veya cinsiyet farklılıklarının eklenmesiyle, kişiselleştirilmemiş duygu tanıma sistemi üzerinde sonuçların toplanması zorlaşır.

Bu araştırmada, kişiye bağımlı ve kişiden bağımsız duygu tanıma modelleri önerilmiştir. Deneyler, SAVEE ve RML yüze ait video veritabanları kullanılarak gerçekleştirilmiştir. İlk olarak, görüntüler ve görüntülerdeki yüzlere karşılık gelen dönüm noktası özellikleri videolardan çıkarılmıştır. K-en önemli kareleri elde etmek için çıkarılan özelliklere K-means kümeleme algoritması uygulanmıştır. Her video dizisini k anahtar kareleriyle temsil ettikten sonra, önerilen sistemin eğitimi ve test edilmesi için Destek Vektör Makinesi sınıflandırıcısı kullanılmıştır. Deneysel sonuçlar, kişiye bağlı modelin tanınma performansının, kişiden bağımsız modelden daha yüksek olduğunu göstermektedir.

Anahtar Kelimeler: Makine Öğrenme; Görüntü Analizi; Duygu Tanıma; Yüz Duygusu Tanıma; Destek Vektör Makinası

To My Family

ACKNOWLEDGMENT

I would first like to express my deepest gratitude to my brilliant thesis supervisor, Prof. Dr. Hasan Demirel for the continuous support of my Masters study and research. His door was always open for all of my questions, he steered me in the right direction whenever he thought I needed it and his guidance helped me in all the time of research and writing of this thesis. I could not imagine having a better supervisor and mentor for my Masters study.

I would also like to thank to the rest of my thesis committee: Asst. Prof. Dr. Kamil Yurtkan and Asst. Prof. Dr. Arif Akkeleş for their encouragement and constructive comments.

My sincere thanks also goes to my friend Noushin Hajarolasvadi for her patience in guiding me throughout this journey and sharing her precious knowledge without hesitation.

Finally, I must represent my profound appreciation to my father Rza for holding my hand during my education career and to my mother Elnara and my sibling Aylin for their continuous moral and emotional support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

k	Constant in *k*-means
K	Constant in *K*-Fold Cross Validation
AU	Action Unit
AUC	Area Under the Curve
BR	Biometric Recognition
CFS	Correlation based Feature Selection
CNN	Convolutional Neural Network
CV	Cross Validation
DF	Degree of Freedom
DT	Decision Tree
ER	Emotion Recognition
FACS	Facial Action Coding System
FCBF	Fast Correlated-Based Filter
FER	Facial Emotion Recognition
FN	False Negative
FP	False Positive
FPR	False Negative Rate
GA	Genetic Algorithm
KMA	*k*-means Algorithm
ILS	Iterated Local Search
LASSO	Least Absolute Shrinkage and Selection Operator
LDA	Linear Discriminant Analysis

ML	Machine Learning
OvA	One versus All
pdf	Probability Density Function
PR	Pattern Recognition
RF	Random Forest
RML	Ryerson Multimedia Lab emotion database
ROC	Receiver Operating Characteristic
SA	Simulated Annealing
SAVEE	Surrey Audio-Visual Expressed Emotion
SS	Sum of Squares
SU	Symmetrical Uncertainty
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
TPR	True Positive Rate

Chapter 1

INTRODUCTION

1.1 Introduction

Modern technology is improving each and every day without our notice. From personal usage to private businesses Biometric Recognition (BR) which is also known as biometrics has become one of the prospective topics of modern technology. Automatic identification of the individual from their physiological and/or behavioral characteristics is the process of BR [1]. Pattern Recognition (PR) system which implements a feature vector based on a persons characteristics to identify the person is another way of describing a biometric system [2]. Usage of biometrics allows the confirmation of one’s identity based on his/her personal traits rather than using external verification tools(such as an ID card).

Humans intuitively use physical characteristics such as face, voice or body language to recognize each other. This ideology is the basis for biometrics as well as Emotion Recognition (ER) which is a field evolving around BR and PR. Some features and relationships are closely related to identifying emotional state, separating them from other states and not recognizing them as an emotional state at all. When it comes to automatic ER simple understanding of the nature of an emotional state is not sufficient. A framework for ER is created when the rigid structure of emotional states, the features required for confirmation of the states and the type of output which will be delivered by the recognizer are combined together [3].

In human interaction, two communication channels have been identified [4] in the form of explicit and implicit messages. Explicit messages are about anything where the speaker is expressing their thoughts in a particular topic. Implicit messages on the other hand are about the speaker itself. The implicit channel has strong association to the speaker's emotions [3]. ER takes the implicit channel as input stream. When these inputs are person's facial features then the process is called Facial Emotion Recognition (FER) [5].

1.2 Facial Expression Analysis

In 1994, J. A. Russel [6] published his article on universality of recognition of emotion which is based on cross-cultural analysis of ER through facial expressions. Later, P. Ekman [7] and C. E. Izard [8] discussed the weaknesses found in Russel's work such as recognition of emotion depends more than just the facial expressions. Due to the opposition, J. A. Russel published another work collecting arguments of others and readjusting his own thesis in 1995 [9]. In this latest work, joint resolution on existence of minimal universality have been met. People, without differentiating their location, can sense others emotions from their facial expressions. Additionally, three crucial points for ER has been established in terms of identifying facial emotions; patterns of facial muscle movements, expressiveness of facial movements and universality of acknowledging same emotion everywhere. Integration of these ideas into a Machine Learning (ML) system is the definition of FER. From physiological point of view, when a human was in an intense emotional situation there was six basic emotions that stood out. Bloch et al. [10] identified that these basic emotions are anger, disgust, fear, happiness, sadness and surprise. Mainly, the research in ER is based either on these 6 basic emotions or with an addition of 7th neutral state [11]–[17]. Bartlett et al. [15] have compared ML methods such as AdaBoost, Support Vector Machine (SVM) and linear discriminant analysis in terms of recognition of facial emotions. They identified that subset selection with AdaBoost followed by SVM classification yielded the best result. Z. Yu and C. Zhang [16] established their FER system by applying facial detection algorithm static images to gather facial features. Then, classification between 7 emotion states using multiple deep convolutional neural networks (CNN) was carried out. Their work proved minimization in log likelihood loss and hinge loss kept recognition rate above average compared to their target baselines. These two studies were based on person-dependent scenarios of FER. A person-independent study was carried out on spontaneous facial expressions in a dynamic environment [17]

From the works explained above, it is clear that there are many applications of FER. Automatic recognition of facial expressions is a challenge in two models of personalized ER. Main aim of this research is to distinguish difficulties between person-dependent and person-independent models of FER. Focus of this thesis is optimizing the two models in feature extraction process to provide a better learning experience for the machine.

1.3 Problem Definition

The challenge of using video clips in ER is to maintaining a healthy learning process. A video consists of vast number of frames and naturally not all of these frames consist of useful information for the classifier. For the purpose of identifying the most discriminant keyframes, clustering algorithms are employed. In a multi-dimensional classification environment it is crucial to ensure that the data consists of only from beneficial information. Multiple studies have been carried out in terms of multi-dimensional FER systems [15], [17]–[19]. Feature extraction and selection process is imperative part of an ML application. A multi-dimensional classification system can be optimized in terms of speed and accuracy rate by introducing improvisation to the learning model. The main problem in such an ER application is choosing right amount of clusters. In this study, a varying number of sets of clustering was performed in order to come up with the best performing cluster to solve the problem of multi-dimensional learning process of person-dependent and person-independent FER.

1.4 Proposed Methodology

Initially, static facial expression images from various databases are gathered. Facial expression detection is carried out by using facial landmarks. These landmarks are located around the face of the person for recognition of facial regions. Feature selection and extraction is proceeded by using landmarks. In order to reduce the cost of processing, clustering algorithm with changing variable is employed. Learning is carried out by using clustered sets of data both in person-dependent and person-independent models. *K*-Fold Cross-Validation technique is also used for artificially increasing the number of test samples found in the system.

1.5 Thesis Contributions

A FER application is a multi-dimensional classification problem. In a non-binary learner system, size of the dataset and hence the speed of learning process becomes an issue. A solution for tackling these issues in a FER application is given by using a *k*-means based dimensionality reduction. First contribution of this study is the use of *k*-means clustering algorithm to reduce each video into keyframes. With this each video is represented by *k* number of keyframes which results in lower dimensional classification data and faster simulation speed. In this study, Cross Validation (CV) technique has been utilized. CV has been used as a way to use datasets uniformly through training and testing without introducing any bias to the system. This also allows artificial enlargement of the dataset. Finally, a score fusion method using score averaging method is introduced for calculation of the prediction results for each video rather than each keyframe which is the final contribution of the study.

1.6 Thesis Overview

This thesis consists of 6 chapters which are arranged as follows: Chapter 1 introduces a short introduction to FER, problem definition, methodology and contributions. Chapter 2 contains recent literature studies about FER. Chapter 3 describes the methodology, FER framework, databases, landmarks, feature descriptors, clustering into keyframes, clustering and CV, sensitivity and specificity, description of accuracy and loss and receiver operating characteristic. Chapter 4 and chapter 5 includes person-dependent and person independent FER models, respectively. Additionally, these chapters describe the implementation details and results of the corresponding model simulations. Finally, conclusion and discussion about available future works are presented in chapter 6.

Chapter 2

LITERATURE REVIEW ON FACIAL EMOTION RECOGNITION

2.1 Introduction

ER is the task of labeling given input samples into emotion categories. From human point of view ER is done automatically however computational methodologies are being developed. ER is a classification problem and is closely related to sub-areas of computer science such as signal processing, machine learning and computer vision. Initial step in a classification problem such as ER is data preparation and feature selection.

Pre-processing is generally a step carried out during or after the data preparation process. Smart implementation of pre-processing techniques can provide better platform for feature extraction. For instance, in FER image pre-processing techniques are divided into four categories; pixel brightness transformations, geometric transformations, pre-processing using local neighborhood of the processed pixel and image restoration methods. These pre-processing techniques have been materialized from the radius of pixel coverage which will be used in new pixel brightness calculation [20]. Multiple exemplary studies performing image resolution enhancement was carried out using discrete and stationary wavelet decomposition

[21]–[23]. Similarly, in other branches of ER such as speech or text based ER, pre-processing techniques can have a crucial role in maintaining a level of consistency in information.

A feature is an individual physical or non-physical property in a given observable [24]. In general, features are divided into two categories, numeric features and structural features. A collection of features is called a feature vector. Feature vectors are the building blocks in any classification problem since all the computation is based on them. Feature extraction is a step where the raw or pre-processed data is transformed into features which represents a pattern for the learning algorithms. In any data representation type (image, audio, text etc.), there are large number of different features that can be used by the system. However, when the number of chosen feature vectors kept high, accuracy of the algorithm will drop causing over-fitting. Similarly, if the number of selected features are low then the system will fail on capturing the underlying trend of the data which will cause under-fitting [25]This is particularly why the most appropriate features must be selected in a feature selection step.

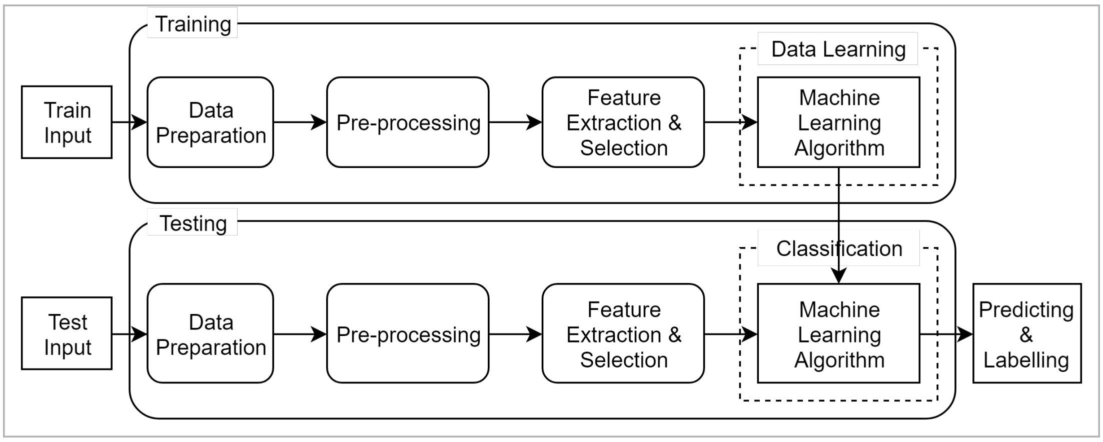


Figure 2.1: Generalized flowchart of a machine learning classifier system.

Figure 2.1 illustrates a block diagram of a classification system. It can be observed from the diagram that the feature extraction and selection will effect both training and testing phases of classification process. The designing of the algorithm is the next step following feature selection process. During the training phase, designated algorithm will generate a predicting model which afterwards will be used in the testing phase for classification of the data.

2.2 Applications on Facial Expressions

There are large number of studies carried out in FER containing many different approaches [26]–[32]. For instance, one of the work considered dominant and complementary emotion recognition using head-pose estimation and micro emotions as features [33]. Another study was carried out in 3D convolutional neural networking platform to analyze speech emotion recognition. In this study, authors have using K-Means clustering algorithm and spectograms to improve their recognition results [34]. Similarly, Kwon et al. [35] performed emotion recognition using speech signals. They have divided their work into three stages; feature extraction, feature selection and classification. Their work concluded that on variable-length utterances, classification using Gaussian SVM and for short utterances, HMM-based classification did perform the best. Face and facial expression recognition are closely related to FER. In 2014, K. Yurtkan and H. Demirel studies entropy-based feature selection for improving 3D facial expression recognition [36]. In their study, they were able to outperform the latest feature selection methods in the literature. Some of the studies also consider using different classification techniques in order to improve their results. In a study of facial expression recognition, different Bayesian network classifiers were introduced for classification of image sequences from videos [37].

2.3 Feature Selection Methods

Feature extraction and selection has become an exacting step for machine learning applications. Considering the volume of the database and vast number of features present, feature selection gives the opportunity to optimize the pre-learning stage of the application. If work will be done on huge quantities of data, one should be able to differentiate between irrelevant and redundant information than the useful and important ones [38]. At this stage, the importance of identification of useful features comes up.

Claude Shannon's information theory provides a path for measuring the relativity of the information gathered from a random variable compared to the output of the application. This is introduced with concepts of entropy and mutual information. The term entropy stands for the amount of uncertainty a random variable carries. Formula for this uncertainty or in other words the entropy is

$$H(X)=-\sum_{x\in X'} p(x)\log(p(x)) \qquad (2.1)$$

where *X* has *X'* alphabets and is the probability density function (pdf). The log is of base 2 and the entropy unit is in bits. If another discrete random variable was introduces as *Y,* than the joint pdf of *X* and *Y* is $p(x,y)$ and the joint entropy function of *X* and *Y* is defined as

$$H(X,Y)=-\sum_{x\in X'}\sum_{y\in Y'} p(x,y)\log(p(x,y))\,. \qquad (2.2)$$

This formulation leads to the information shared between two random variables which has a great importance in identification of the reliable features. Below formula is defined as the mutual information between two variables

$$I(X;Y)=\sum_{x\in X'}\sum_{y\in Y'} p(x,y)\log(\frac{p(x,y)}{p(x)p(y)})\,. \qquad (2.3)$$

As mutual information between two random variables gets close to zero, the variables are concluded as independent from each other. However, if the mutual information of variables is large, than this proves a greater relativity between them [39], [40].

This is a common theory behind most of the feature selection methods. Commonly used feature selection methods in machine learning divide into three categories, filter methods, wrapper methods and embedded methods. Figure 2.2 illustrate the three selection methods.

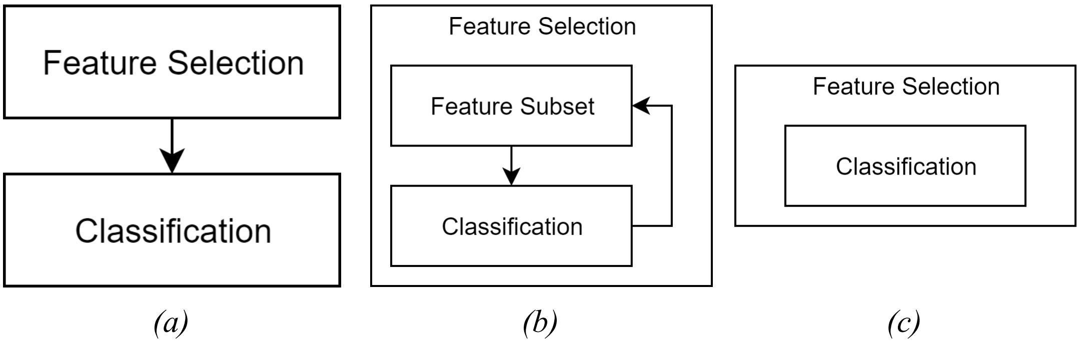


Figure 2.2: Filter (a), Wrapper (b) and Embedded (c) feature selection block diagram.

Filter methods perform selection process with no connection to classification model while wrapper methods work inline with classification model and check the prediction data to decide the faith of features. On the other hand, feature selection process is integrated to the classification model in embedded methods. Below subsections describes the three methods for feature selection and discuss some of the underlying algorithms.

2.3.1 Filter Methods

Pre-processing is generally where filter methods are utilized. Filter feature selection techniques bind a score to each feature by using statistical measurements. The features are than ranked according to the scores accordance to their relationship with the outcome variable. Final rankings decide whether a feature is to be kept or removed from the dataset. RELIEF, correlation based feature selection, fast correlated based filter, INTERACT, Chi-Square, linear discriminant analysis and ANOVA are some of the filter feature selection methods that are discussed below.

2.3.1.1 RELIEF

Kira et al. [41]describes original RELIEF algorithm as a way to estimate the relative quality of two feature. This is done with the idea of how one instance stands out in comparison to the others that are close to each other. RELIEF looks for the nearest two neighbors from a randomly selected instance. If the randomly selected instances are given as $x_i = \{x'_{1i}, x'_{2i}, \ldots, x'_{ni}\}$, than the neighbor which is from the same class is called the nearest hit *H* and the neighbor which is from a different class is called nearest miss *M*. It then continuously works the relative quality of each feature with respect to x_i, *M* and *H*. This original RELIEF algorithm can work on both continuous features and discrete features however, it is limited to two-class properties. Later, an enhanced algorithm was invented with a new name called ReliefF [42]. ReliefF was a robust algorithm which had the capability of working on noisy and incomplete data. At this stage limitations for multi-class problems were also removed. RReliefF algorithm was the final and third version which was adapted to work on regression type problems as well as having the previous capabilities of its ancestors [43]. Relief algorithms are generally preferable due to the way they communicate with features and their low bias.

2.3.1.2 Correlation Based Feature Selection

Correlation based Feature Selection (CFS) ranks the subsets considering their correlation to the classification. If a feature subset has high correlation with the classification then it must be recognized as related feature subsets and yet must not related to the others. CFS employs a correlation based heuristic evaluation function. Due to their low correlation to the class, unrelated features will be ignored while features that are highly correlated to the remaining ones are marked as redundant and are discarded. Ranking of a feature is determined by to which extent predicted classes are in the area that is not predicted for other features.

$$M_S = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}} \tag{2.4}$$

Above formula describes how heuristically merit, M_s, a feature subset consisting of k features is. $\overline{r_{cf}}$ is the mean value of correlation between all feature and classifications and $\overline{r_{ff}}$ is the mean value for correlation between all features. The amount of redundancy there is between the features is the denominator and the amount of how predictive a feature subset is of the class is the nominator in this equation [44], [45].

2.3.1.3 Fast Correlated Based Filter

Symmetrical uncertainty (SU) is the proportion of information gain and entropy between two features, x and y.

$$SU(x,y)=2\frac{IG(x/y)}{H(x)+H(y)}, \tag{2.5}$$

where $H(x)$ and $H(y)$ are the entropy and IG is the information gain described as:

$$IG(x/y)=H(y)+H(x)-H(x,y), \tag{2.6}$$

where $H(x,y)$ is the joint entropy.

The fast correlated-based filter (FCBF) is based on SU. This method is suitable for multi-dimensional data which has been proven to be efficient at filtering of redundant and unwanted features. But it cannot deal with interactions between features.

2.3.1.4 INTERACT

In terms of ranking measurements, INTERACT algorithm is similar to FCBF filter however INTERACT also contains consistency contribution (c-contribution). This new parameter decides how impactful the discarding of a feature could impact the stability. This algorithm has two significant parts. Initially, ranking of features is carried on according to their SU values in descending order. Then, in the second part, evaluation is done starting from the bottom of the ranking list of the features. When the c-contribution is less than a predetermined value, feature is removed, otherwise, feature is added to the pool. Zhao et al. [46] mentions that this algorithm is capable of dealing with feature interaction and effectively choose desired features with the newly developed method.

2.3.1.5 Chi Squared

Chi-square is a method commonly found in testing of relationships between categorical variables. When chi-square test results in null value, the result is that there is no existing relationship for the categorical variables in the current context, in other words, variables are independent. The scoring for chi-square test is given by:

$$X^2=\sum\frac{(f_0-f_e)^2}{f_e}, \tag{2.7}$$

where f_0 is the number of occurrences of the feature in the set and f_e is the expected number of occurrences without any relationship to the variables [47]. It can be observed from the formula that chi-square method is based on ratio of the frequency of actual observations of the variable and the expectation of no true relationship between variables.

2.3.1.6 ANOVA

ANOVA is shorthand for analysis of variance and is closely related to LDA where a variable is tried to be expressed by linear combination of other features. Test of ANOVA is carried out by computation of means and variances and comparing the ratios of two variances to the previously chosen value to get a statistical significance. Definition of variance is given by:

$$s^2 = \frac{1}{(n-1)} \sum (y_i - \bar{y})^2 , \tag{2.8}$$

where *(n-1)* is the degree of freedom (DF) and $\sum (y_i - \bar{y})^2$ is called the sum of squares (SS). ANOVA computes a total variance, an error variance and a treatment variance. Crucial step in the algorithm is to partition the SS and DF into components. In a model for simplified version of ANOVA following approaches are used:

$$SS_{Total} = SS_{Error} + SS_{Treatments} , \tag{2.9}$$

$$DF_{Total} = DF_{Error} + DF_{Treatments} . \tag{2.10}$$

Next, f-test is used for the comparison of total deviations. For instance, in a single-factor ANOVA, statistical factors are tested by using test statistic, *F*, as:

$$F = \frac{variance\,between\,treatments}{variance\,within\,treatments} = \frac{MS_{Treatments}}{MS_{Error}} = \frac{SS_{Treatments}/(I-1)}{SS_{Error}/(n_T - I)} , \tag{2.11}$$

where *MS* stands for mean square, *I* is the treatment count and n_T is the total amount of cases. ANOVA F-test is optimal for lowering false negative faults. This algorithm can be used for single factor or multiple factors [48], [49].

2.3.1.7 Linear Discriminant Analysis

When there are more than two classes involved in a classification problem, Linear Discriminant Analysis (LDA) is the preferable linear classification algorithm. LDA computes statistical information of the data for each present class. For instance, in a single input variable, the result is a single mean and variance value for each class. In a multi variable scenario, similar calculation is carried out over the means and covariance matrix. LDA assumes that the given data is Gaussian or in other words when plotted, the graph looks like a bell curve. LDA also assumes that each sample has the same variance and on average, each variable is similarly close to the mean.

Algorithm, with these assumptions estimates a probability for a new subset of inputs of each class predicting the highest probability class as the output class [50].

2.3.2 Wrapper Methods

Wrapper methods deal with the feature selection as a search problem. Idea behind these types of methods is to look in the space of feature subsets by utilizing a learning algorithm in order to guide the search. For each feature, an estimate of accuracy is calculated on the learning algorithm. This accuracy is gathered from a cross-validation process over the training dataset. Then a comparison between the estimated accuracy on current feature and selected feature subset is carried on. If current feature is estimated to be lower than the selected features, then the feature is dropped [51]. This is the main idea behind a wrapper method. Some examples to wrapper feature selection algorithms are Genetic Algorithm, Simulated Annealing and Iterated Local Search for which the explanations can be found below.

2.3.2.1 Genetic Algorithm

Genetic Algorithm (GA) is developed from the concept of evolution and it is a search algorithm. Initially, a collection of feature subsets are formed. This collection contains randomly selected k_1 feature subsets out of total n feature subsets. For each feature subset in this collection, their fitness level is computed. Fitness level describes how fit a feature subset is compared to the other. With the help of fitness, subsets are ranked against each other resulting in an ordered list of feature subsets. Subsets with better fitness level are chosen for crossover and mutation procedures.

```
Algorithm: Genetic Algorithm
Begin:
    Let: S = { collection of initial feature subset population }
    F = ComputeFitness( S )
    While population is no more converging
        S = SelectBest( S, F )
        S = Crossover( S )
        S = Mutate( S )
        F = ComputeFitness( S )
End;
```

Crossover is a process of combining two features (parents) to produce single feature (child). Crossover process results in a new population. However, a child may result in a useless generation considering the evolution of the population. For tackling this problem, parents could be altered slightly so that their offspring is no longer useless. This is the process of mutation. Finally, child subsets from these procedures combined with better feature subsets from the population and make a new population. This procedure is carried on iteratively, usually resulting in a lower number of population than the previous iterations until a breaking point is hit which results in a collection of best feature subsets for feature selection [52], [53].

2.3.2.2 Simulated Annealing

Simulated Annealing (SA) similar to GA is a stochastic search algorithm. Essentially, SA creates non-homogeneous Markov chain from an objective function which moves towards the optimal point of the function. Algorithm starts from an initial starting string S_0 assuming a set of binary strings S such that $S=\{(S_1,\ldots,S_n)' \mid S_i\in\{0,1\}\}$. S_i is used to take a random neighboring string which is denoted as S^*_{i+1}. The algorithm is as follows:

```
Algorithm: Simulated Annealing
Begin:
```

```
Let: $S=\{(S_1,\dots,S_n)' \mid S_i \in \{0,1\}\}$
For each S
    Let: $S^*_{i+1}$ = random neighbor
    if $\hat{F}(S_i) > \hat{F}(S^*_{i+1})$
        then $S_{i+1} = S^*_{i+1}$
    else
        Let: c = random(0, 1)
        if $c \geq e^a$ where $a = -(\hat{F}(S^*_{(i+1)}) - \hat{F}(S_i)) / T_i$ then
            $S_{i+1} = S^*_{i+1}$
        else ( $c < e^a$ )
            $S_{i+1} = S_i$
End;
```

In the above algorithm [54], $\hat{F}()$ is the objective function and *T* stands for the hypothetical number of steps or it can be seen as a timer function which is decreasing. As the algorithm proceeds, *T* gets smaller and it becomes harder for $\hat{F}()$ to choose a point that does not get smaller. Finally, when $\hat{F}()$ is not getting a useful result, the algorithm stops [54], [55].

2.3.2.3 Iterated Local Search

Iterated Local Search (ILS) deals with optimization of discrete problems. In local search functions, there could be point where a local minimum is stuck if there are no neighbors to be improved. A change is introduced in the name of iterating calls to the searching function where the algorithm looks for a different starting point on each iteration. Using the same terminology from SA, below is the algorithm for ILS.

Algorithm: Iterated Local Search

```
Begin:
    Let: $S_0$ = { initially generated solutions set }
    $S^*$ = LocalSearch( $S_0$ )
```

```
  While termination condition is not met
      \* Perturbation *\
      if $S^*$ is weak compared to previous result
          then set back the local minimum
      else if $S^*$ is strong compared to previous result
          then set randomize the local minimum and restart
      \* Perturbation *\
      $S^{*'}$ = LocalSearch( $S'$ )
      $S^*$ = CheckAcceptance( $S^*$, $S^{*'}$, previous_results )
End;
```

With the perturbation step, the algorithm avoids imprisonment and restarts from another random point. If there is no imprisonment, then the corresponding local search function is run and acceptance rules are checked similar to a non-iterative local search [56].

2.3.3 Embedded Methods

Since filter methods are designed for use before learning and wrapper methods do not take structure of classification or regression functions into the consideration, a bridge methodology was introduced. Compared to the other two, embedded methods do not separate the pre-learning and learning part of the process. In a sense, embedded feature selection methods connect filter and wrapper methods [57]. A common embedded method is Least Absolute Shrinkage and Selection Operator (LASSO).

2.3.3.1 Least Absolute Shrinkage and Selection Operator

The process of finding affinity between variables is called regression analysis. This is highly related to machine learning topics such as supervised learning, clustering and classification [24]. LASSO was introduced by Robert Tibshirani in 1996. This method uses fixed upper bound value as a constraint to the sum of absolute values of the model parameters. To do this, algorithm uses a regulatory function which as a result transforms some of the vectors of regression to zero. As a result, variables that have non-zero coefficients will survive the feature selection process. The aim of the LASSO regularization process is to optimize the accuracy of the predictions [58], [59].

2.4 Learning and Classification Algorithms

In machine learning, supervised learning is a term used for the process of generating a mapping between input and output pairs using sample input-output relationships [60]. Specifically, these input-output relationships are group of training examples in a labeled data form. These data are the features gathered from the feature selection and extraction phase. There are also other learning algorithms which do not take labels as precursor, unsupervised and reinforcement learning [61]. Unsupervised learning partitions the features into groups by finding previously unknown patterns. On the other hand, reinforcement learning makes use of some external information provided by trainer. This given information is in the form of scalar reinforcement signals which indicates how well the system is operating. Learner tries each action one by one to find the best outcome. The problem of classification is considered to be a part of supervised learning area. A classification model tries to solve the problem of predicting the category of an observed information by using generated set of rules from the training algorithm and applying these rules to an observation in the light of finding the most appropriate category [62].

In the field of ER, labeling guides the classification process. Without this crucial information there would be no exact way to know which set of features belong to what emotion. As humans, our cognitive consciousness and instincts let us differentiate different emotions. However, this is not the case for machines. Thus, supervised learning algorithms are the most appropriate choice when compared to a human consciousness. Not to mention that not just in ER but also in general, majority of practice in ML is carried out using supervised learning algorithms. Some of the most commonly seen algorithms for supervised machine learning are given below.

2.4.1 Decision Tree

Decision Tree (DT) is a collection of binary classification tools. A DT sculpts a training model using a prediction class or a given set of learning rules. The algorithm uses tree structure as the base data structure model. Each non-leaf node represents a question where the answers are in binary format. After answering a question, either another non-leaf node with a different question is encountered or a leaf node with a final prediction for the leaf to root path is made [63]. Following is the DT algorithm.

Algorithm: Decision Tree

```
Begin:
    Let: F = { set of features }
    T = { root of the tree with best applicable feature }
    While termination condition has not met
        Split(T)
        LabelLeafNodes(T, S)
        Predict(T)
End;
```

Split function, splits the leaf nodes that are labeled for splitting as the name suggests. Then the leaf nodes are labeled either for more splitting or for predicting a result. Predict function gives a resulting prediction for the leaf node according to the corresponding labeling. Finally, the function is terminated when none of the leaf nodes contain a split command and all leaf nodes contains a prediction [64].

2.4.2 Random Forest

When multiple DTs are incorporated together at training phase a Random Forest (RF) learning method is formed. The final prediction set of RF is the overall outcome taken from the incorporated trees. In the work of Kleinberg [65], it has been stated an ideal mathematical analysis is required to tackle the over-fitting in a PR problem. Due to the construction of how the prediction model is created in RF the problem of over-fitting has been overcome.

Algorithm: Random Forest

```
Begin:
    Let: $F$ = { set of features }
    For i to n
        $R$ = { k number of random features from F }
        $T_i$ = DecisionTree( R )
    $P$ = FinalizePredictions( T ) /* calculating prediction target from all trees
        according to the combining rule*/
End;
```

The algorithm takes *k* random number of the features from the initial feature set and employs DT algorithm to create *n* number of trees. After the designated number is reached, RF proceeds to create a final tree composed of predictions. For a single prediction target, votes are counted for different results and the prediction with the highest vote is counted as final prediction. The final tree is formed from these final predictions. When a test data is given, final tree in RF predicts the result. In a sense, by passing through this final tree, test data has been tested on all iterations of DTs [65], [66].

2.4.3 Support Vector Machine

SVM tries to compute the optimal hyperplane for the given labeled training data. This output hyperplane defines the fine line between categories. When the space is in two dimensions, the hyperplane becomes a line dividing the plane into two parts which results in binary classification model. The separation of classes is demonstrated in the figure 2.3:

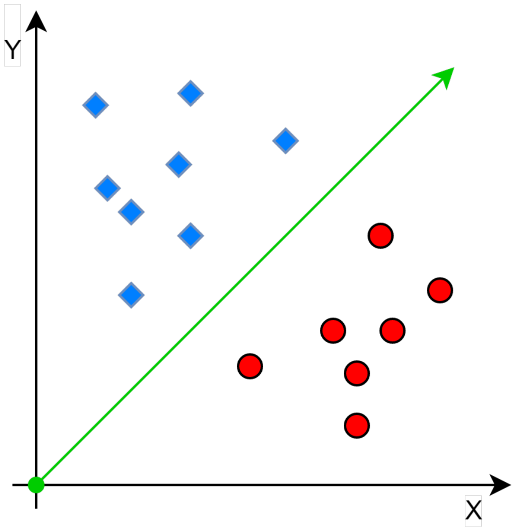


Figure 2.3: Linear division of two classes in SVM [83].

Thus, on a plane involving only *x and y* dimensions, given data is separated into two classes with a line. However, when given data is multi-dimensional, a simple straight line will not be sufficient. Lets assume that in addition to *x* and *y* axes, we also have *z*-axis. Demonstration for the three dimensional plane is given in the figures 2.4 and 2.5.

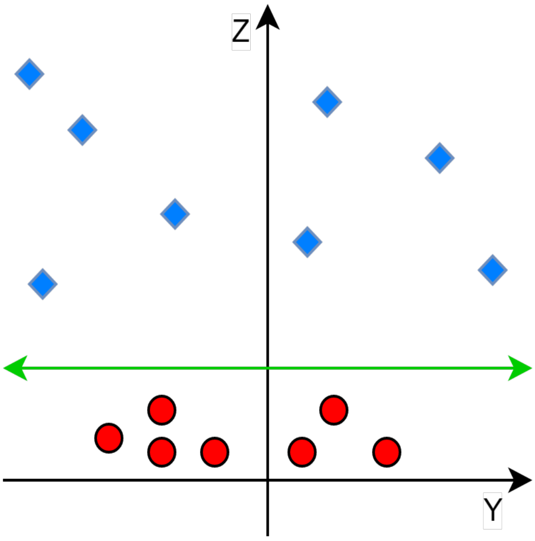


Figure 2.4: 3D Class separation on the y and z axes [67].

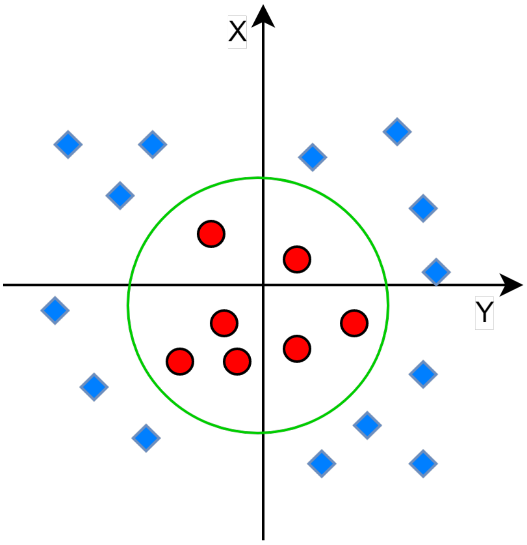


Figure 2.5: 3D Class separation on the x and y axes [67].

In this case, a circular line is required for the separation. These transformations in the SVM are done by kernels. Kernel function calculates inner products between given input vectors. This way of computation allows kernels to work in multi-dimensional plane without actually performing calculations on the coordinate plane itself. This bypassing of mapping approach is called the kernel trick [67]. A kernel could employ linear, Gaussian, polynomial or any given type of formulation. A kernel function is denoted as:

$$K(x,y)=<\Phi(v_1),\Phi(v_2)>, \tag{2.12}$$

which is the inner product of input vectors. Using this description, linear, polynomial and Gaussian kernels are defined respectively as:

$$K(v_1,v_2)=v_1^T v_2+c, \tag{2.13}$$

$$K(v_1,v_2)=(av_1^T v_2+c)^d, \tag{2.14}$$

$$K(v_1,v_2)=\mathrm{e}^{\left(\frac{-\|v_1-v_2\|^2}{2\sigma^2}\right)}. \tag{2.15}$$

Simplest kernel function is the linear kernel where there is only an additional constant c is involved. Polynomial kernel functions are suitable for normalized training data since there is an addition of gradient a and polynomial degree d. Third kernel function, Gaussian, has adjustable parameter of σ. This parameter must carefully tuned. In case of underestimation, the function will be highly sensitive to noisy data and in case of overestimation, the function will act as a linear function [67]–[69].

Without discrimination, on any kernel function, there will be the problem of overlapping of data between classes. Margins determine the level of separability between the classes. A hyperplane can be written in the form $wv-b=0$ and the margin of distance between the two hyperplanes becomes $2\ /\ \|w\|$. In order to maximize the distance between the hyperplanes, $\|w\|$ must be minimized. This is demonstrated in the following figure with perpendicular lines to the separation of classes.

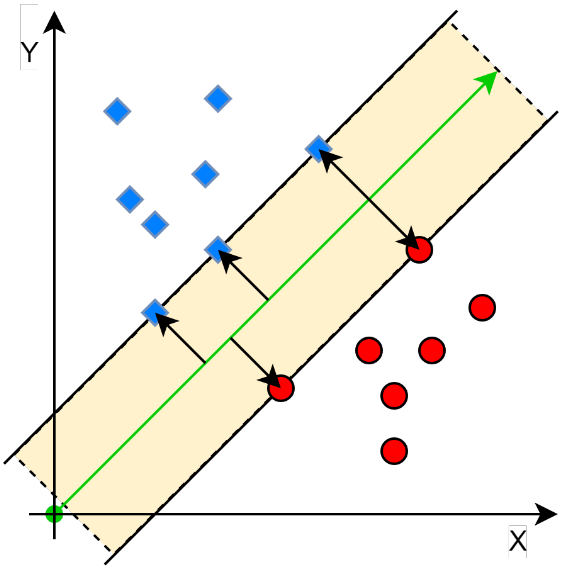


Figure 2.6: Separation of classes with margin [67].

Observing from the figure 2.6, SVM maximizes the distance between green separator line and the data points as much as possible with equal margin, black arrow lines. Final hyperplane is drawn when maximum number of data points are touching to the margin lines. Finally, there is the concept of minimum and maximum margins. This introduces breathing space for margins. Data points are not expected fall exactly on the tip of the arrow but rather could fall into a range of distance between minimum and maximum margins. This approach is used for optimizing the hyperplane position. Lastly, in each iteration of the learning process, SVM uses loss function to improve predictions in the next steps. Loss function is an error calculation method. The amount of difference between classifiers prediction score and actual result is the error interval. By using this error interval, classifier adjusts the values of margin for the next iteration. This is continued until there is no longer any minimization on total error.

Chapter 3

FACIAL EMOTION RECOGNITION METHODOLOGY

3.1 Facial Emotion Recognition Framework

A FER application consists of pre-processing, feature extraction and classification phases. Initially, static images are gathered either directly from databases or by extracting from video clips. Then, feature selection and extraction is carried out. Finally, clustering and classification is done. Pre-processing is a non-compulsory step which is usually carried out either before or during feature extraction phase.

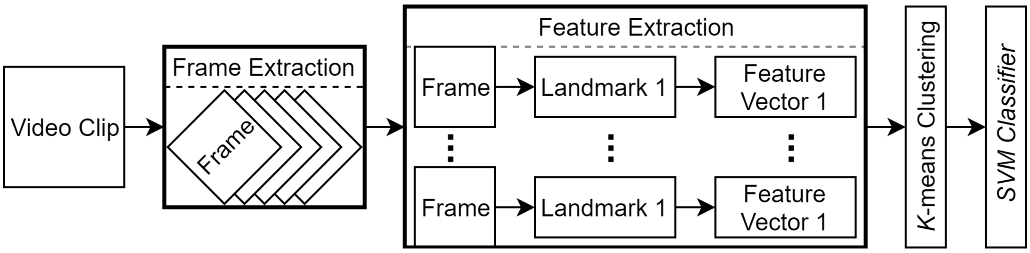


Figure 3.1: FER learning system.

Generalized flowchart of a FER application is included in figure 3.1. Clustering is a step which reduces the size of the dataset while containing valuable information required during the learning process. During classification, smaller sized data sets are preferable since classification is highly time consuming process and this is directly proportional to the amount of data at hand. Also, in a sense clustering filters the unnecessary information which may lead to redundancy in learning.

3.2 Facial Databases

For a FER system to work accurately, databases that are wealthy in terms of features and that are applicable in the context must be chosen. Surrey Audio-Visual Expressed Emotion (SAVEE) database [70] and Ryersan Multimedia Lab emotion database (RML) [71] are highly suitable for FER experiemnts. SAVEE and RML are databases consisting of various number of videos for each basic emotion category. Each database had different number of subjects demonstrating a role of an emotion. Videos are structured such that the face of the person who is performing the corresponding emotion category was captured. In addition to visual act, audio is also a part of these videos where in some cases actor demonstrates the emotion using a sentence.

In SAVEE there are 480 total number of videos and each video has 60 frames per second while in RML there are 720 samples each captured with a frame rate of 30 frames per second. For each dataset, number of actors for each subject is varying. Table 3.1 and 3.2 illustrates number of samples found on each category from each subject.

Table 3.1: Subject distribution per emotion category in SAVEE

Subject	Angry	Disgust	Fear	Happiness	Sadness	Surprise
1	15	15	15	15	15	15
2	15	15	15	15	15	15
3	15	15	15	15	15	15
4	15	15	15	15	15	15

Table 3.2: Subject distribution per emotion category in RML

Subject	Angry	Disgust	Fear	Happiness	Sadness	Surprise
1	3	4	5	3	5	5
2	3	5	4	4	5	4
3	8	16	11	18	15	12
4	33	25	31	23	29	31
5	19	17	16	25	13	15
6	16	15	17	18	17	17
7	24	23	23	17	24	22
8	14	15	13	12	12	14

3.3 Landmark Detection

Facial Action Code System (FACS) is an observer-based system for measuring facial expressions [72]. With the help of this system, visually identifiable facial movements which are action units (AU) can be decomposed [73]. Thus with the help of AUs, a machine learning algorithm can differentiate fairly specific facial behaviors. The first internationally accepted standard for multimedia communication is MPEG-4 [74]. MPEG-4 is a standard introduced for group of audio, video and 3d graphics in order to identify a synthetic or natural head and body animations. Facial expressions are the source communication channel for human interaction. The study of automatic recognition of facial motions is called facial expression analysis [75]. MPEG-4 has been the basis for work in facial expression analysis. Crucial step in identifying the facial features is to detect feature points such as corner of the eyes or peak points of lips. These points located on the surface of the face are called facial landmarks. Landmarks are the basis for detection of facial features. OpenFace [76] is a framework used for facial expression analysis. This framework can pick out frames from the videos while extracting facial landmarks.

OpenFace calculates landmark points that are located strategically on 68 different points of the human face. This is based on Locations of each landmark allows the machine to portray facial structure of the face in vector form. In the figure 3.2, blue dots represent the landmarks. The areas covered by the landmarks are left eyebrow, right eyebrow, left eye, right eye, upper mouth, lower mouth, nose and chin.

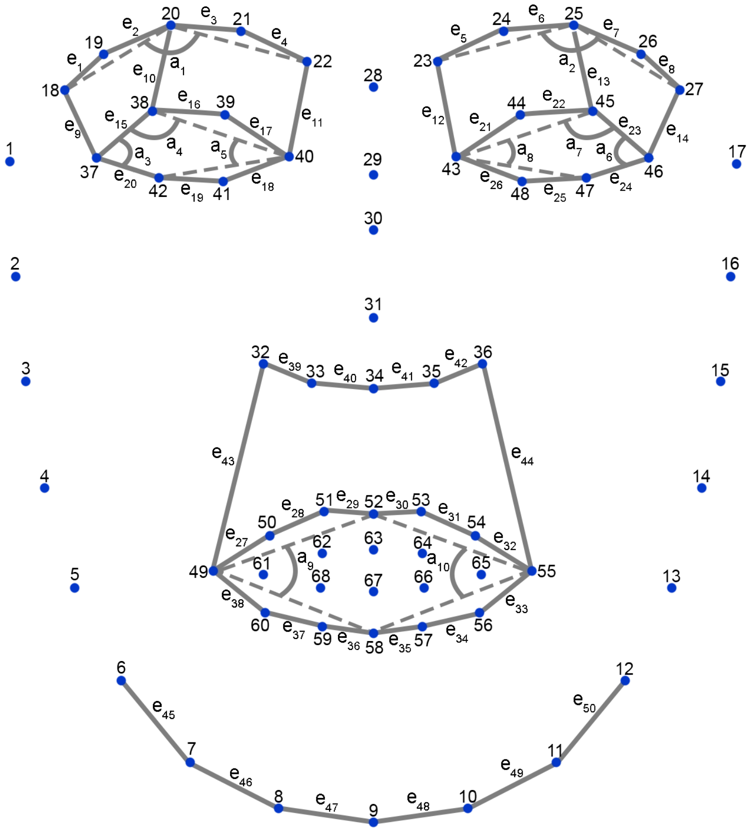


Figure 3.2: Facial landmarks, distance and angle descriptors [76].

3.4 Distance and Angle Descriptors

In a geometric based facial feature selection, distance and angle geometric descriptors are used. This approach is seen on the work of Noroozi et al. [77]. Visualization of distances and angles can be found in figure 3.2. Distances are the straight lines and angles are the curved lines between straight or dashed lines. Full list of distance descriptors e_i and angle descriptors a_i are found in the table 3.3.

Table 3.3: Distance and angle descriptors found in each facial region.

Facial Region	**Distances**	**Angles**
Left Eyebrow	e_1,e_2,e_3,e_4	a_1
Right Eyebrow	e_5,e_6,e_7,e_8	a_2
Left Eyebrow and Eye	e_9,e_{10},e_{11}	-
Right Eyebrow and Eye	e_{12},e_{13},e_{14}	-
Left Eye	$e_{15},e_{16},e_{17},e_{18},e_{19},e_{20}$	a_3,a_4,a_5
Right Eye	$e_{21},e_{22},e_{23},e_{24},e_{25},e_{26}$	a_6,a_7,a_8
Upper Mouth	$e_{27},e_{28},e_{29},e_{30},e_{31},e_{32}$	a_9
Lower Mouth	$e_{33},e_{34},e_{35},e_{36},e_{37},e_{38}$	a_{10}
Nose	$e_{39},e_{40},e_{41},e_{42},e_{43},e_{44}$	-
Chin	$e_{45},e_{46},e_{47},e_{48},e_{49},e_{50}$	-

The distance descriptors are calculated as Euclidean distances e(l_i, l_j) between the selected landmarks where l_i and l_j represent the corresponding landmarks.

$$e(l_i,l_j)=\sqrt{(l_{j,x}-l_{i,x})^2+(l_{j,y}-l_{i,y})^2} \tag{3.1}$$

Following with a normalization of each individual distance descriptor. Normalization is done via the division of each descriptor by the amount of region belonged to each corresponding distance.

$$\hat{e}(l_i,l_j)=\frac{e(l_i,l_j)}{\sum e(l_m,l_n)}\,, \tag{3.2}$$

where $\sum e(l_m,l_n)$ belongs to the corresponding distance region. For instance, if the distance $e(l_i,l_j)$ is in the eyebrow region, then it is normalized by the total length covered in left and right eyebrow regions with points covered by $\{l_{18},l_{19},l_{20},l_{21},l_{22}\}$ and $\{l_{23},l_{24},l_{25},l_{26},l_{27}\}$ respectively. Similarly, points used for left eye, right eye, nose, mouth and chin are $\{l_{37},\ldots,l_{41}\}$, $\{l_{43},\ldots,l_{46}\}$, $\{l_{32},\ldots,l_{36}\}$, $\{l_{49},\ldots,l_{60}\}$ and $\{l_6,\ldots,l_{12}\}$ respectively. Thus, 50 distance features are calculated.

Angle descriptors are defined as the angle between two lines sharing one common landmark point. Accordingly, for a triplet of points $\{l_i,l_k,l_j\}$ following angle formulation was used:

$$a_j=\arccos\frac{e(l_i,l_k)^2+e(l_i,l_j)^2-e(l_k,l_j)^2}{2e(l_i,l_k)e(l_i,l_j)}. \quad (3.3)$$

10 angle descriptors have been calculated as a result from triplets of eyebrow regions; $\{l_{18},l_{22},l_{20}\}$, $\{l_{23},l_{27},l_{25}\}$, eye regions; $\{l_{38},l_{42},l_{37}\}$, $\{l_{37},l_{40},l_{38}\}$, $\{l_{38},l_{42},l_{40}\}$, $\{l_{45},l_{47},l_{46}\}$, $\{l_{43},l_{46},l_{45}\}$, $\{l_{45},l_{47},l_{43}\}$ and mouth region; $\{l_i,l_j,l_k\}$, $\{l_{52},l_{58},l_{55}\}$.

3.5 Clustering into Keyframes

The *k*-means clustering algorithm captures the most significant frame of the video. These frames are called keyframes. Clustering is a process of collecting the data into groups by a certain similarity. KMA is a hill climbing clustering algorithm. Simply put, hill climbing algorithms use local search optimization technique where algorithm starts by choosing a random solution node to the problem and performs incremental local search through neighboring nodes to try find a better solution [78].

For the description of KMA, let *N* be the number of data points, *x* be the data vector from x_1 to x_n, *I* is the components of each vector x^i and *d(a, b)* is a function defining the distances between the points *a* and *b*. Initially in KMA, *k* means $\{m_1, \ldots, m_k\}$ are initialized as random values from the set. Then, the algorithm works in two-step iterations. First step is the assignment step where each data point is assigned to a nearest mean while also calculating an indicator responsibility value. This is formulated as:

$$\hat{k}_n = \underset{k}{argmin}\{d(m_k, x_n)\}\ , \tag{3.4}$$

$$r_n^k = \{1 if\ \hat{k}_n = k, 0\, otherwise\}\ , \tag{3.5}$$

where, the data point is clustered to the nearest mean with r_n^k responsibility value. This is followed by the second step of the KMA, updating of the centroids. In each iteration of the algorithm, adjustment of the centroids or means are carried out since they match match the sample means of the corresponding data points. This is calculated as;

$$m_k = \frac{\sum_n r_n^k x_n}{R_k}\ , \tag{3.6}$$

where the total responsibility value R_k of centroid k is;

$$R_k = \sum_n r_n^k\ . \tag{3.7}$$

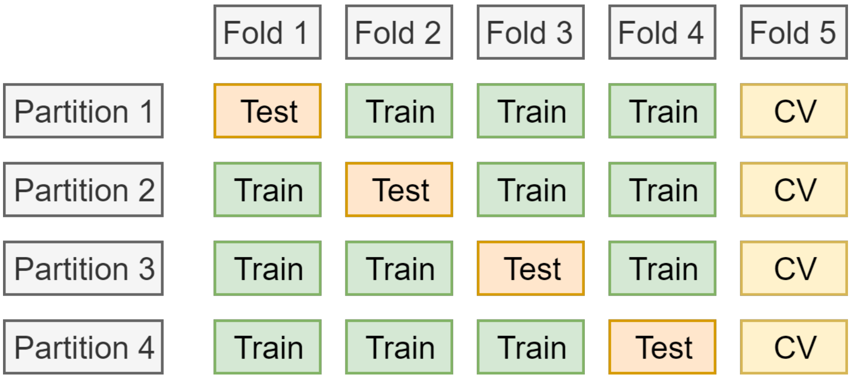
Assignment and update steps are carried out until there is no room for improvements. In other words, KMA converges when there is no change resulting in k number of clusters [18].

In a ML application, speed is a difficult problem to overcome. Other than the main goal of getting the keyframes, by clustering the feature sets, the size of the input is reduced as well. This is a positive side effect since a smaller input set results in faster computation. Sweet spot for clustering can be found by carrying the experiment for varying number of *k* and choosing the best performing *k* during learning process.

3.6 Classification, Cross Validation and Testing

Cross Validation (CV) is a technique used in classification process. Purpose of CV in prediction problem is to test the algorithms prediction ability of testing newly unknown data and capturing any over-fitting or selection bias issues. In other words, CV is used for testing the test results. When the data are in multi-dimensional format, instead of binary classifier, learner is modified as multi-modal classifier. In general, two strategies are widely used for training multi-modal classifiers. First strategy is one-vs-one strategy where each dataset is divided into pairs and each category of dataset is classified separately with other categories. Second strategy is one-vs-all (OvA) strategy which is the strategy used in this study. In strategy of OvA, one category is classified with comparison to every other class of categories [24].

For partitioning of feature sets, K-fold cross-validation technique is used. This approach allows artificial enlargement of the dataset by randomly partitioning the original set of features into K equal sized subsets [79]. Thus, the number of data in each subset would be same as the original set but each subset will have different partitioning in terms of train, test and CV allocation. The partitioning is carried out as; out of K subsets, one is used for testing, one is used for CV and the rest is used for training. As a general constraint, CV set is assigned to a constant fold for each partition in order to ensure an equal judgment of testing phase. Following this ideology, figure 3.3 shows how a 5-fold configuration would look like.



	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Partition 1	Test	Train	Train	Train	CV
Partition 2	Train	Test	Train	Train	CV
Partition 3	Train	Train	Test	Train	CV
Partition 4	Train	Train	Train	Test	CV

Figure 3.3: K-fold configuration with K = 5 [80].

Number of partitions is *K-1* instead of *K* since last fold is dedicated to CV. Note that, each partition has the same number of data-sets as original data-set. After *K*-fold partitioning is complete, classifier is trained.

In the final step of the learning and classification process, classifier is tested with test samples. Testing phase is where predictions are made for given sample and label pairs. When given a test sample, each learner in the classifier calculates scores for each category. Higher the score for an emotion category, higher the chance of sample belonging to the category. Predictions are carried out according to these scores. Lastly, from all learners, one learner for each state of emotion, predictions are combined and resulting predicted label is given to the sample. Classifier is evaluated by comparing predicted labels with true labels. Yet there is an issue about combining results from the classifiers. Assume that for a given sample, classifier gets equal votes from learners for two emotion category. In order to surpass this problem, score results from all 6 learners are added together and normalized. This means that 6 learners are combined to act like a single learner. Final prediction is made according to this combined score. The operation for getting combined scores is called score averaging. This being said, there could still be a case where emotions contain same score however this is a very unlikely scenario. In such a scenario, category is picked by alphabetical order of emotion states.

3.7 Sensitivity and Specificity

In statistical analysis, terms sensitivity and specificity are commonly used. These terms are used to quantitatively describe how good or bad the system is performing during the tests. Sensitivity is a term defining a learners ability to correctly predict samples from the dedicated category. On the other hand, specificity defines how well learner rules out samples that did not belong to the learner itself.

A binary learner can only predict if the sample belongs to the category or not. Thus the result from a binary learner is in binary format. Assume a binary learner is trying to distinguish a given sample if it belongs to angry emotion or not. Then, if given sample is angry and the prediction is correct, the result is a True Positive (TP). However, if the prediction was wrong then the result would be False Negative (FN). Now, let the sample be from another category. If the learner predicts the sample as an angry state, then the result is a False Positive (FP) while learner did not put the sample into the angry category, result would be True Negative (TN). This concept is illustrated in the table 3.4.

Table 3.4: Terms used to define sensitivity, specificity and accuracy [80].

Predicted Label	**True/False Label**		
	Positive	**Negative**	**Row Total**
Positive	TP	FP	TP+FP
Negative	FN	TN	FN+TN
Column Total	TP+FP	FP+TN	N=TP+TN+FP+FN

Sensitivity is found as the number of true positive results over the number of all samples belonging to that category and specificity is calculated as number of true negative predictions over number of all samples not belonging to that category [80].

$$Sensitivity=\frac{(TP)}{(TP+FN)}=\frac{(Number\ of\ true\ positive\ predictions)}{(Number\ of\ all\ samples\ \mathrm{in}\ the\ category)} \tag{3.8}$$

$$Specificity=\frac{(TN)}{(TN+FP)}=\frac{(Number\ of\ true\ negative\ predictions)}{(Number\ of\ all\ samples\ not\ \mathrm{in}\ the\ category)} \tag{3.9}$$

3.8 Accuracy and Loss

In a prediction based machine learning system, recognition rate of the system is calculated by the correctness of the predictions in comparison to true labels [80].

$$Accuracy=\frac{(TN+TP)}{(TN+TP+FN+FP)}=\left(\frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Samples}\right) \tag{3.10}$$

This gives a percentage value for systems ability to label samples. For each binary learner, a separate accuracy calculation is performed. Mean accuracy of all binary learners give overall accuracy of the multi-modal learner system. In addition to accuracy results, loss function is calculated. This function is an essential part of SVM algorithm however it can be used to visualize the total error made during the learning and prediction processes. For each prediction, there is cost of error which is defined as the weight of samples inside the margin of hyperplane which contributes to the overall error. Total loss is calculated by division of cost of error by total error. Learning process is evaluated by this distribution of loss function over each iteration of learning.

$$Total\ Loss=\frac{Total\ Error}{Cost\ of\ Error} \tag{3.11}$$

3.9 Receiver Operating Characteristic

Receiver Operating Characteristic (ROC) is used to diagnose healthy and unhealthy decisions taken over the curse of a performing simulation. In a classification problem, this diagnosis corresponds to decision making ability of the learners. Relationship between sensitivity and specificity is graphically shown in ROC curve diagram.

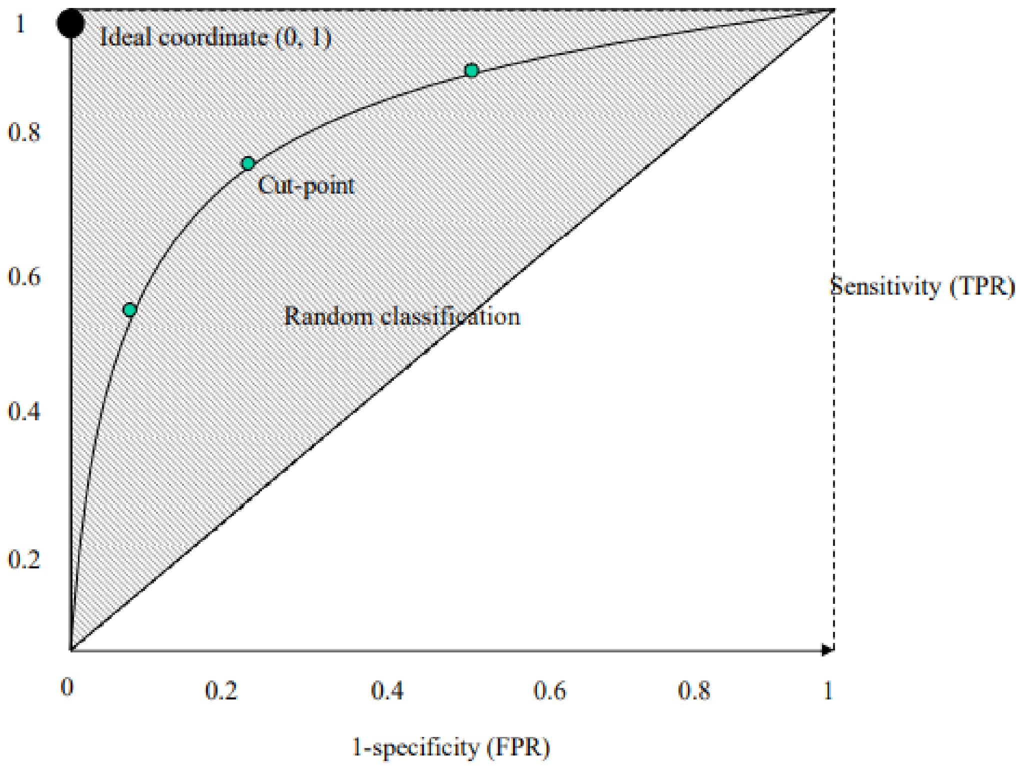


Figure 3.4: ROC Curve plot [80].

With the help of true positive rate (TPR) and false positive rate (FPR), ROC curve is plotted. In a ROC plot, closer the cut point is to the ideal coordinate (0, 1), better the accuracy while if the curve gets closer to diagonal line, then the system becomes less accurate. Area under the curve (AUC) and shape of the curve helps reader to comprehend learners performance. AUC is calculated as:

$$AUC = \int_{1}^{0} ROC(x)\,dx \tag{3.12}$$

where *x = (1-specificity)* and *ROC(x)* is the sensitivity [81]. One can observe that an AUC value of closer to 1 points to a good learning process as AUC value is between 0 and 1.

Table 3.5: Diagnosis of AUC [80].

AUC Range	**Classification**
0.9 - 1.0	Perfect
0.8 - 0.9	Very Good
0.7 - 0.8	Good
0.6 - 0.7	Satisfactory
0.5 - 0.6	Poor
< 0.5	Invalid

Table 3.5 summarizes the diagnosis of AUC. Although, it gives an idea about how well the tests went, AUC is not a definitive answer to diagnosis of a classification problem. Two same AUC results could be obtained due to two different reasons, high sensitivity and high specificity.

Chapter 4

PERSON DEPENDENT EMOTION RECOGNITION USING FACIAL INFORMATION

4.1 Person-Dependent Model

Person-dependent FER is where a person is found both in training and testing stages of the learning. For both of the databases, a person acts on various emotions more than once. From this information we can deduce that it is possible for each emotion category to give at least one video of a person in training stage and testing stages.

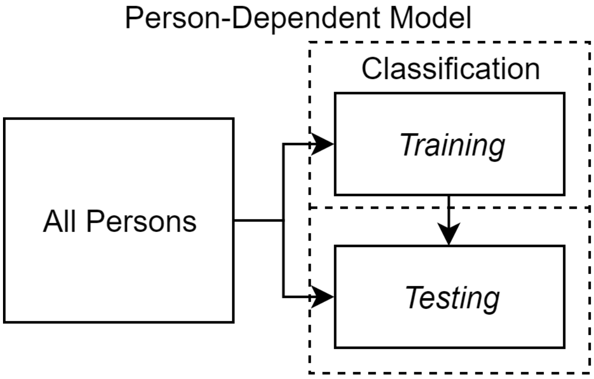


Figure 4.1: Person-dependent block diagram.

4.2 Implementation Details

Keyframes gathered from KMA are out of order compared to their corresponding video-frame sequence. Assume that *k* is assigned as arbitrary number in KMA. The result of clustering will be *k* keyframes, however algorithm does not necessarily return these keyframes in any order. To respect features as videos, keyframes are ordered manually corresponding to the video they are extracted from. Form this point onward, each *k*-tuple keyframe set corresponding to the same video is treated as one. After getting the ordered sets, *K*-fold partitioning is applied so that there are *K* number of folds with equal number of keyframe sets from each emotion category. At this point, list of keyframe sets in each fold starts with sets of angry emotion category, then continuing with disgust, fear, happiness, sadness and surprise categories in order. This list pattern is not suitable for training of classifier since learning will be done on angry emotion category first which will create a bias inside the classification process. With the aim of tackling the bias of selection before training of classifier, sets of keyframes are shuffled in each fold. Finally, folds are distributed into train, test and CV sets and the classification is carried out.

4.3 Results

Judgment of how well classifier performed is done by analyzing the accuracy of test results. This analysis is carried out for each different *k* value. Final test results are picked by corresponding highest validation accuracy on obtained. Accuracy result distributions over *k* keyframes for SAVEE and RML databases are given in Figure 4.2.

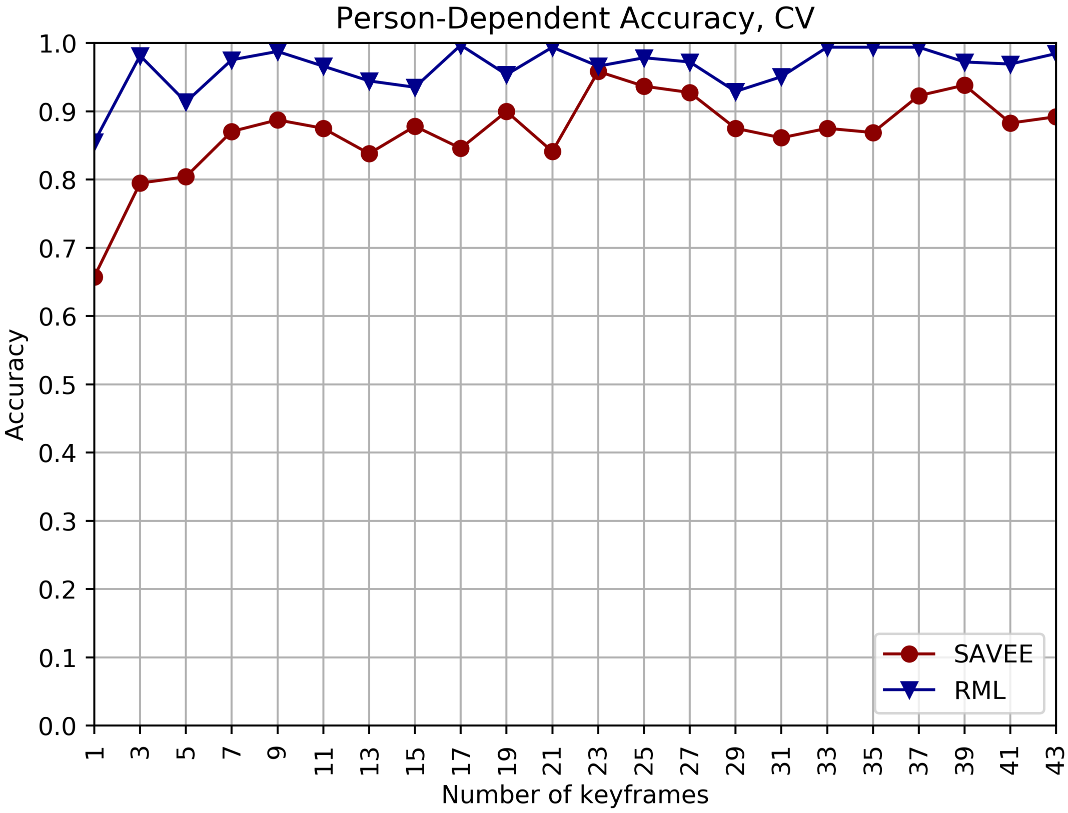


Figure 4.2: RML and SAVEE, person-dependent accuracy distribution.

Person-dependent approach yielded very high accuracy results. This was expected since the classifier is given information about facial features of the person. Since SAVEE database contains less number of persons with higher frames per second in their videos, accuracy results surpass the RML database in general. Percentage accuracy of the system goes higher as *k* increases.

Table 4.1: Person-dependent accuracy and loss table for SAVEE.

	SAVEE: Person Dependent				
	Cross Validation		**Test**		**Train**
k	**Accuracy**	**Loss**	**Accuracy**	**Loss**	**Loss**
1	0.8549	0.1451	0.8704	0.1296	0.0212
3	0.9815	0.1029	0.929	0.1368	0.027
5	0.9136	0.1661	0.9352	0.129	0.0387
7	0.9753	0.1124	0.9599	0.116	0.0301
9	0.9877	0.0971	0.9691	0.1067	0.0314

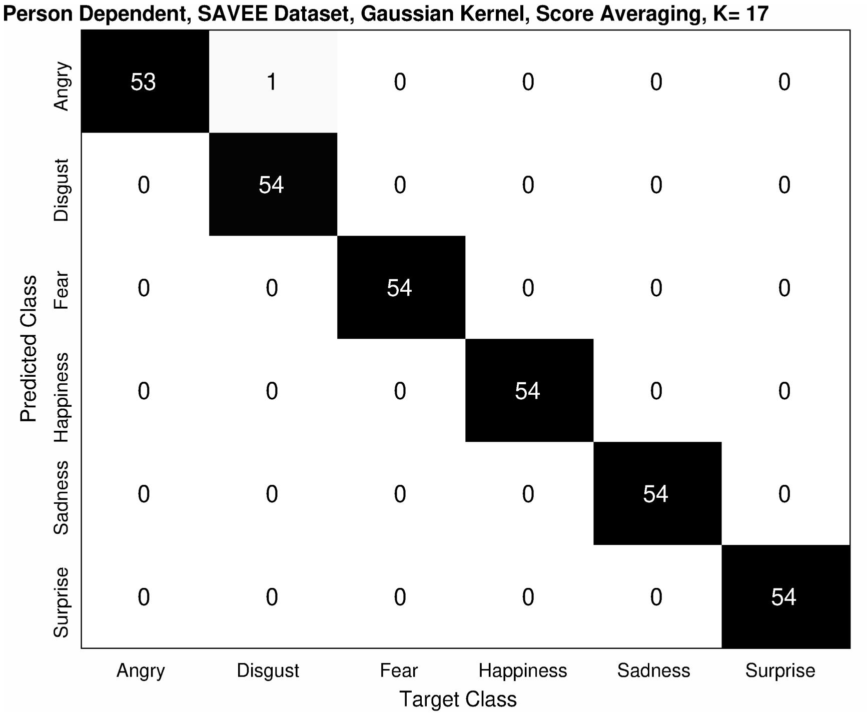
11	0.9661	0.1044	0.9599	0.1066	0.0303
13	0.9444	0.1011	0.9722	0.1045	0.028
15	0.9352	0.1274	0.9784	0.094	0.0279
17	0.9969	0.1022	0.9722	0.0968	0.0283
19	0.9537	0.1202	0.9753	0.088	0.0233
21	**0.9938**	**0.1067**	**0.9691**	**0.0947**	**0.0265**
23	0.9661	0.1027	0.9691	0.084	0.0247
25	0.9784	0.0757	0.9753	0.0847	0.0223
27	0.9722	0.074	0.9815	0.0861	0.0216
29	0.929	0.0939	0.9784	0.0896	0.0226
31	0.9506	0.0987	0.9877	0.0858	0.0192
33	0.9938	0.0734	0.9784	0.0874	0.0217
35	0.9938	0.0784	0.9753	0.0848	0.02
37	0.9938	0.0715	0.9784	0.0883	0.0203
39	0.9722	0.0722	0.9784	0.0781	0.0183
41	0.9691	0.0713	0.9877	0.0793	0.0179
43	0.9846	0.0676	0.9815	0.082	0.0177

Table 4.2: Person-dependent accuracy and loss table for RML.

	RML: Person Dependent				
	Cross Validation		**Test**		**Train**
k	**Accuracy**	**Loss**	**Accuracy**	**Loss**	**Loss**
1	0.6574	0.3426	0.6821	0.3179	0.0338
3	0.7948	0.3247	0.8256	0.3082	0.0657
5	0.804	0.2954	0.8673	0.2793	0.0696
7	0.8704	0.2753	0.8611	0.2725	0.0726
9	0.8874	0.2205	0.8719	0.2701	0.0708
11	0.875	0.2458	0.8704	0.2614	0.069
13	0.838	0.2843	0.9028	0.2442	0.0638
15	0.8781	0.2538	0.8843	0.2524	0.0643
17	0.8457	0.283	0.892	0.2385	0.0602
19	0.8997	0.2147	0.8951	0.2445	0.0591
21	0.8411	0.2786	0.8904	0.2362	0.0578
23	**0.9583**	**0.2036**	**0.8935**	**0.2434**	**0.0532**
25	0.9367	0.1873	0.892	0.247	0.0553
27	0.9275	0.213	0.8966	0.2459	0.054
29	0.875	0.2582	0.8904	0.2342	0.0505
31	0.8611	0.2483	0.892	0.2385	0.0496
33	0.875	0.2623	0.8951	0.2359	0.048
35	0.8688	0.2738	0.9043	0.2294	0.0444
37	0.9228	0.2163	0.9012	0.2349	0.0473
39	0.9383	0.2074	0.8997	0.2377	0.0473

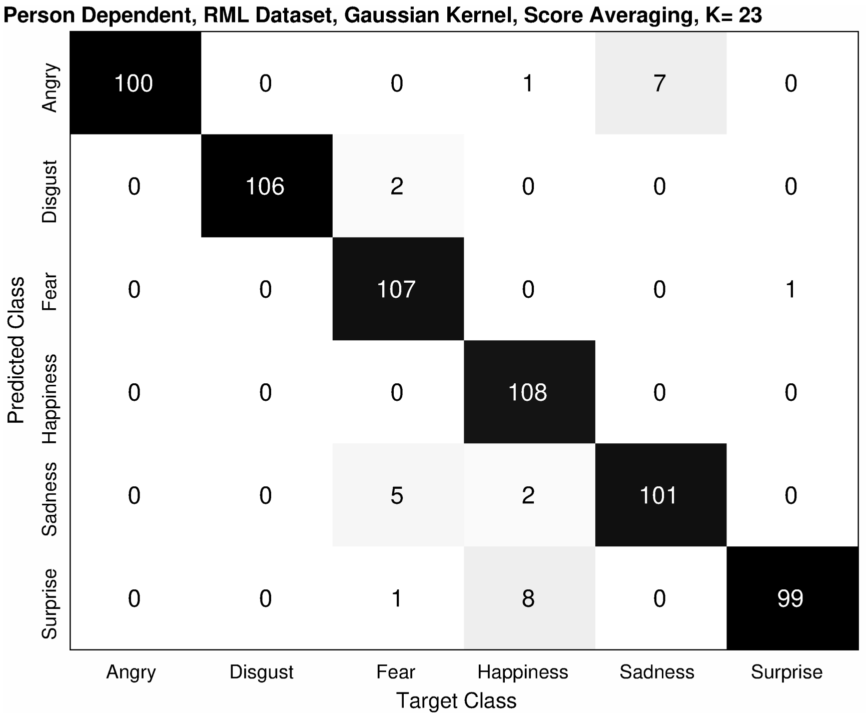
41	0.8827	0.2179	0.9074	0.236	0.0458
43	0.892	0.2571	0.8966	0.2341	0.0456

Tables 4.1 and 4.2 give the resulting accuracy and loss data for CV, test and training phases for both SAVEE and RML respectively. The highest accuracy for each datasets are %99.7 for SAVEE in *k*=17 and %95.8 for RML in *k*=23. Confusion matrices for the corresponding keyframes are in figures 4.3 and 4.4. ROC graph together with AUC calculations for the mentioned accuracy results are given in figures 4.5 and 4.6. In case where there are more than one same accuracy result for different keyframes, the lowest keyframe is chosen.



	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	53	1	0	0	0	0
Disgust	0	54	0	0	0	0
Fear	0	0	54	0	0	0
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	0	0	0	54

Figure 4.3: SAVEE, person-dependent confusion matrix for k = 17.



	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	100	0	0	1	7	0
Disgust	0	106	2	0	0	0
Fear	0	0	107	0	0	1
Happiness	0	0	0	108	0	0
Sadness	0	0	5	2	101	0
Surprise	0	0	1	8	0	99

Figure 4.4: RML, person-dependent confusion matrix for k = 23.

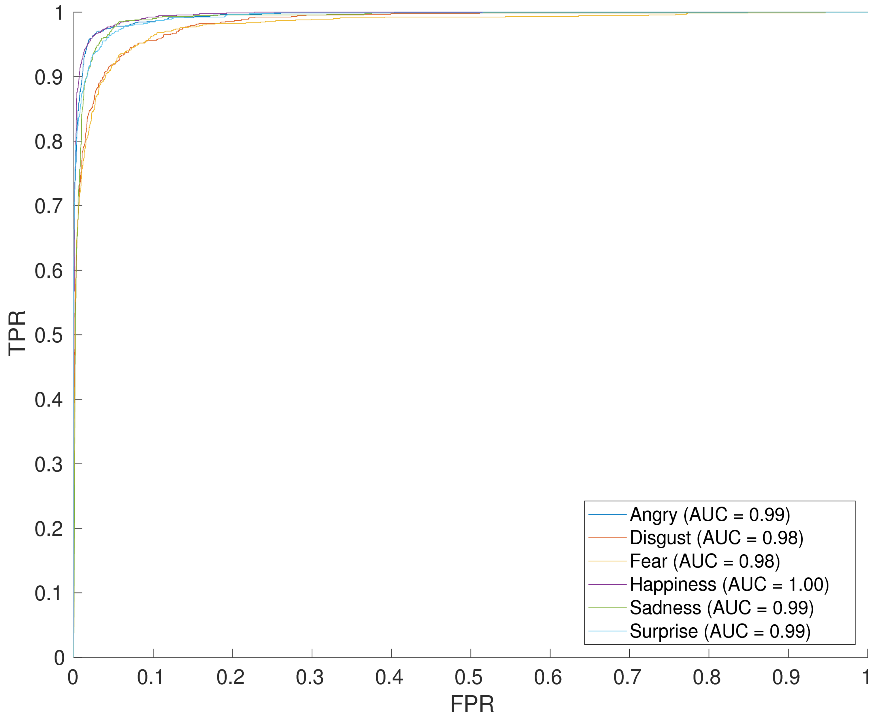


Figure 4.5: SAVEE, person-dependent ROC chart for k = 17.

Similarly, due to the composition of SAVEE database, learning process was very smooth. Classifier was very quick at optimizing the hyperplane near to the perfect precision as the graph is quickly approaching to the optimal point seen in figure 4.5. This is also observed from the AUCs of each emotion category which are between 0.98 and 1.00.

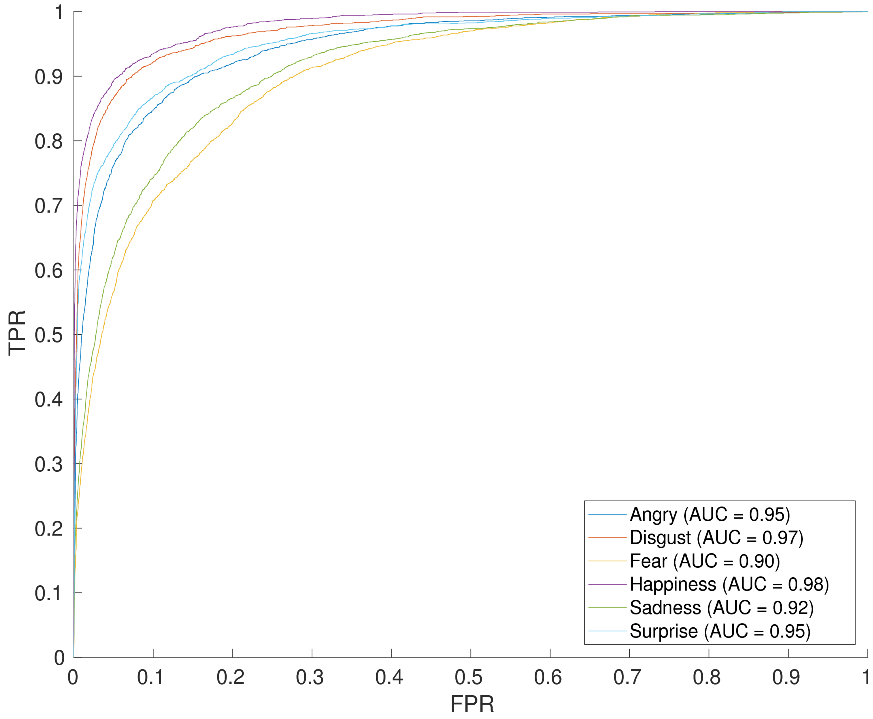


Figure 4.6: RML, person-dependent ROC chart for k = 23.

In RML database, number of videos per person is varied. From tables 3.1 it is observed that SAVEE has a very straightforward structure in person to video distributions over each emotion categories. However table 3.2 shows that for RML there was no uniform distribution on person to number of emotion categories let alone total number of each person changes greatly as well. For instance, person 1 and

2 has 25 total number of videos while person 4 has 172 videos. This affects the learning process which is seen in figure 4.6. A simple realization of the situation is that classifier is given a chance to learn about person 4 about 6 times more than person 1 or 2.

Table 4.3: Person-dependent results for best representative number of keyframes.

dataset	k	accuracy	loss	AUC
SAVEE	17	0.97	0.1	0.99
SAVEE [82]	-	0.88	-	-
RML	23	0.88	0.24	0.94

In table 4.3 it can be observed that compared to literature our model outperformed accuracy results in SAVEE however there were no study carried out for person-dependent FER on RML.

Chapter 5

PERSON INDEPENDENT EMOTION RECOGNITION USING FACIAL INFORMATION

5.1 Person-Independent Model

Second personalized model in this study is person-independent model. This model is defined by either using a persons videos during training or testing phase. Classifier is asked to predict samples of a person without learning anything about them. For this purpose, there are some implementation difference to person-dependent model and results are significantly different.

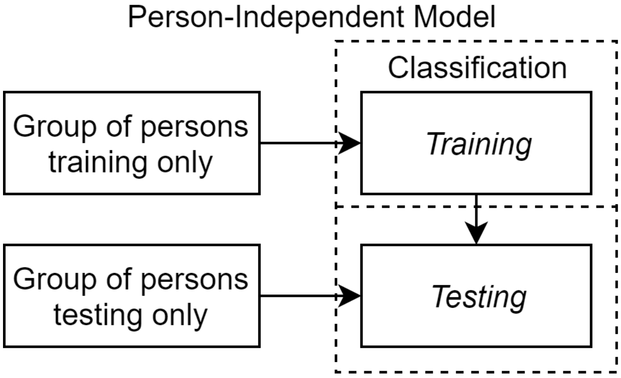


Figure 5.1: Person-independent block diagram.

5.2 Implementation Details

Initially, keyframe ordering according to video-frame sequence is carried out similarly to the person-dependent model. After getting the ordered keyframe sets, shuffling is carried out. However, before *K*-fold partitioning is carried out, each individuals' videos are grouped together and these videos are shuffled within each group. Before the shuffling process, sequence of videos of a person was in uniform order from starting from angry emotion to surprise emotion category. The purpose of this process is to break this uniformity and mix videos from emotion categories together.

Finally, when there are groups of videos of each person, *K*-fold partitioning is carried out. Since, SAVEE has 4 individual and RML has 8 individuals in their datasets, different *K* values are used in this step. For SAVEE value of *K* is 4 and for RML value of *K* is 8. Structure of partitioning is such that for each fold all of the videos of a single person was given. In the case of RML database, this means that for each fold there are different number of videos. Thus, folds 1 to 7 are used as train and test sets in each partition interchangeably and fold 8 is used as the CV set. Number of videos in fold 8 is 80 out of 720 total videos which is sufficient enough to carry CV process. Partitioning in SAVEE database is simple, *K* is set to 4. In each partition, there are 2 training sets, 1 test set and 1 CV set. Resulting in 50% of the dataset being assigned to training and other 50% is divided in half over test and CV phases.

5.3 Results

Person-independent accuracy results are within the 40-20% band which is given in figure 5.2. Accuracy and loss results are given in tables 5.1 and 5.2, confusion matrices for the best accuracy results are included in and ROC curve graphs are illustrated in figures 5.5 and 5.6 and for SAVEE and RML respectively for both tables and figures.

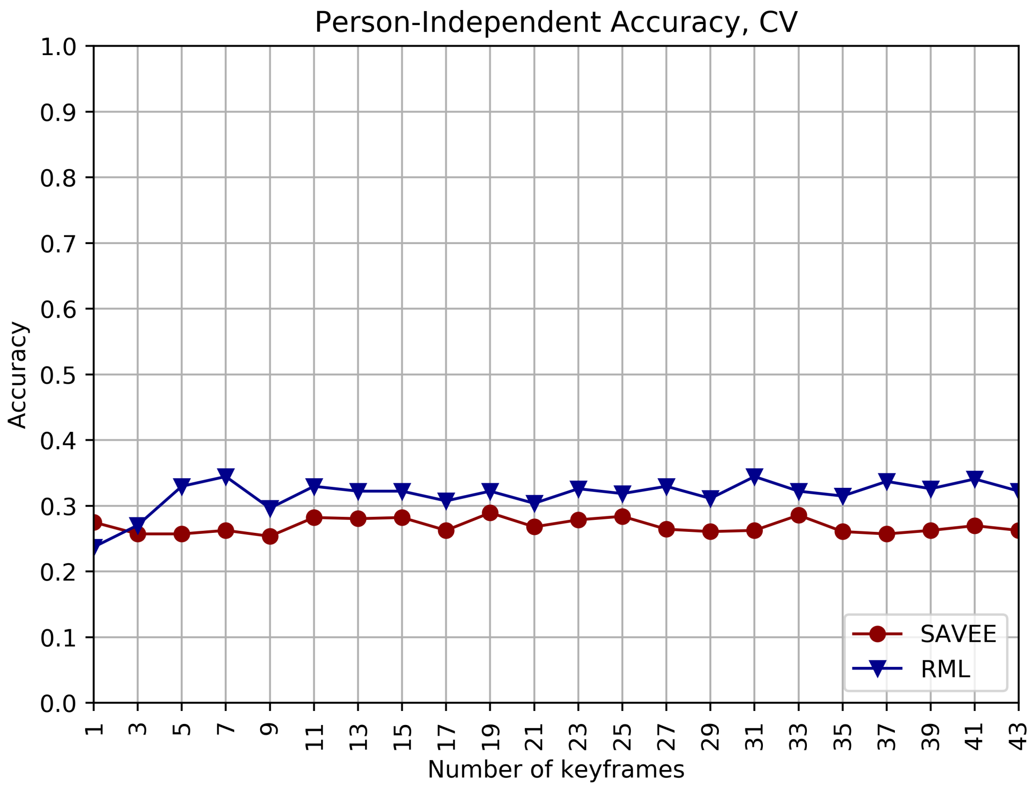


Figure 5.2: RML and SAVEE, person-independent accuracy distribution.

For the most part, SAVEE datasets overrule RML in terms of accuracy. The reason for this is that SAVEE dataset had 4 subjects with same number of videos performing the emotions while in RML 8 subjects with varying number of videos are found. Having greater number of subjects results in greater challenge since during testing phase classification becomes a higher complexity job.

Table 5.1: Person-independent accuracy and loss table for SAVEE.

	SAVEE: Person Independent				
	Cross Validation		**Test**		**Train**
k	**Accuracy**	**Loss**	**Accuracy**	**Loss**	**Loss**
1	0.237	0.763	0.3	0.7	0.0222
3	0.2704	0.7519	0.3519	0.6728	0.0247
5	0.3296	0.7304	0.3482	0.6756	0.0333
7	**0.3444**	**0.7101**	**0.3852**	**0.646**	**0.0315**
9	0.2963	0.7198	0.363	0.672	0.028

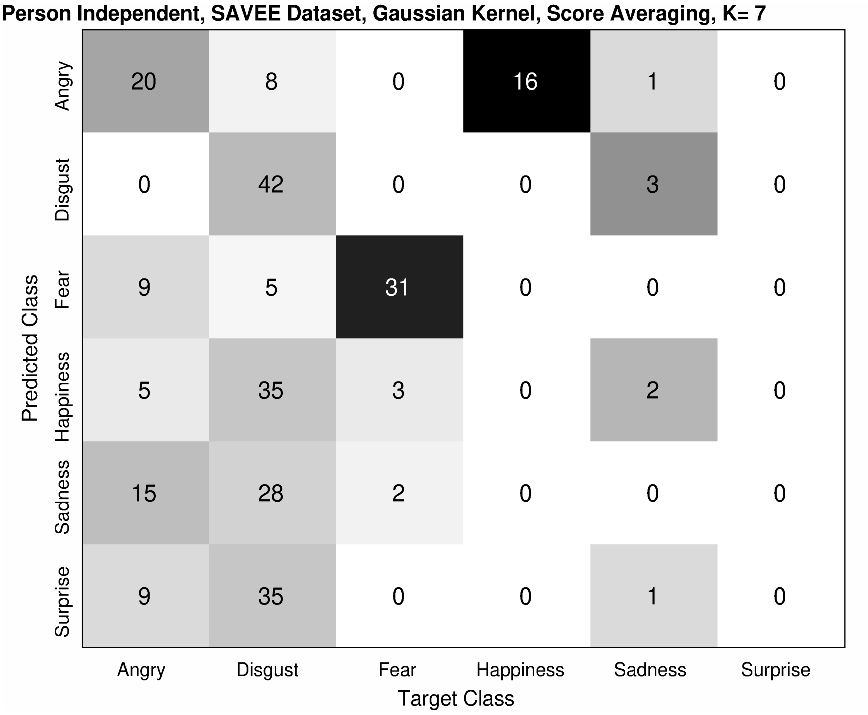
11	0.3296	0.7162	0.3926	0.6512	0.0311
13	0.3222	0.7131	0.3889	0.6524	0.0289
15	0.3222	0.7146	0.3926	0.6536	0.0272
17	0.3074	0.7185	0.3815	0.6505	0.0261
19	0.3222	0.6979	0.4111	0.6423	0.023
21	0.3037	0.7106	0.3852	0.6667	0.0217
23	0.3259	0.7039	0.3778	0.6623	0.0271
25	0.3185	0.7084	0.3889	0.653	0.023
27	0.3296	0.7096	0.3926	0.6435	0.0213
29	0.3111	0.7157	0.3815	0.6579	0.0222
31	0.3444	0.6906	0.3778	0.6645	0.0229
33	0.3222	0.7115	0.3741	0.6608	0.0205
35	0.3148	0.6985	0.3926	0.6544	0.0199
37	0.337	0.7039	0.3556	0.6683	0.0197
39	0.3259	0.7071	0.3852	0.6604	0.0204
41	0.3407	0.6985	0.3852	0.6565	0.0186
43	0.3222	0.7138	0.3704	0.6617	0.0199

Table 5.2: Person-independent accuracy and loss table for RML.

	RML: Person Independent				
	Cross Validation		**Test**		**Train**
k	**Accuracy**	**Loss**	**Accuracy**	**Loss**	**Loss**
1	0.275	0.3426	0.3047	0.3179	0.0338
3	0.2571	0.3247	0.3266	0.3082	0.0657
5	0.2571	0.2954	0.3844	0.2793	0.0696
7	0.2625	0.2753	0.3875	0.2725	0.0726
9	0.2536	0.2205	0.3703	0.2701	0.0708
11	0.2821	0.2458	0.3922	0.2614	0.069
13	0.2804	0.2843	0.3609	0.2442	0.0638
15	0.2821	0.2538	0.3766	0.2524	0.0643
17	0.2625	0.283	0.3859	0.2385	0.0602
19	**0.2893**	**0.2147**	**0.3719**	**0.2445**	**0.0591**
21	0.2679	0.2786	0.3813	0.2362	0.0578
23	0.2786	0.2036	0.3781	0.2434	0.0532
25	0.2839	0.1873	0.3766	0.247	0.0553
27	0.2643	0.213	0.3797	0.2459	0.054
29	0.2607	0.2582	0.3625	0.2342	0.0505
31	0.2625	0.2483	0.3641	0.2385	0.0496
33	0.2857	0.2623	0.3609	0.2359	0.048
35	0.2607	0.2738	0.3641	0.2294	0.0444
37	0.2571	0.2163	0.3688	0.2349	0.0473
39	0.2625	0.2074	0.3688	0.2377	0.0473

41	0.2696	0.2179	0.3688	0.236	0.0458
43	0.2625	0.2571	0.3688	0.2341	0.0456

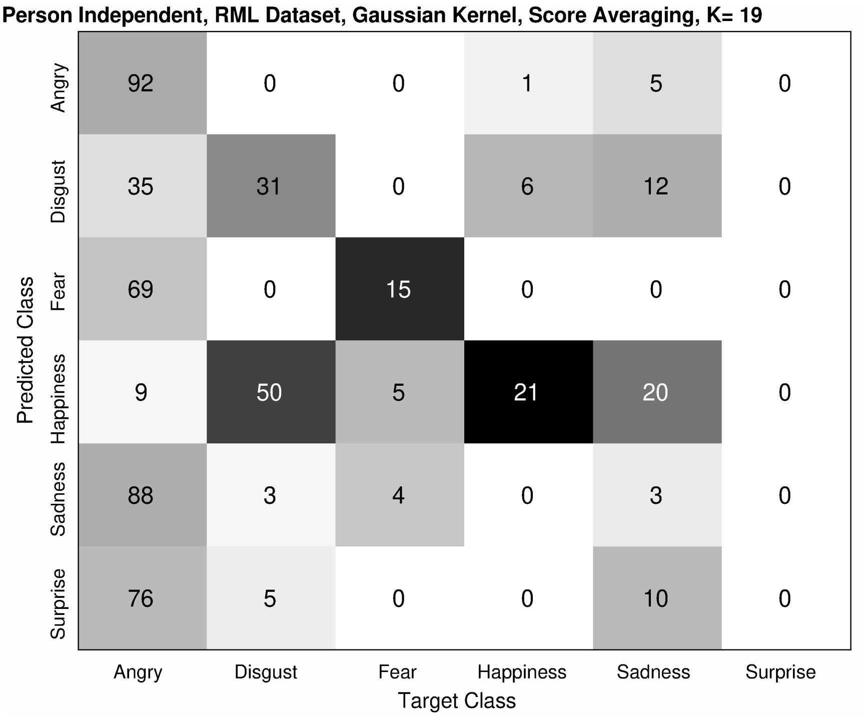
Figures 5.5 and 5.6 are the ROC curves given for SAVEE k = 7 and RML k = 19 respectively. For SAVEE, curves are very unstable. This is due to the fact that SAVEE dataset consisted of low number of video clips. Also, when one emotion was learned quickly, others were lacking behind. For instance, emotion categories sadness and fear have a lower AUC in both databases.



Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 7

Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	20	8	0	16	1	0
Disgust	0	42	0	0	3	0
Fear	9	5	31	0	0	0
Happiness	5	35	3	0	2	0
Sadness	15	28	2	0	0	0
Surprise	9	35	0	0	1	0

Figure 5.3 SAVEE, person-independent confusion matrix for k = 7.



Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 19

Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	92	0	0	1	5	0
Disgust	35	31	0	6	12	0
Fear	69	0	15	0	0	0
Happiness	9	50	5	21	20	0
Sadness	88	3	4	0	3	0
Surprise	76	5	0	0	10	0

Figure 5.4 RML, person-independent confusion matrix for k = 19.

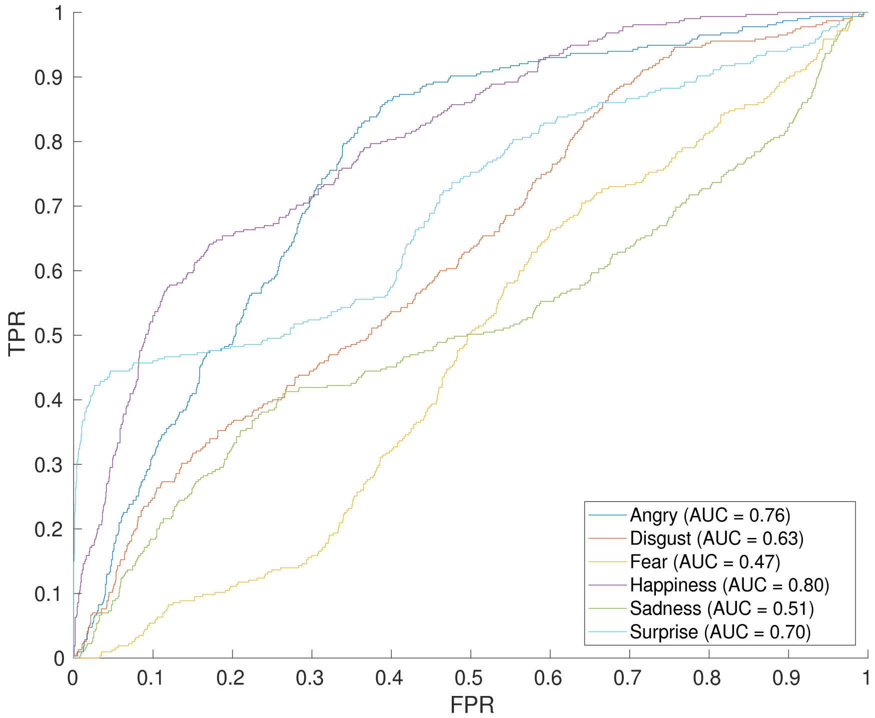


Figure 5.5: SAVEE, person-independent ROC chart for k = 7.

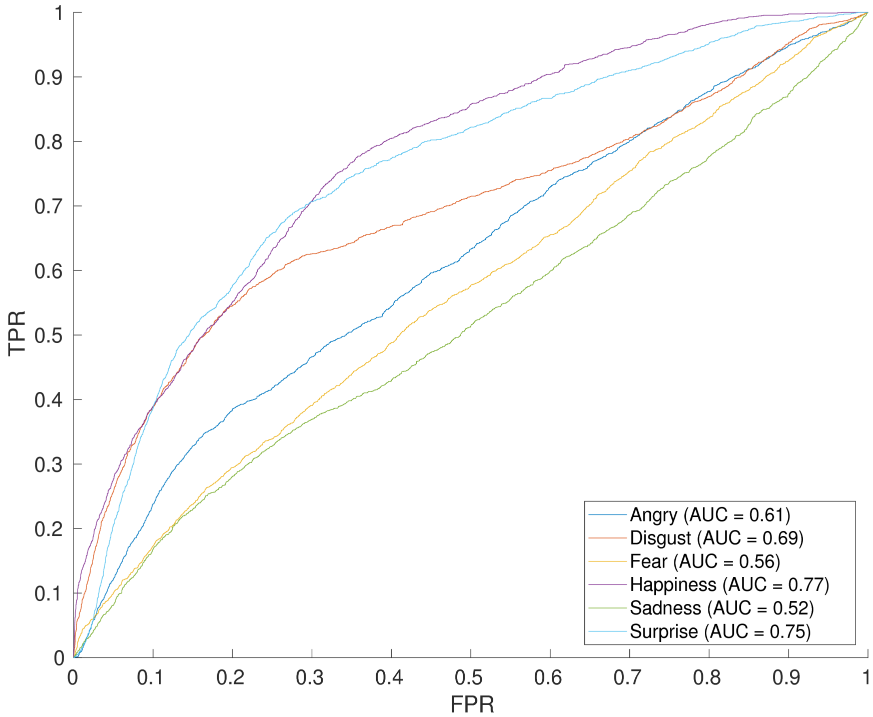


Figure 5.6: RML, person-independent ROC chart for k = 19.

When considering feature vectors, learner is essentially looking for distinct differences between categories. If there is no such differences between one category and others, then learner will struggle in classification. Another reason for low AUC is when there is similarity between categories in terms of features. In fear and sadness categories, this is observed. Facial structure during these emotions are similar to each other and the distance and angle descriptors are not sufficient to identify the differences between the two categories. However, for the most part, AUCs are mostly above 0.5 which is a decent result considering that facial structure of the person tested is not given during learning phase.

Table 5.3: Person-independent results for best representative number of keyframes.

dataset	k	accuracy	loss	AUC
SAVEE	7	0.39	0.65	0.65
SAVEE [77]	4	0.36	-	-

RML	19	0.37	0.67	0.65
RML [77]	4	0.32	-	-

Observing from table 5.3, results gathered in this study outperformed compared to the results from literature found in person-independent models for both of the databases.

Chapter 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this study, person-dependent and person-independent models of FER was studied. The focus was to improve the learning phase in terms of speed and efficiency by introducing the concept of keyframes. The experiments were performed on facial databases of SAVEE and RML. For the purpose of FER, facial landmarks were used as a basis for extraction of distance and angle feature descriptors. The features were extracted from 68 landmark points located on the face of the subject. These features were then clustered into keyframes. In order to preserve the integrity of the original videos, the keyframes were ordered similar to the videos and are used as a whole during classification. CV technique was employed in classification phase. This was optimized for both of the model structures. CV provided a larger sample space to work with during learning and prediction stages. Finally when the test results were gathered, it has been observed that for some k number of keyframes, classifier performed better compared to others. The corresponding k are chosen to be the best performing keyframe sets.

Experimental results show that mid-range k clustering performance was best compared to lower and upper k values. Observing from simulations, it is clear that person-dependent approach outperformed person-independent approach. However, this study proved outstanding results in person-dependent model while also resulting decent results for person-independent model. In addition, it is possible to optimize the classification process by adjusting the number of keyframes used.

6.2 Future Work

This research revealed multiple improvements in terms of both models of ER. While person-dependent analysis did perform sufficiently high in terms of accuracy, person-independent model was behind. To improve recognition rates, a better selection of features can be performed. Also, since the topic of the study was focused on FER, voice counterparts of the emotion categories were not considered. However, there is a potential for improvement if both voice and image counterparts of emotions were used together to create a better platform for classification. This would be a closer approach to real life human to human communication.

REFERENCES

[1] A. K. Jain, A. Ross, S. Prabhakar, and others, “An introduction to biometric recognition,” *IEEE Trans. circuits Syst. video Technol.*, vol. 14, no. 1, 2004.

[2] S. Prabhakar, S. Pankanti, and A. K. Jain, “Biometric recognition: Security and privacy concerns,” *IEEE Secur. Priv.*, no. 2, pp. 33–42, 2003.

[3] R. Cowie *et al.*, “Emotion recognition in human-computer interaction,” *IEEE Signal Process. Mag.*, vol. 18, no. 1, pp. 32–80, 2001.

[4] R. Cowie and E. Douglas-Cowie, “Speakers and hearers are people: Reflections on speech deterioration as a consequence of acquired deafness,” *Profound Deaf. speech Commun.*, pp. 510–527, 1995.

[5] S. V Ioannou, A. T. Raouzaiou, V. A. Tzouvaras, T. P. Mailis, K. C. Karpouzis, and S. D. Kollias, “Emotion recognition through facial expression analysis based on a neurofuzzy network,” *Neural Networks*, vol. 18, no. 4, pp. 423–435, 2005.

[6] J. A. Russell, “Is there universal recognition of emotion from facial expression? A review of the cross-cultural studies.,” *Psychol. Bull.*, vol. 115, no. 1, p. 102, 1994.

[7] P. Ekman, “Strong evidence for universals in facial expressions: a reply to Russell’s mistaken critique.,” 1994.

[8] C. E. Izard, “Innate and universal facial expressions: evidence from developmental and cross-cultural research.,” 1994.

[9] J. A. Russell, “Facial expressions of emotion: What lies beyond minimal universality?,” *Psychol. Bull.*, vol. 118, no. 3, p. 379, 1995.

[10] S. Bloch and H. G. Santibáñez, “Training emotional ,effection ‘in humans: significance of its feedback on subjectivity,” *Psicobiol. del Aprendizaje. Santiago Publ. Fac. Med., Univ. Chile*, 1972.

[11] A. J. Calder, “Facial emotion recognition after bilateral amygdala damage: Differentially severe impairment of fear,” *Cogn. Neuropsychol.*, vol. 13, no. 5, pp. 699–745, 1996.

[12] C. G. Kohler *et al.*, “Facial emotion recognition in schizophrenia: intensity effects and error pattern,” *Am. J. Psychiatry*, vol. 160, no. 10, pp. 1768–1774, 2003.

[13] M. B. Harms, A. Martin, and G. L. Wallace, “Facial emotion recognition in autism spectrum disorders: a review of behavioral and neuroimaging studies,” *Neuropsychol. Rev.*, vol. 20, no. 3, pp. 290–322, 2010.

[14] R. Adolphs *et al.*, “Recognition of facial emotion in nine individuals with bilateral amygdala damage,” *Neuropsychologia*, vol. 37, no. 10, pp. 1111–1117, 1999.

[15] M. S. Bartlett, G. Littlewort, M. Frank, C. Lainscsek, I. Fasel, and J. Movellan, “Recognizing facial expression: machine learning and application to spontaneous behavior,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, 2005, vol. 2, pp. 568–573.

[16] Z. Yu and C. Zhang, “Image based static facial expression recognition with multiple deep network learning,” in *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, 2015, pp. 435–442.

[17] M. S. Bartlett *et al.*, “Automatic recognition of facial actions in spontaneous expressions.,” *J. Multimed.*, vol. 1, no. 6, pp. 22–35, 2006.

[18] Y. Yang, “Information theory, inference, and learning algorithms.” Taylor & Francis, 2005.

[19] A. A. A. Youssif and W. A. A. Asker, “Automatic facial expression recognition system based on geometric and appearance features,” *Comput. Inf. Sci.*, vol. 4, no. 2, p. 115, 2011.

[20] M. Sonka, V. Hlavac, and R. Boyle, *Image processing, analysis, and machine vision*. Cengage Learning, 2014.

[21] H. Demirel and G. Anbarjafari, “Image resolution enhancement by using discrete and stationary wavelet decomposition,” *IEEE Trans. image Process.*, vol. 20, no. 5, pp. 1458–1460, 2010.

[22] H. Demirel, C. Ozcinar, and G. Anbarjafari, “Satellite image contrast enhancement using discrete wavelet transform and singular value decomposition,” *IEEE Geosci. Remote Sens. Lett.*, vol. 7, no. 2, pp. 333–337, 2009.

[23] H. Demirel and G. Anbarjafari, “Satellite image resolution enhancement using complex wavelet transform,” *IEEE Geosci. Remote Sens. Lett.*, vol. 7, no. 1, pp. 123–126, 2009.

[24] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[25] Y. Bengio and N. Chapados, “Extensions to metric-based model selection,” *J. Mach. Learn. Res.*, vol. 3, no. Mar, pp. 1209–1227, 2003.

[26] H. Soyel and H. Demirel, “Facial expression recognition using 3D facial feature distances,” in *International Conference Image Analysis and Recognition*, 2007, pp. 831–838.

[27] H. Soyel and H. Demirel, “Localized discriminative scale invariant feature transform based facial expression recognition,” *Comput. Electr. Eng.*, vol. 38, no. 5, pp. 1299–1309, 2012.

[28] H. Soyel, U. Tekguc, and H. Demirel, “Application of NSGA-II to feature selection for facial expression recognition,” *Comput. Electr. Eng.*, vol. 37, no. 6, pp. 1232–1240, 2011.

[29] H. Soyel and H. Demirel, “Facial expression recognition based on discriminative scale invariant feature transform,” *Electron. Lett.*, vol. 46, no. 5, pp. 343–345, 2010.

[30] H. Soyel and H. Demirel, “Optimal feature selection for 3D facial expression recognition using coarse-to-fine classification,” *Turkish J. Electr. Eng. Comput. Sci.*, vol. 18, no. 6, pp. 1031–1040, 2010.

[31] K. Yurtkan, H. Soyel, H. Demirel, H. Özkaramanl\i, M. Uyguro\uglu, and E. Varo\uglu, “Face modeling and adaptive texture mapping for model based video coding,” in *International Conference on Computer Analysis of Images and Patterns*, 2005, pp. 498–505.

[32] K. Yurtkan, H. Soyel, and H. Demirel, “Feature Selection for Enhanced 3D Facial Expression Recognition Based on Varying Feature Point Distances,” in *Information Sciences and Systems 2013*, Springer, 2013, pp. 209–217.

[33] I. Lüsi *et al.*, “Joint challenge on dominant and complementary emotion recognition using micro emotion features and head-pose estimation: Databases,” in *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, 2017, pp. 809–813.

[34] N. Hajarolasvadi and H. Demirel, “3D CNN-Based Speech Emotion Recognition Using K-Means Clustering and Spectrograms,” *Entropy*, vol. 21, no. 5, p. 479, 2019.

[35] O.-W. Kwon, K. Chan, J. Hao, and T.-W. Lee, “Emotion recognition by speech signals,” in *Eighth European Conference on Speech Communication and Technology*, 2003.

[36] K. Yurtkan and H. Demirel, “Entropy-based feature selection for improved 3D facial expression recognition,” *Signal, Image Video Process.*, vol. 8, no. 2, pp. 267–277, 2014.

[37] I. Cohen, N. Sebe, A. Garg, L. S. Chen, and T. S. Huang, “Facial expression recognition from video sequences: temporal and static modeling,” *Comput. Vis. image Underst.*, vol. 91, no. 1–2, pp. 160–187, 2003.

[38] M. Dash and H. Liu, “Feature selection for classification,” *Intell. data Anal.*, vol. 1, no. 1–4, pp. 131–156, 1997.

[39] N. Kwak and C.-H. Choi, “Input feature selection for classification problems,” *IEEE Trans. neural networks*, vol. 13, no. 1, pp. 143–159, 2002.

[40] F. M. Reza, *An introduction to information theory*. Courier Corporation, 1994.

[41] K. Kira and L. A. Rendell, “A practical approach to feature selection,” in *Machine Learning Proceedings 1992*, Elsevier, 1992, pp. 249–256.

[42] I. Kononenko, “Estimating attributes: analysis and extensions of RELIEF,” in *European conference on machine learning*, 1994, pp. 171–182.

[43] M. Robnik-Šikonja and I. Kononenko, “Theoretical and empirical analysis of ReliefF and RReliefF,” *Mach. Learn.*, vol. 53, no. 1–2, pp. 23–69, 2003.

[44] E. Corchado Rodr\'\iguez and H. Yin, *Intelligent Data Engineering and Automated Learning-IDEAL 2009*. Springer, 2009.

[45] M. A. Hall, “Correlation-based feature selection for machine learning,” 1999.

[46] Z. Zhao and H. Liu, “Searching for interacting features in subset selection,” *Intell. Data Anal.*, vol. 13, no. 2, pp. 207–228, 2009.

[47] K. C. C. Chan, “A statistical technique for extracting classificatory knowledge from databases,” *Knowl. Discov. databases*, 1991.

[48] E. R. Girden, *ANOVA: Repeated measures*, no. 84. Sage, 1992.

[49] D. S. Moore, G. P. McCabe, W. M. Duckworth, and S. L. Sclove, *The practice of business statistics: using data for decisions*. Freeman and Company, 2003.

[50] G. D. Garson, "Discriminant function analysis," *Asheboro, NC Stat. Assoc. Publ. North Carolina State Univ.*, 2012.

[51] S. Das, "Filters, wrappers and a boosting-based hybrid for feature selection," in *Icml*, 2001, vol. 1, pp. 74–81.

[52] P. Xuan, M. Z. Guo, J. Wang, C. Y. Wang, X. Y. Liu, and Y. Liu, "Genetic algorithm-based efficient feature selection for classification of pre-miRNAs," *Genet. Mol. Res.*, vol. 10, no. 2, pp. 588–603, 2011.

[53] S. C. Shah and A. Kusiak, "Data mining and genetic algorithm based gene/SNP selection," *Artif. Intell. Med.*, vol. 31, no. 3, pp. 183–196, 2004.

[54] G. Kapetanios, "Variable selection using non-standard optimisation of information criteria," 2005.

[55] R. Meiri and J. Zahavi, "Using simulated annealing to optimize the feature selection problem in marketing applications," *Eur. J. Oper. Res.*, vol. 171, no. 3, pp. 842–858, 2006.

[56] H. R. Lourenço, O. C. Martin, and T. Stützle, “Iterated local search,” in *Handbook of metaheuristics*, Springer, 2003, pp. 320–353.

[57] T. N. Lal, O. Chapelle, J. Weston, and A. Elisseeff, “Embedded methods,” in *Feature extraction*, Springer, 2006, pp. 137–165.

[58] Y. Zhou, R. Jin, and S. C.--H. Hoi, “Exclusive lasso for multi-task feature selection,” in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 988–995.

[59] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *J. R. Stat. Soc. Ser. B*, vol. 58, no. 1, pp. 267–288, 1996.

[60] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited, 2016.

[61] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, “Supervised machine learning: A review of classification techniques,” *Emerg. Artif. Intell. Appl. Comput. Eng.*, vol. 160, pp. 3–24, 2007.

[62] E. Alpaydin, *Introduction to machine learning*. MIT press, 2009.

[63] T. G. Dietterich, “Ensemble methods in machine learning,” in *International workshop on multiple classifier systems*, 2000, pp. 1–15.

[64] J. E. Hopcroft and J. D. Ullman, *Data structures and algorithms*. 1983.

[65] E. M. Kleinberg and others, “An overtraining-resistant stochastic modeling method for pattern recognition,” *Ann. Stat.*, vol. 24, no. 6, pp. 2319–2349, 1996.

[66] A. Liaw, M. Wiener, and others, “Classification and regression by randomForest,” *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[67] S. Theodoridis, K. Koutroumbas, and others, “Pattern recognition,” *IEEE Trans. Neural Networks*, vol. 19, no. 2, p. 376, 2008.

[68] B. Schölkopf, A. Smola, and K.-R. Müller, “Kernel principal component analysis,” in *International conference on artificial neural networks*, 1997, pp. 583–588.

[69] T. Hofmann, B. Schölkopf, and A. J. Smola, “Kernel methods in machine learning,” *Ann. Stat.*, pp. 1171–1220, 2008.

[70] P. Jackson and S. Haq, “Surrey audio-visual expressed emotion (savee) database,” *Univ. Surrey Guildford, UK*, 2014.

[71] Z. Xie, “Ryerson Multimedia Research Lab,” *Univ. Surrey Guildford, UK*, 2014.

[72] R. Ekman, *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*. Oxford University Press, USA, 1997.

[73] J. F. Cohn, Z. Ambadar, and P. Ekman, “Observer-based measurement of facial expression with the Facial Action Coding System,” *Handb. Emot. elicitation Assess.*, pp. 203–221, 2007.

[74] J. Ostermann, “Animation of synthetic faces in MPEG-4,” in *Proceedings Computer Animation’98 (Cat. No. 98EX169)*, 1998, pp. 49–55.

[75] Y.-L. Tian, T. Kanade, and J. F. Cohn, “Facial expression analysis,” in *Handbook of face recognition*, Springer, 2005, pp. 247–275.

[76] T. Baltrušaitis, P. Robinson, and L.-P. Morency, “Openface: an open source facial behavior analysis toolkit,” in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016, pp. 1–10.

[77] F. Noroozi, M. Marjanovic, A. Njegus, S. Escalera, and G. Anbarjafari, “Audio-visual emotion recognition in video clips,” *IEEE Trans. Affect. Comput.*, 2017.

[78] S. Edelkamp and S. Schroedl, *Heuristic search: theory and applications*. Elsevier, 2011.

[79] T. Hastie, R. Tibshirani, and J. Friedman, “The elements of statistical learning: data mining, inference, and prediction, Springer Series in Statistics.” Springer New York, 2009.

[80] W. Zhu, N. Zeng, N. Wang, and others, “Sensitivity, specificity, accuracy, associated confidence interval and ROC analysis with practical SAS implementations,” *NESUG Proc. Heal. care life Sci. Balt. Maryl.*, vol. 19, p. 67, 2010.

[81] M. H. Zweig and G. Campbell, “Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine.,” *Clin. Chem.*, vol. 39, no. 4, pp. 561–577, 1993.

[82] S. Haq, P. J. B. Jackson, and J. Edge, “Speaker-dependent audio-visual emotion recognition.,” in *AVSP*, 2009, pp. 53–58.

[83] B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

APPENDICES

Appendix A: Charts for test and loss results

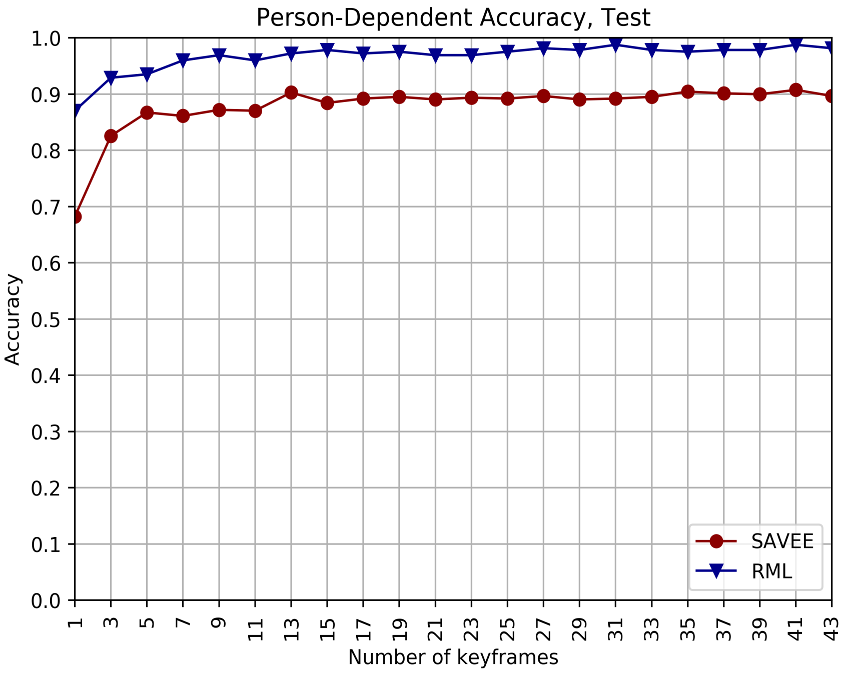


Figure 6.1: Person-dependent accuracy graph on test data.

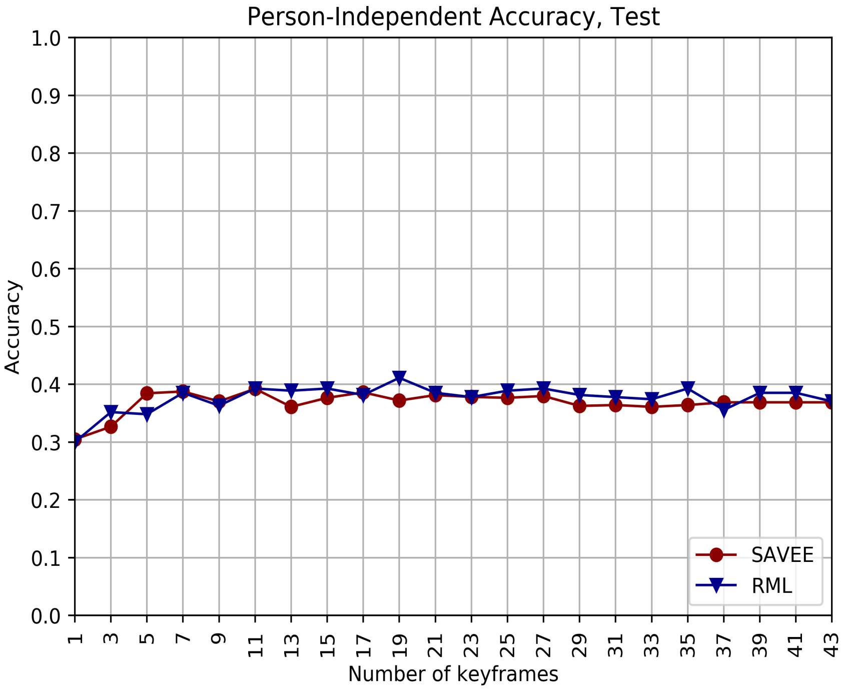


Figure 6.2: Person-independent accuracy graph on test data.

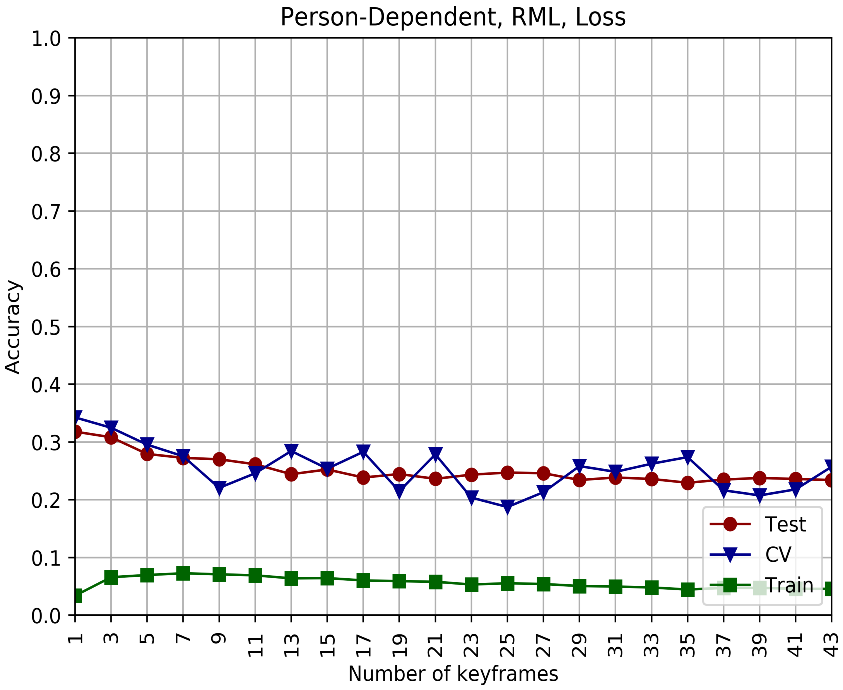


Figure 6.3: Person-dependent loss graph on RML dataset.

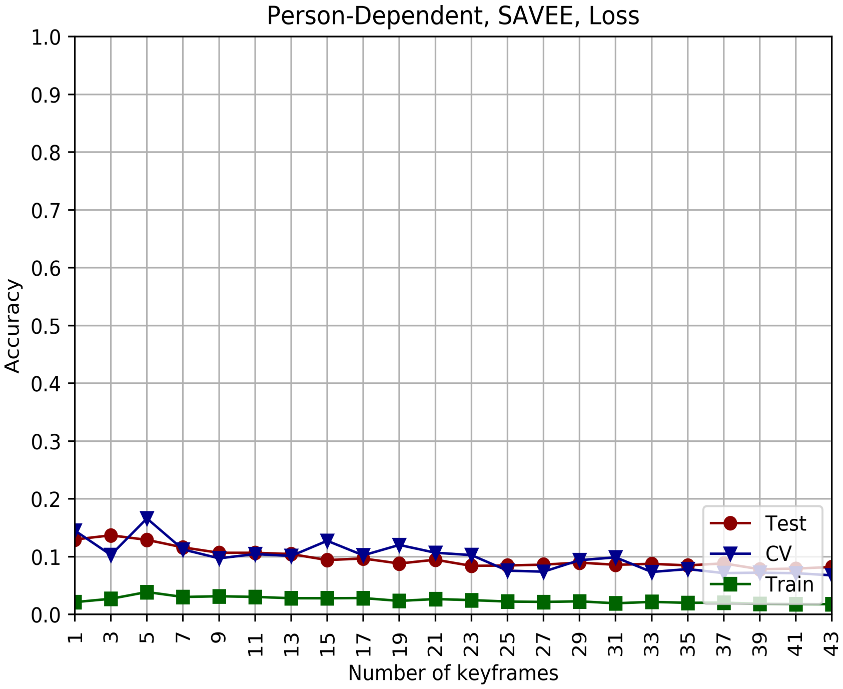


Figure 6.4: Person-dependent loss graph on SAVEE dataset.

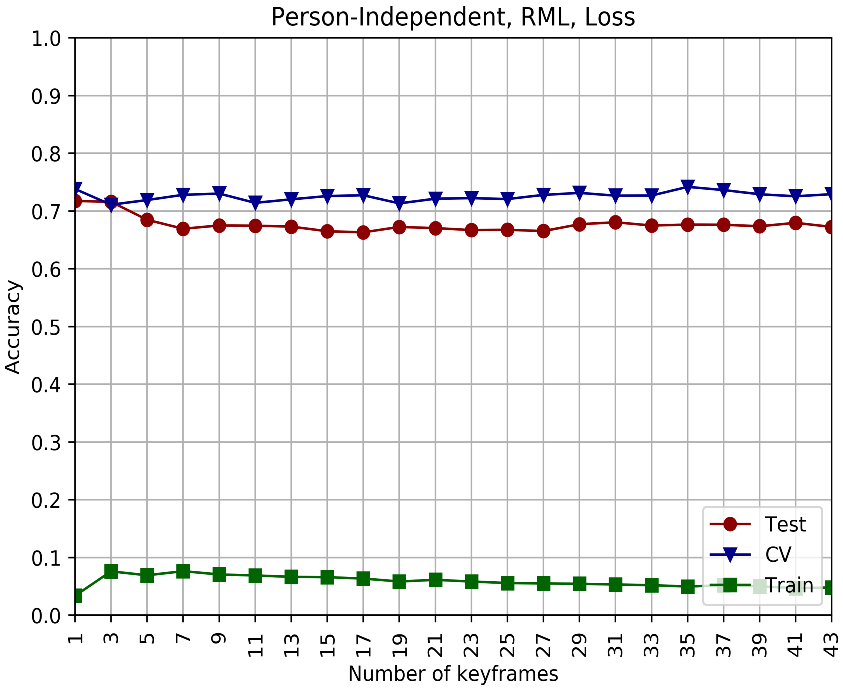


Figure 6.5: Person-independent loss graph on RML dataset.

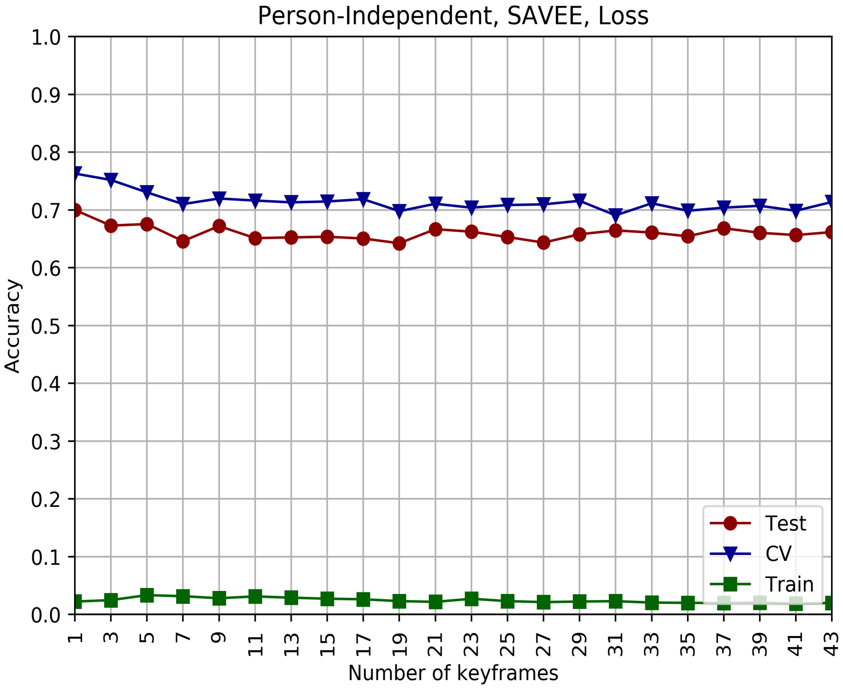
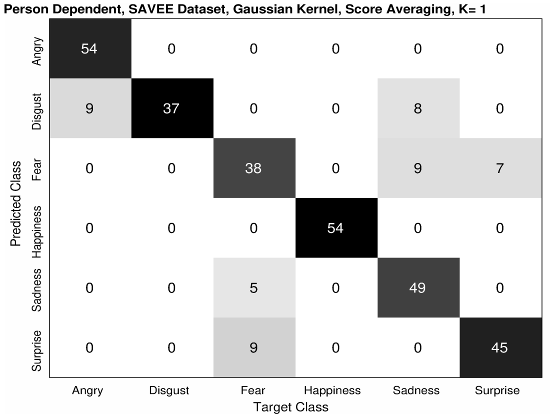


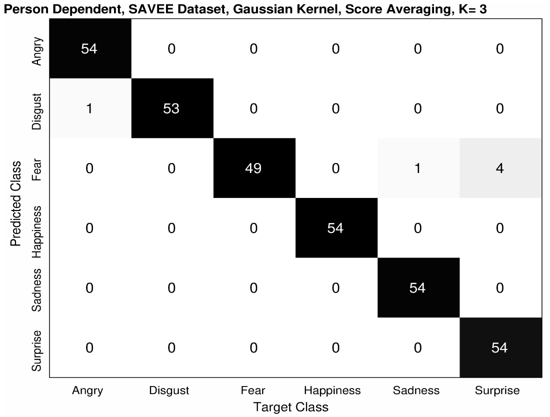
Figure 6.6: Person-independent loss graph on SAVEE dataset.

Appendix B: Confusion matrices (SAVEE, Person-Dependent)



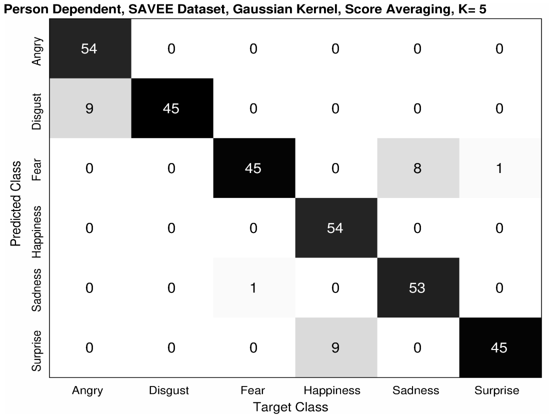
Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 1

	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	54	0	0	0	0	0
Disgust	9	37	0	0	8	0
Fear	0	0	38	0	9	7
Happiness	0	0	0	54	0	0
Sadness	0	0	5	0	49	0
Surprise	0	0	9	0	0	45



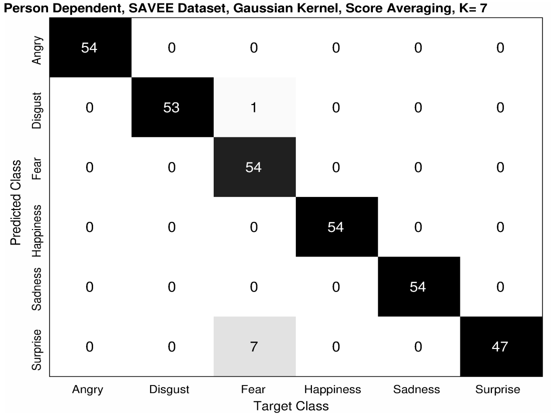
Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 3

	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	54	0	0	0	0	0
Disgust	1	53	0	0	0	0
Fear	0	0	49	0	1	4
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	0	0	0	54



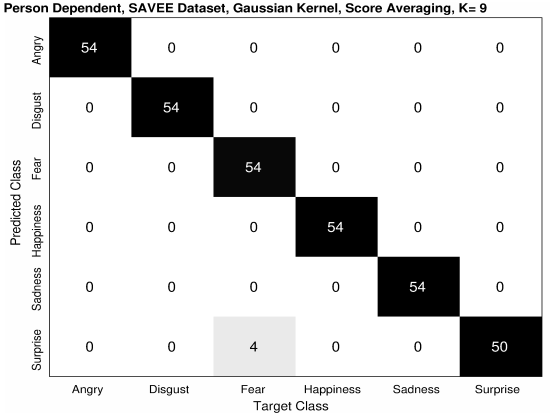
Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 5

	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	54	0	0	0	0	0
Disgust	9	45	0	0	0	0
Fear	0	0	45	0	8	1
Happiness	0	0	0	54	0	0
Sadness	0	0	1	0	53	0
Surprise	0	0	0	9	0	45



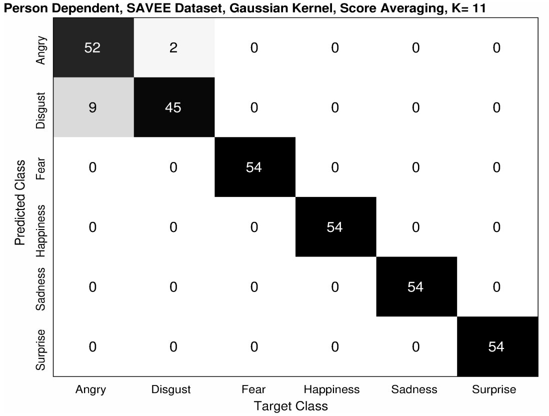
Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 7

	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	54	0	0	0	0	0
Disgust	0	53	1	0	0	0
Fear	0	0	54	0	0	0
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	7	0	0	47



Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 9

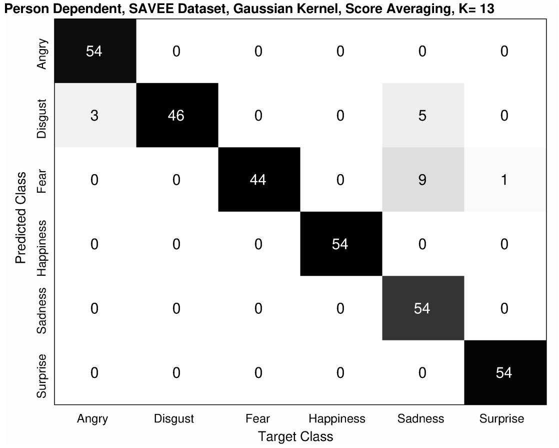
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	54	0	0	0	0	0
Disgust	0	54	0	0	0	0
Fear	0	0	54	0	0	0
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	4	0	0	50



Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 11

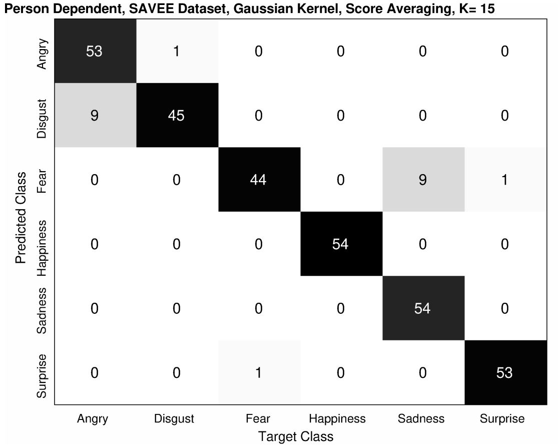
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	52	2	0	0	0	0
Disgust	9	45	0	0	0	0
Fear	0	0	54	0	0	0
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	0	0	0	54

Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 13



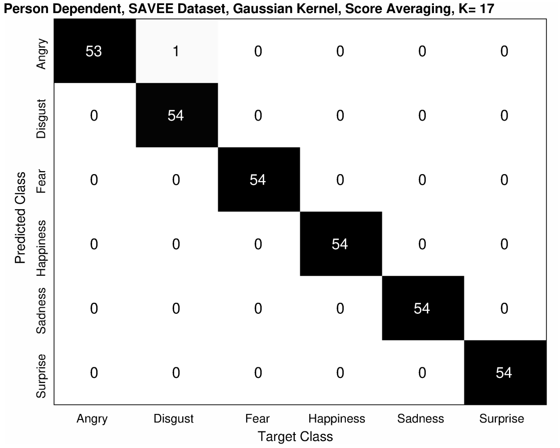
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	54	0	0	0	0	0
Disgust	3	46	0	0	5	0
Fear	0	0	44	0	9	1
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	0	0	0	54

Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 15



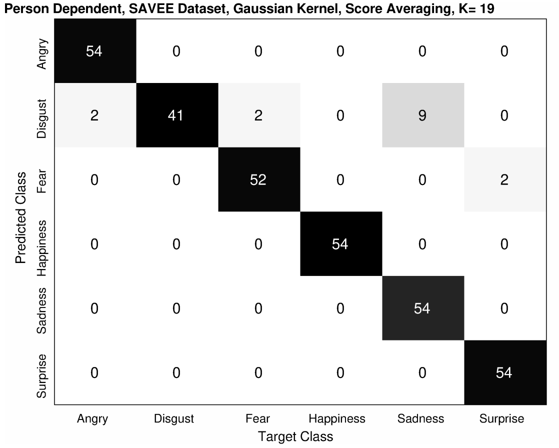
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	53	1	0	0	0	0
Disgust	9	45	0	0	0	0
Fear	0	0	44	0	9	1
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	1	0	0	53

Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 17



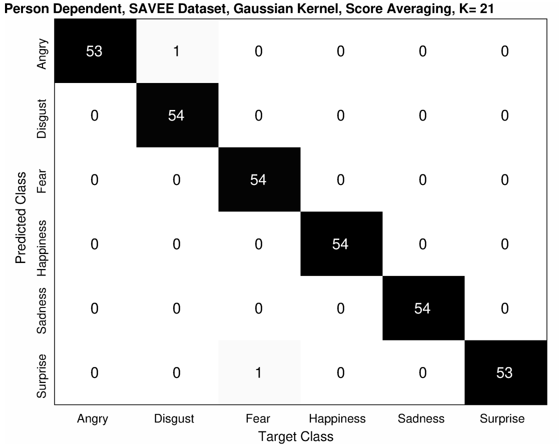
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	53	1	0	0	0	0
Disgust	0	54	0	0	0	0
Fear	0	0	54	0	0	0
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	0	0	0	54

Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 19



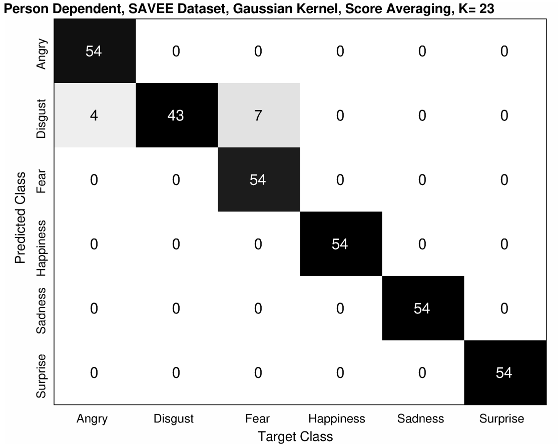
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	54	0	0	0	0	0
Disgust	2	41	2	0	9	0
Fear	0	0	52	0	0	2
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	0	0	0	54

Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 21

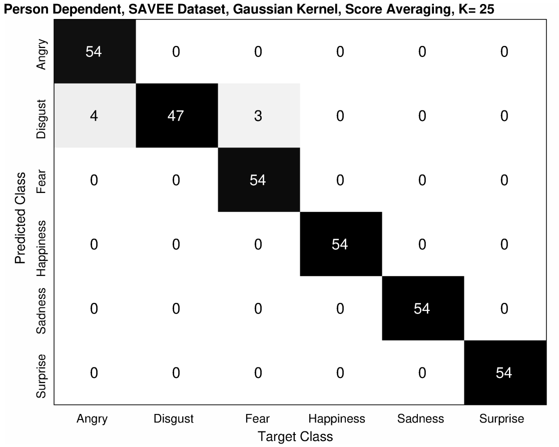


	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	53	1	0	0	0	0
Disgust	0	54	0	0	0	0
Fear	0	0	54	0	0	0
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	1	0	0	53

Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 23

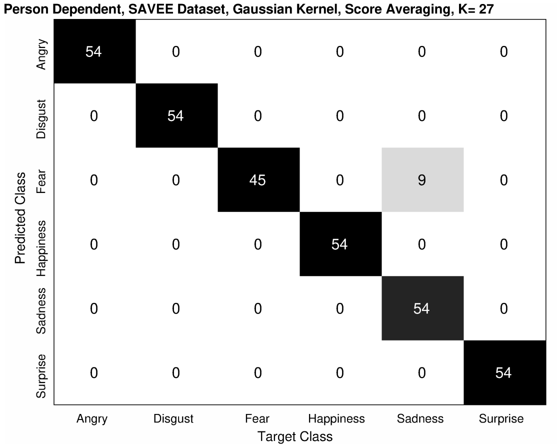


	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	54	0	0	0	0	0
Disgust	4	43	7	0	0	0
Fear	0	0	54	0	0	0
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	0	0	0	54



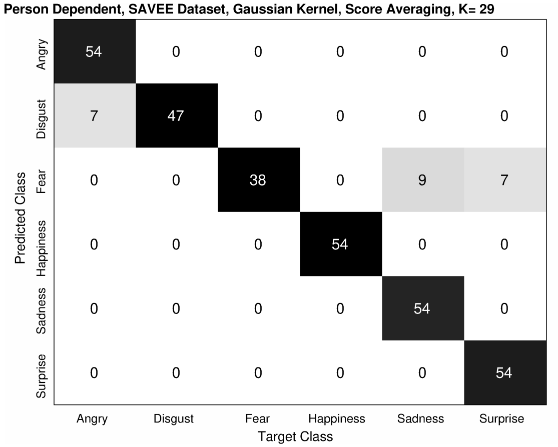
Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 25

Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	54	0	0	0	0	0
Disgust	4	47	3	0	0	0
Fear	0	0	54	0	0	0
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	0	0	0	54



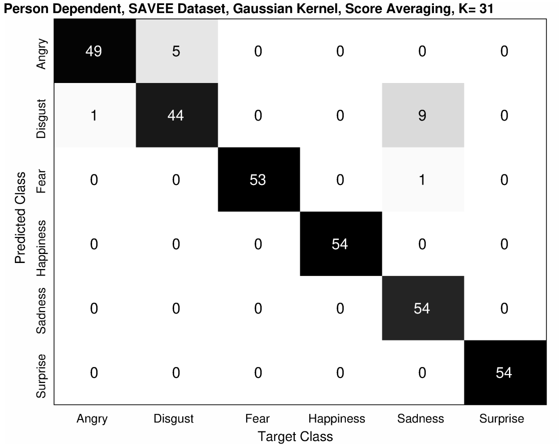
Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 27

Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	54	0	0	0	0	0
Disgust	0	54	0	0	0	0
Fear	0	0	45	0	9	0
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	0	0	0	54



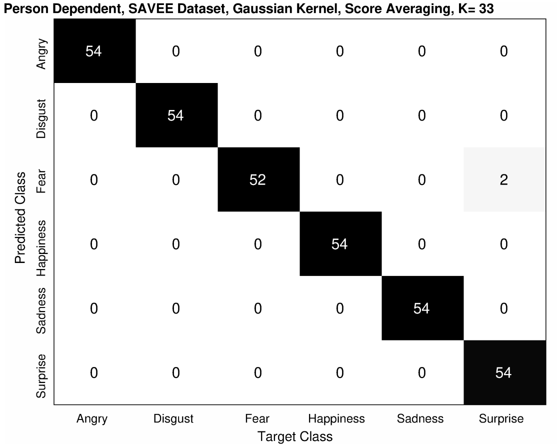
Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 29

Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	54	0	0	0	0	0
Disgust	7	47	0	0	0	0
Fear	0	0	38	0	9	7
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	0	0	0	54



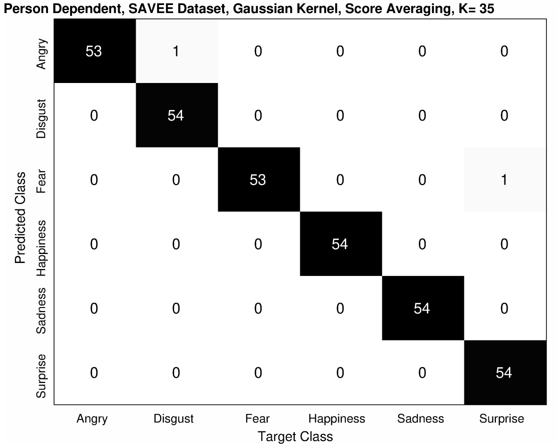
Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 31

Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	49	5	0	0	0	0
Disgust	1	44	0	0	9	0
Fear	0	0	53	0	1	0
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	0	0	0	54



Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 33

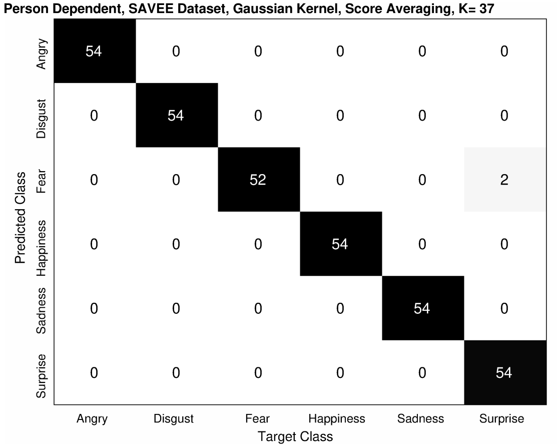
Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	54	0	0	0	0	0
Disgust	0	54	0	0	0	0
Fear	0	0	52	0	0	2
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	0	0	0	54



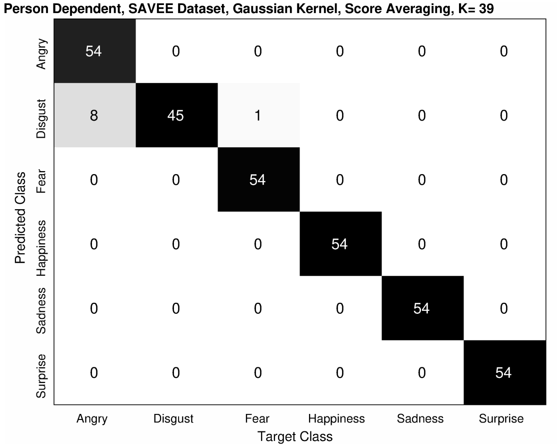
Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 35

Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	53	1	0	0	0	0
Disgust	0	54	0	0	0	0
Fear	0	0	53	0	0	1
Happiness	0	0	0	54	0	0
Sadness	0	0	0	0	54	0
Surprise	0	0	0	0	0	54

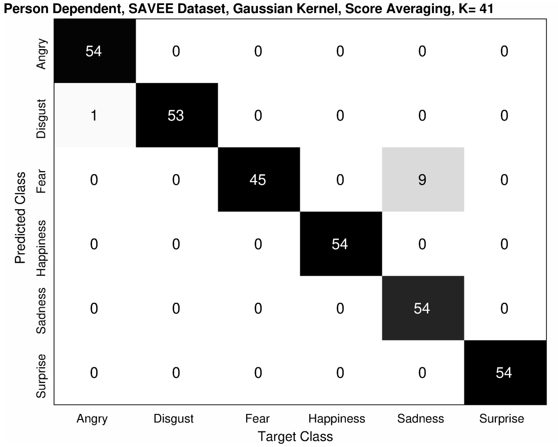
Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 37



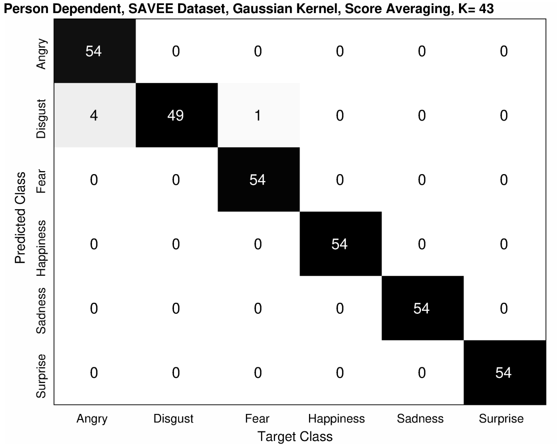
Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 39



Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 41

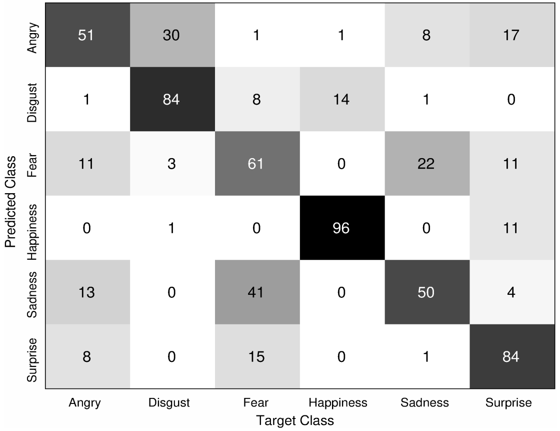


Person Dependent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 43



Appendix C: Confusion matrices (RML, Person-Dependent)

Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 1

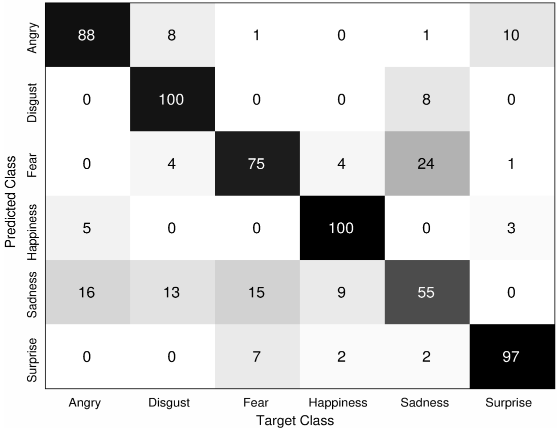


	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	51	30	1	1	8	17
Disgust	1	84	8	14	1	0
Fear	11	3	61	0	22	11
Happiness	0	1	0	96	0	11
Sadness	13	0	41	0	50	4
Surprise	8	0	15	0	1	84

Predicted Class

Target Class

Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 3

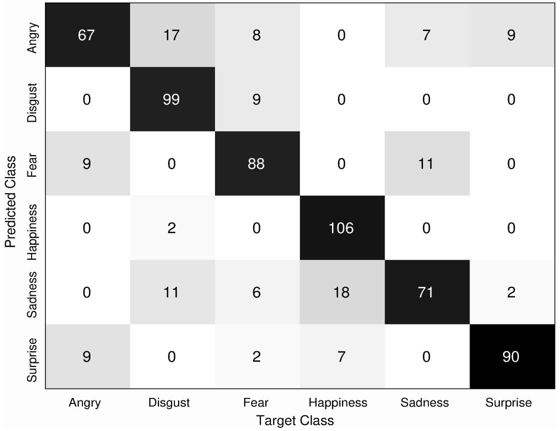


	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	88	8	1	0	1	10
Disgust	0	100	0	0	8	0
Fear	0	4	75	4	24	1
Happiness	5	0	0	100	0	3
Sadness	16	13	15	9	55	0
Surprise	0	0	7	2	2	97

Predicted Class

Target Class

Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 5

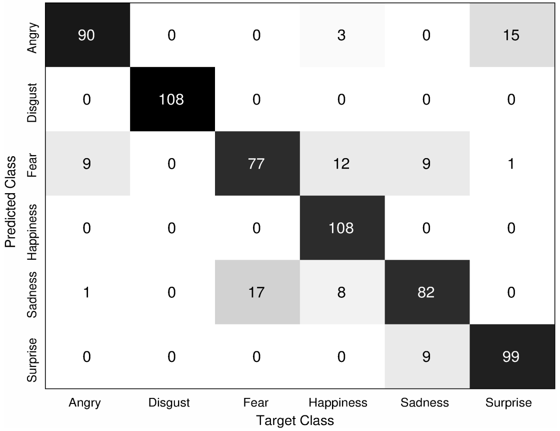


	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	67	17	8	0	7	9
Disgust	0	99	9	0	0	0
Fear	9	0	88	0	11	0
Happiness	0	2	0	106	0	0
Sadness	0	11	6	18	71	2
Surprise	9	0	2	7	0	90

Predicted Class

Target Class

Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 7

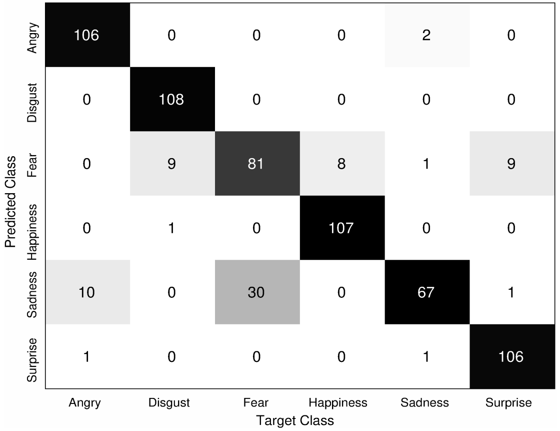


	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	90	0	0	3	0	15
Disgust	0	108	0	0	0	0
Fear	9	0	77	12	9	1
Happiness	0	0	0	108	0	0
Sadness	1	0	17	8	82	0
Surprise	0	0	0	0	9	99

Predicted Class

Target Class

Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 9

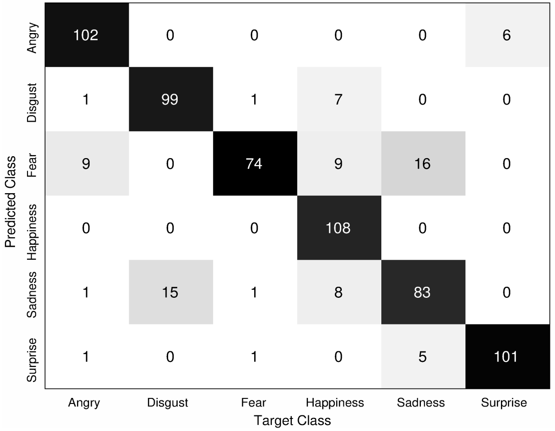


	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	106	0	0	0	2	0
Disgust	0	108	0	0	0	0
Fear	0	9	81	8	1	9
Happiness	0	1	0	107	0	0
Sadness	10	0	30	0	67	1
Surprise	1	0	0	0	1	106

Predicted Class

Target Class

Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 11

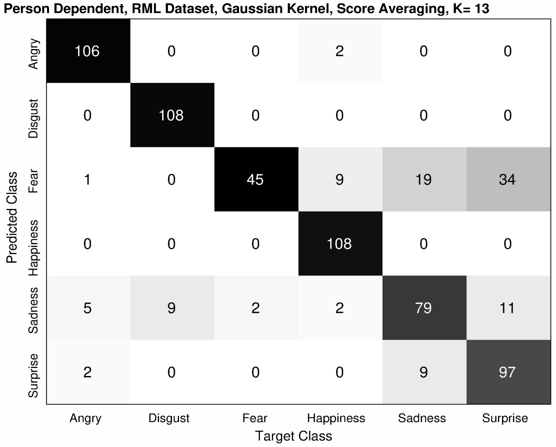


	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	102	0	0	0	0	6
Disgust	1	99	1	7	0	0
Fear	9	0	74	9	16	0
Happiness	0	0	0	108	0	0
Sadness	1	15	1	8	83	0
Surprise	1	0	1	0	5	101

Predicted Class

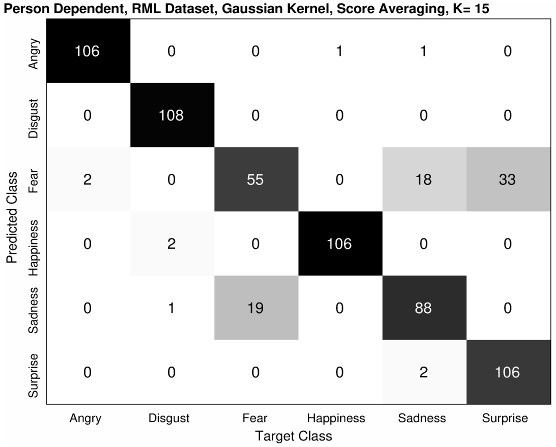
Target Class

Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 13



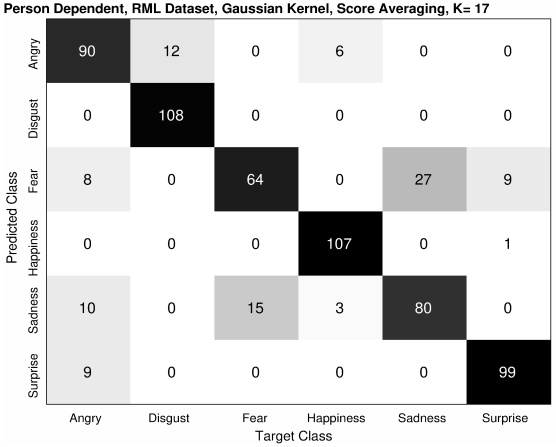
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	106	0	0	2	0	0
Disgust	0	108	0	0	0	0
Fear	1	0	45	9	19	34
Happiness	0	0	0	108	0	0
Sadness	5	9	2	2	79	11
Surprise	2	0	0	0	9	97

Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 15



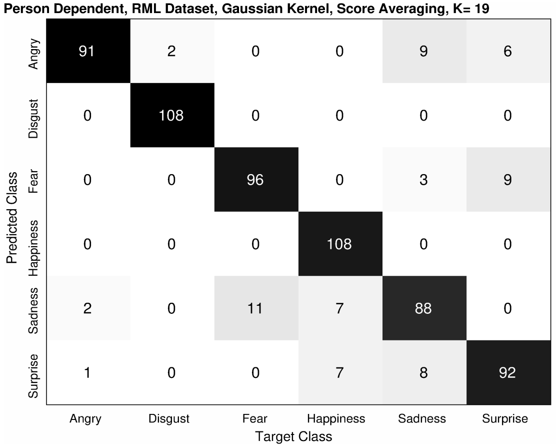
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	106	0	0	1	1	0
Disgust	0	108	0	0	0	0
Fear	2	0	55	0	18	33
Happiness	0	2	0	106	0	0
Sadness	0	1	19	0	88	0
Surprise	0	0	0	0	2	106

Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 17



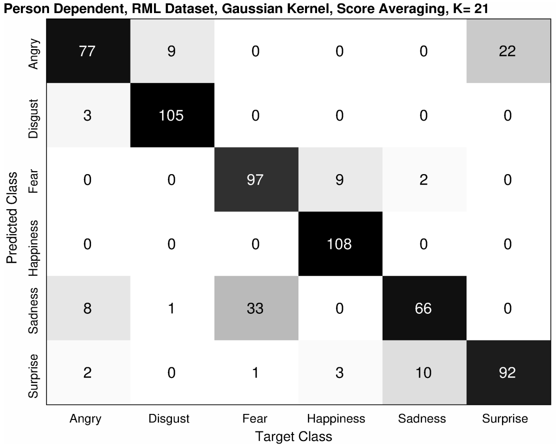
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	90	12	0	6	0	0
Disgust	0	108	0	0	0	0
Fear	8	0	64	0	27	9
Happiness	0	0	0	107	0	1
Sadness	10	0	15	3	80	0
Surprise	9	0	0	0	0	99

Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 19



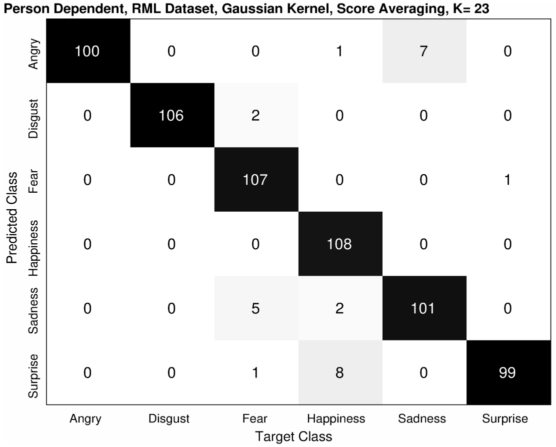
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	91	2	0	0	9	6
Disgust	0	108	0	0	0	0
Fear	0	0	96	0	3	9
Happiness	0	0	0	108	0	0
Sadness	2	0	11	7	88	0
Surprise	1	0	0	7	8	92

Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 21



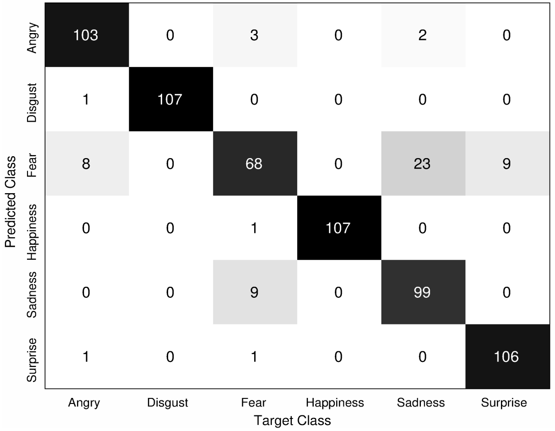
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	77	9	0	0	0	22
Disgust	3	105	0	0	0	0
Fear	0	0	97	9	2	0
Happiness	0	0	0	108	0	0
Sadness	8	1	33	0	66	0
Surprise	2	0	1	3	10	92

Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 23

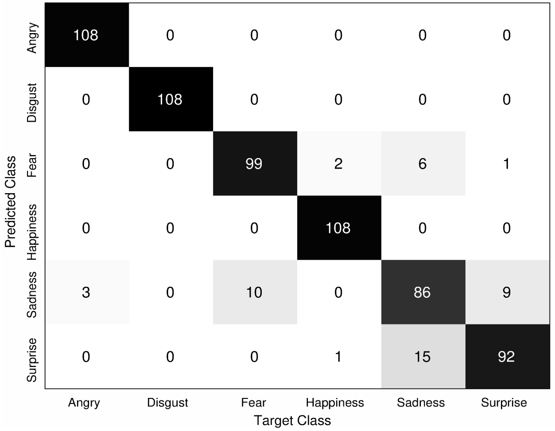


	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	100	0	0	1	7	0
Disgust	0	106	2	0	0	0
Fear	0	0	107	0	0	1
Happiness	0	0	0	108	0	0
Sadness	0	0	5	2	101	0
Surprise	0	0	1	8	0	99

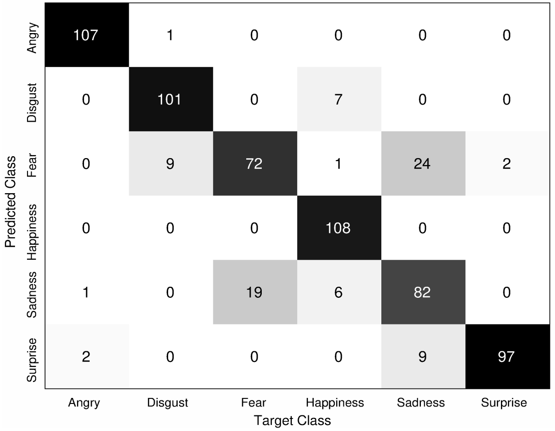
Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 25



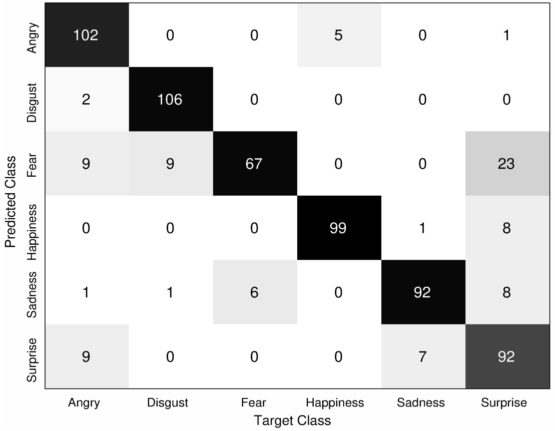
Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 27



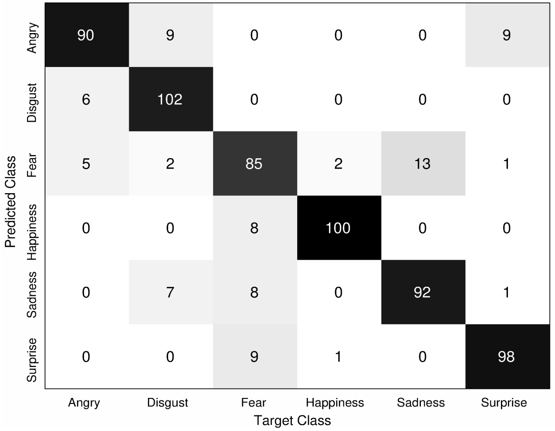
Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 29



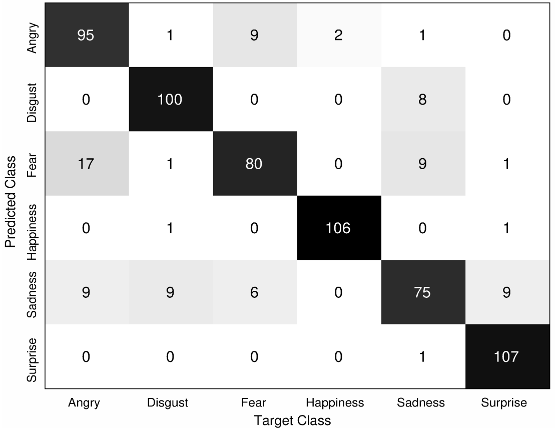
Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 31

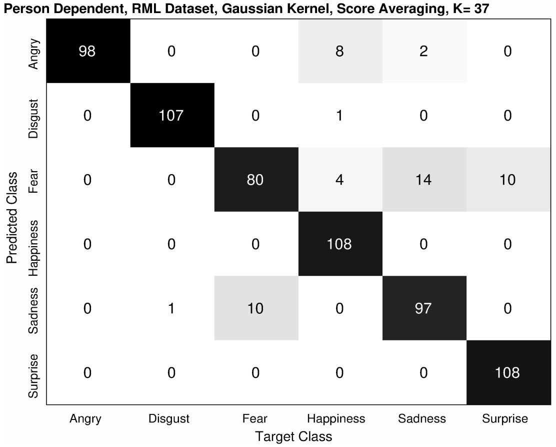


Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 33



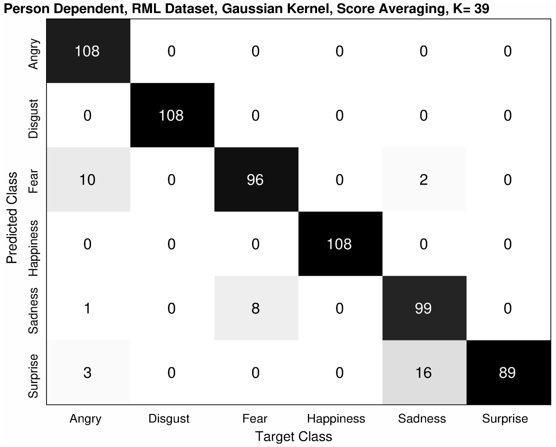
Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 35





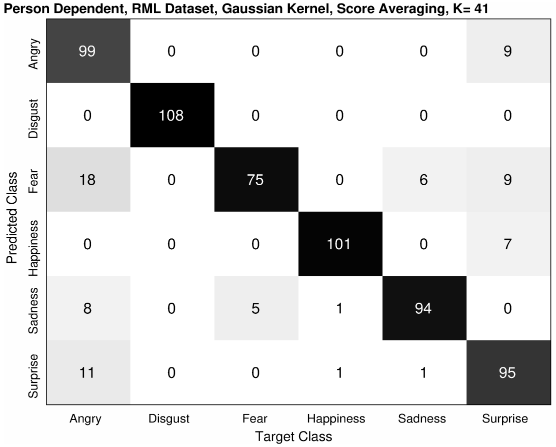
Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 37

Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	98	0	0	8	2	0
Disgust	0	107	0	1	0	0
Fear	0	0	80	4	14	10
Happiness	0	0	0	108	0	0
Sadness	0	1	10	0	97	0
Surprise	0	0	0	0	0	108



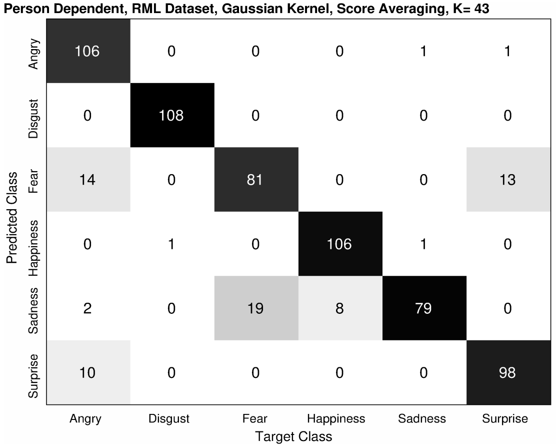
Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 39

Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	108	0	0	0	0	0
Disgust	0	108	0	0	0	0
Fear	10	0	96	0	2	0
Happiness	0	0	0	108	0	0
Sadness	1	0	8	0	99	0
Surprise	3	0	0	0	16	89



Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 41

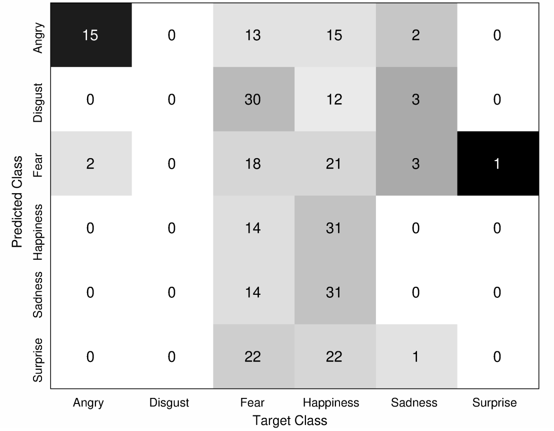
Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	99	0	0	0	0	9
Disgust	0	108	0	0	0	0
Fear	18	0	75	0	6	9
Happiness	0	0	0	101	0	7
Sadness	8	0	5	1	94	0
Surprise	11	0	0	1	1	95



Person Dependent, RML Dataset, Gaussian Kernel, Score Averaging, K= 43

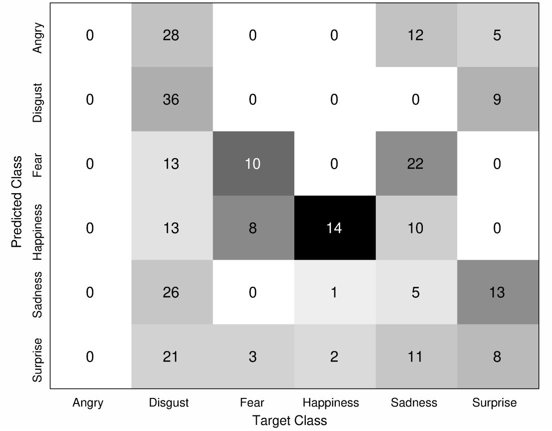
Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	106	0	0	0	1	1
Disgust	0	108	0	0	0	0
Fear	14	0	81	0	0	13
Happiness	0	1	0	106	1	0
Sadness	2	0	19	8	79	0
Surprise	10	0	0	0	0	98

Appendix D: Confusion matrices (SAVEE, Person-Independent)



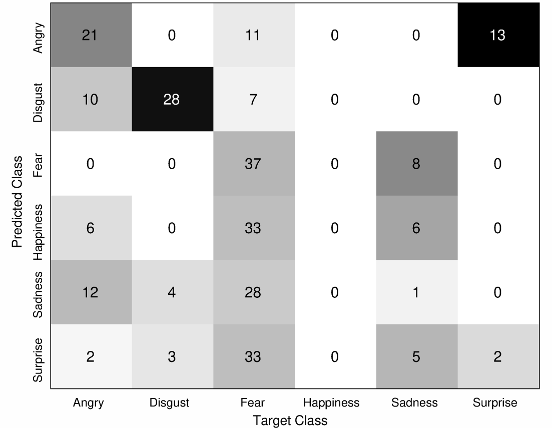
Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 1

Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	15	0	13	15	2	0
Disgust	0	0	30	12	3	0
Fear	2	0	18	21	3	1
Happiness	0	0	14	31	0	0
Sadness	0	0	14	31	0	0
Surprise	0	0	22	22	1	0



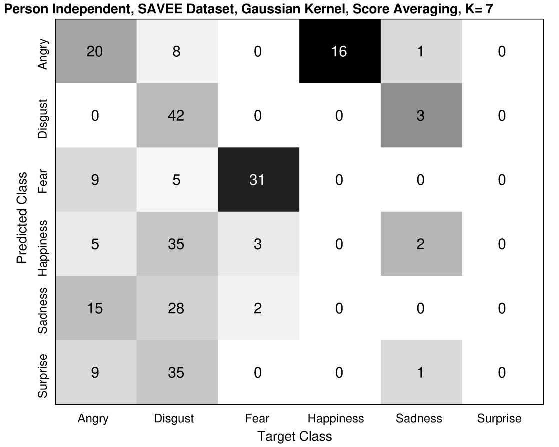
Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 3

Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	0	28	0	0	12	5
Disgust	0	36	0	0	0	9
Fear	0	13	10	0	22	0
Happiness	0	13	8	14	10	0
Sadness	0	26	0	1	5	13
Surprise	0	21	3	2	11	8



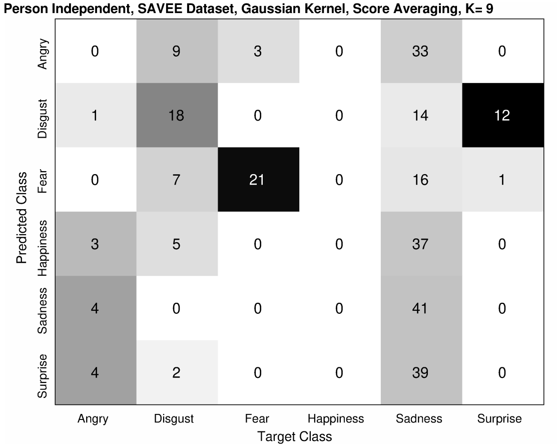
Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 5

Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	21	0	11	0	0	13
Disgust	10	28	7	0	0	0
Fear	0	0	37	0	8	0
Happiness	6	0	33	0	6	0
Sadness	12	4	28	0	1	0
Surprise	2	3	33	0	5	2



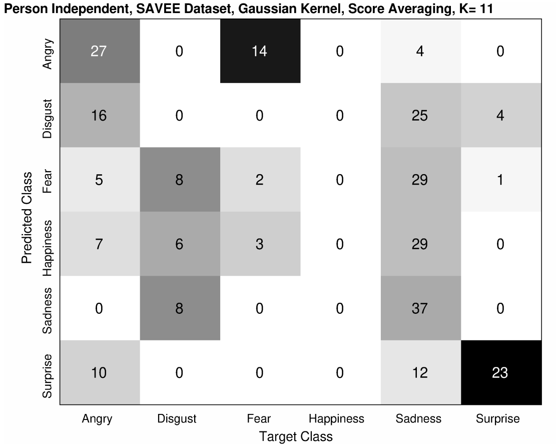
Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 7

Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	20	8	0	16	1	0
Disgust	0	42	0	0	3	0
Fear	9	5	31	0	0	0
Happiness	5	35	3	0	2	0
Sadness	15	28	2	0	0	0
Surprise	9	35	0	0	1	0



Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 9

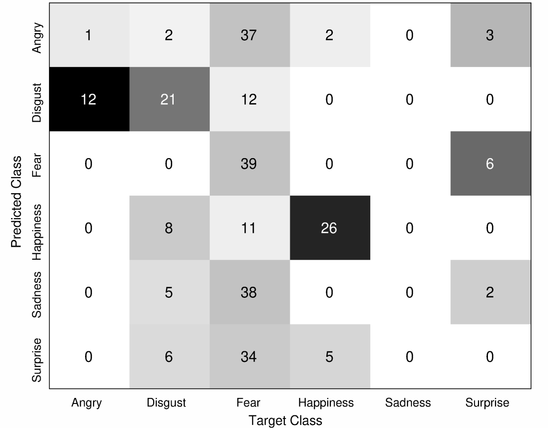
Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	0	9	3	0	33	0
Disgust	1	18	0	0	14	12
Fear	0	7	21	0	16	1
Happiness	3	5	0	0	37	0
Sadness	4	0	0	0	41	0
Surprise	4	2	0	0	39	0



Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 11

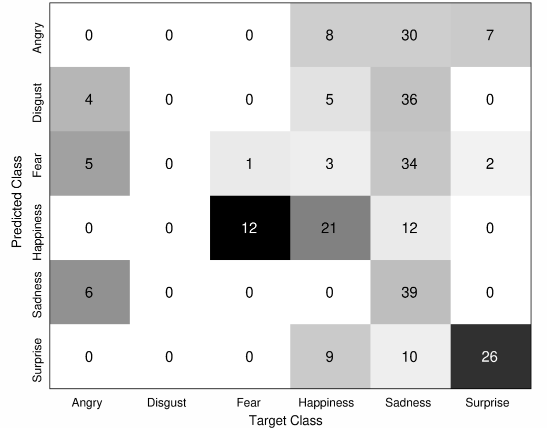
Predicted Class \ Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	27	0	14	0	4	0
Disgust	16	0	0	0	25	4
Fear	5	8	2	0	29	1
Happiness	7	6	3	0	29	0
Sadness	0	8	0	0	37	0
Surprise	10	0	0	0	12	23

Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 13



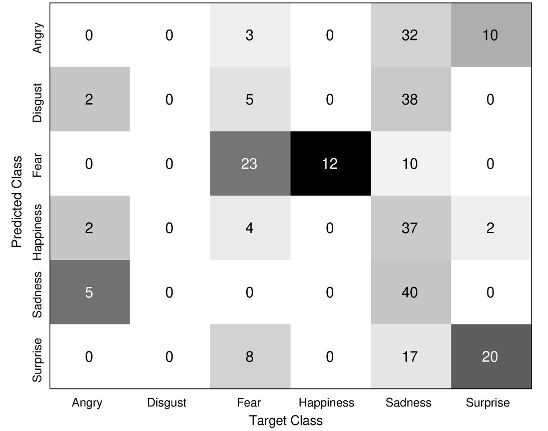
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	1	2	37	2	0	3
Disgust	12	21	12	0	0	0
Fear	0	0	39	0	0	6
Happiness	0	8	11	26	0	0
Sadness	0	5	38	0	0	2
Surprise	0	6	34	5	0	0

Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 15



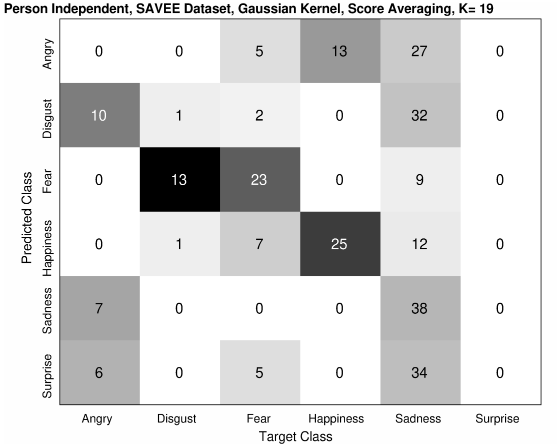
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	0	0	0	8	30	7
Disgust	4	0	0	5	36	0
Fear	5	0	1	3	34	2
Happiness	0	0	12	21	12	0
Sadness	6	0	0	0	39	0
Surprise	0	0	0	9	10	26

Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 17



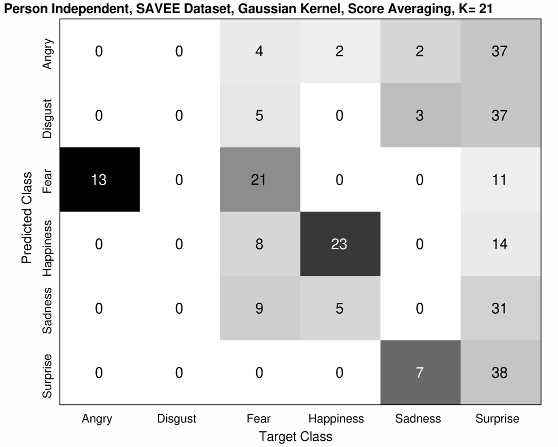
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	0	0	3	0	32	10
Disgust	2	0	5	0	38	0
Fear	0	0	23	12	10	0
Happiness	2	0	4	0	37	2
Sadness	5	0	0	0	40	0
Surprise	0	0	8	0	17	20

Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 19



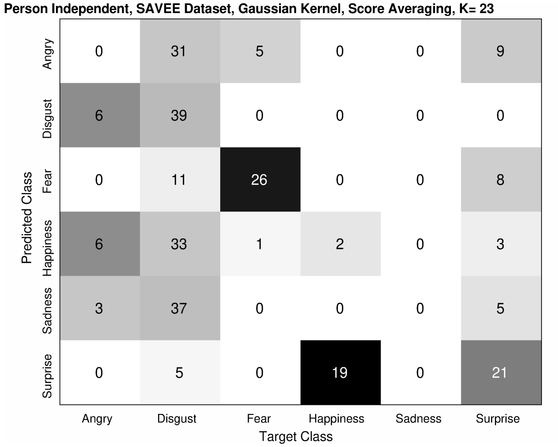
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	0	0	5	13	27	0
Disgust	10	1	2	0	32	0
Fear	0	13	23	0	9	0
Happiness	0	1	7	25	12	0
Sadness	7	0	0	0	38	0
Surprise	6	0	5	0	34	0

Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 21



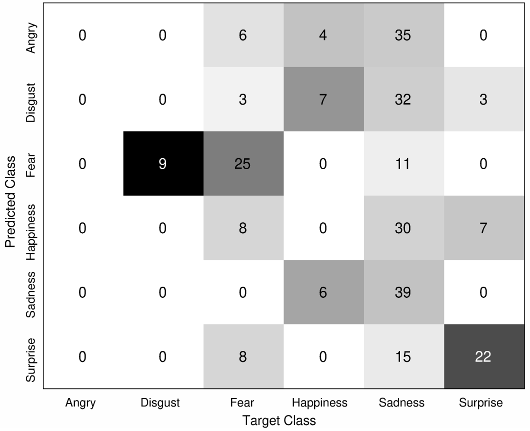
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	0	0	4	2	2	37
Disgust	0	0	5	0	3	37
Fear	13	0	21	0	0	11
Happiness	0	0	8	23	0	14
Sadness	0	0	9	5	0	31
Surprise	0	0	0	0	7	38

Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 23



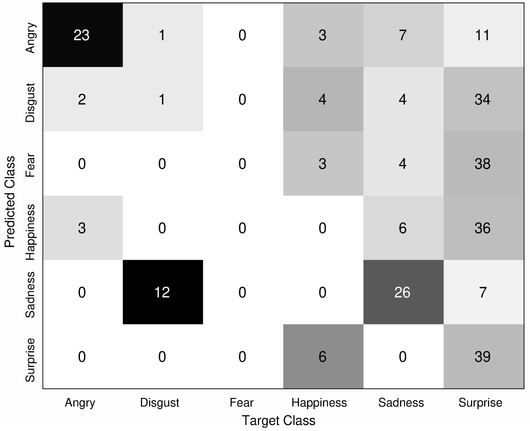
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	0	31	5	0	0	9
Disgust	6	39	0	0	0	0
Fear	0	11	26	0	0	8
Happiness	6	33	1	2	0	3
Sadness	3	37	0	0	0	5
Surprise	0	5	0	19	0	21

Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 25



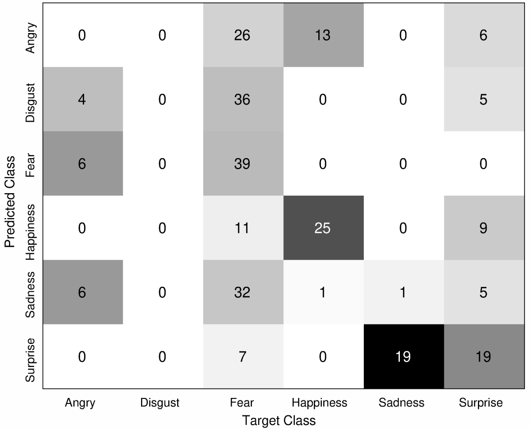
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	0	0	6	4	35	0
Disgust	0	0	3	7	32	3
Fear	0	9	25	0	11	0
Happiness	0	0	8	0	30	7
Sadness	0	0	0	6	39	0
Surprise	0	0	8	0	15	22

Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 27



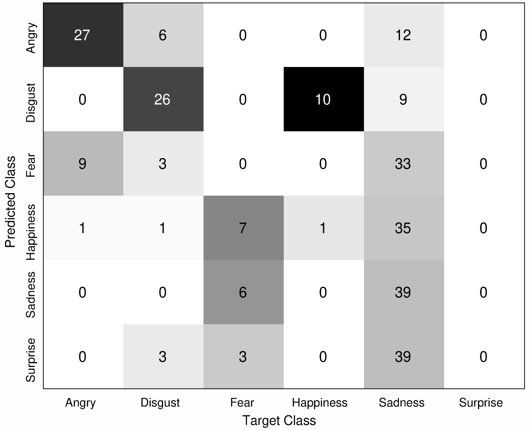
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	23	1	0	3	7	11
Disgust	2	1	0	4	4	34
Fear	0	0	0	3	4	38
Happiness	3	0	0	0	6	36
Sadness	0	12	0	0	26	7
Surprise	0	0	0	6	0	39

Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 29



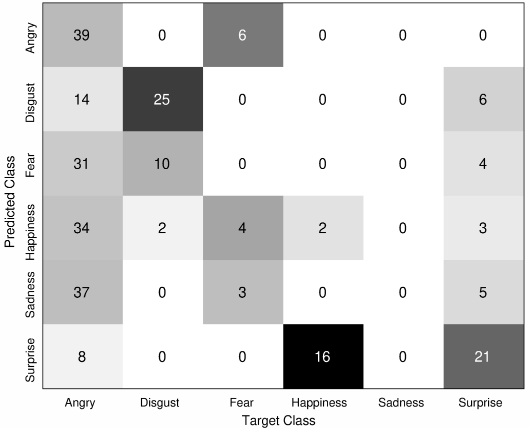
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	0	0	26	13	0	6
Disgust	4	0	36	0	0	5
Fear	6	0	39	0	0	0
Happiness	0	0	11	25	0	9
Sadness	6	0	32	1	1	5
Surprise	0	0	7	0	19	19

Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 31



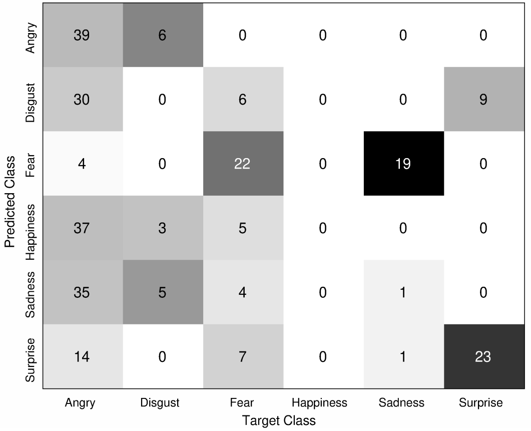
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	27	6	0	0	12	0
Disgust	0	26	0	10	9	0
Fear	9	3	0	0	33	0
Happiness	1	1	7	1	35	0
Sadness	0	0	6	0	39	0
Surprise	0	3	3	0	39	0

Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 33

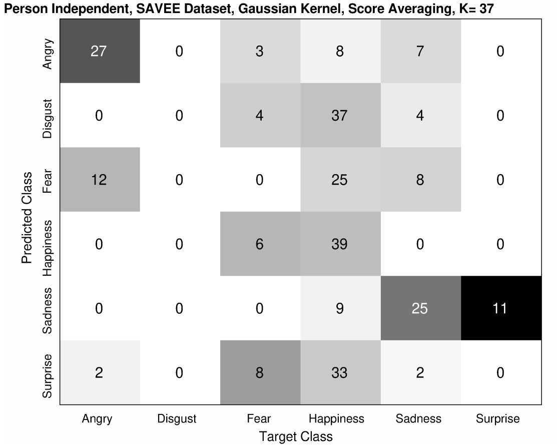


	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	39	0	6	0	0	0
Disgust	14	25	0	0	0	6
Fear	31	10	0	0	0	4
Happiness	34	2	4	2	0	3
Sadness	37	0	3	0	0	5
Surprise	8	0	0	16	0	21

Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 35

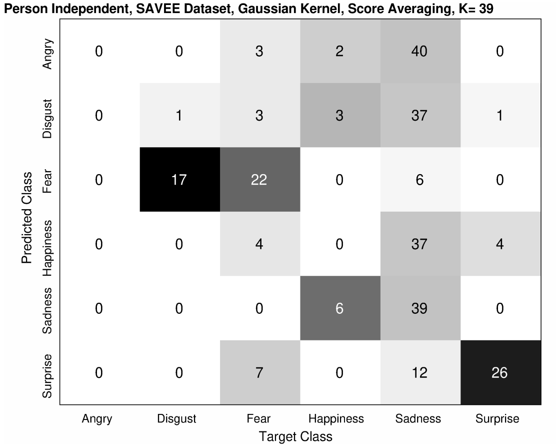


	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	39	6	0	0	0	0
Disgust	30	0	6	0	0	9
Fear	4	0	22	0	19	0
Happiness	37	3	5	0	0	0
Sadness	35	5	4	0	1	0
Surprise	14	0	7	0	1	23



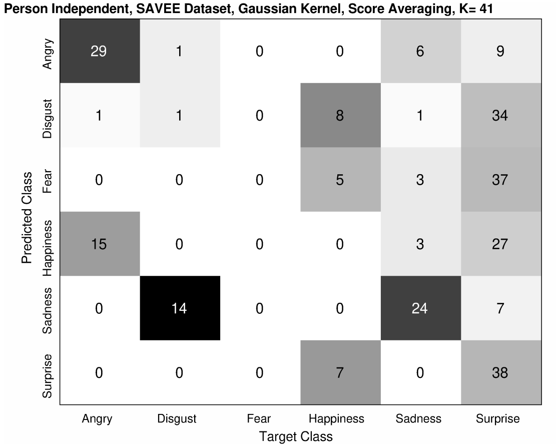
Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 37

	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	27	0	3	8	7	0
Disgust	0	0	4	37	4	0
Fear	12	0	0	25	8	0
Happiness	0	0	6	39	0	0
Sadness	0	0	0	9	25	11
Surprise	2	0	8	33	2	0



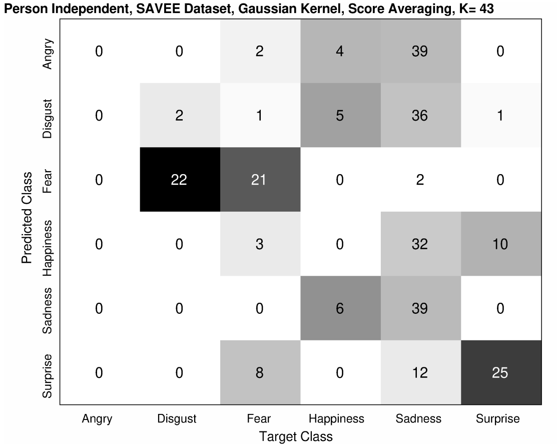
Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 39

	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	0	0	3	2	40	0
Disgust	0	1	3	3	37	1
Fear	0	17	22	0	6	0
Happiness	0	0	4	0	37	4
Sadness	0	0	0	6	39	0
Surprise	0	0	7	0	12	26



Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 41

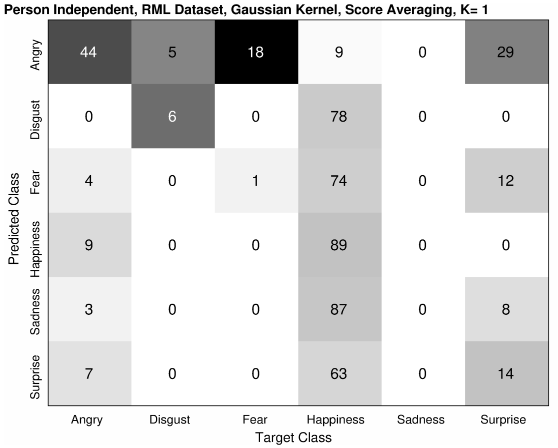
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	29	1	0	0	6	9
Disgust	1	1	0	8	1	34
Fear	0	0	0	5	3	37
Happiness	15	0	0	0	3	27
Sadness	0	14	0	0	24	7
Surprise	0	0	0	7	0	38



Person Independent, SAVEE Dataset, Gaussian Kernel, Score Averaging, K= 43

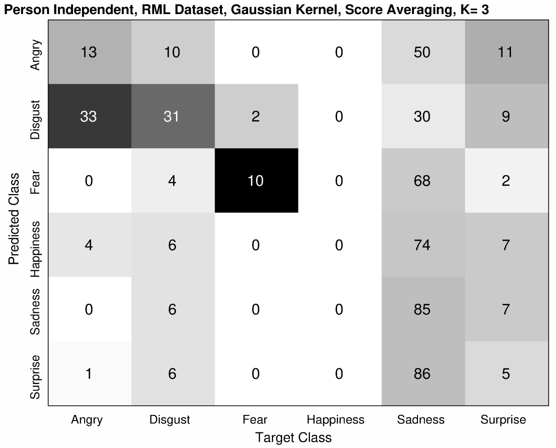
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	0	0	2	4	39	0
Disgust	0	2	1	5	36	1
Fear	0	22	21	0	2	0
Happiness	0	0	3	0	32	10
Sadness	0	0	0	6	39	0
Surprise	0	0	8	0	12	25

Appendix E: Confusion matrices (RML, Person-Independent)



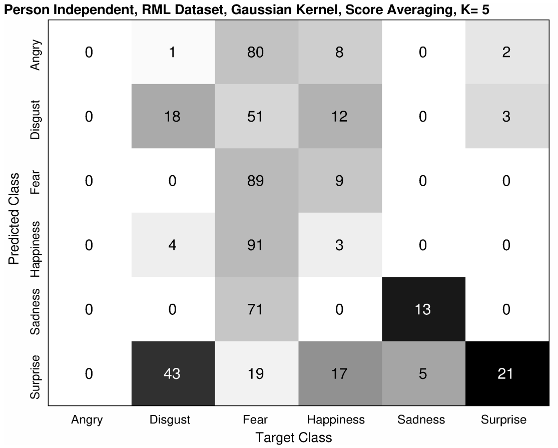
Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 1

	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	44	5	18	9	0	29
Disgust	0	6	0	78	0	0
Fear	4	0	1	74	0	12
Happiness	9	0	0	89	0	0
Sadness	3	0	0	87	0	8
Surprise	7	0	0	63	0	14



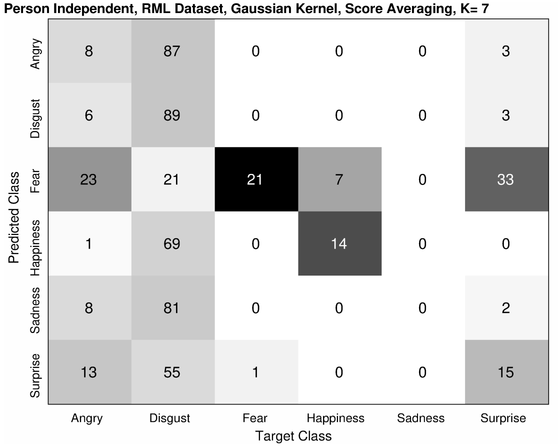
Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 3

	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	13	10	0	0	50	11
Disgust	33	31	2	0	30	9
Fear	0	4	10	0	68	2
Happiness	4	6	0	0	74	7
Sadness	0	6	0	0	85	7
Surprise	1	6	0	0	86	5



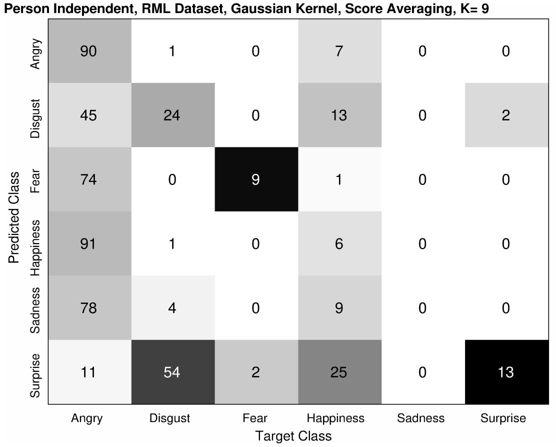
Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 5

	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	0	1	80	8	0	2
Disgust	0	18	51	12	0	3
Fear	0	0	89	9	0	0
Happiness	0	4	91	3	0	0
Sadness	0	0	71	0	13	0
Surprise	0	43	19	17	5	21



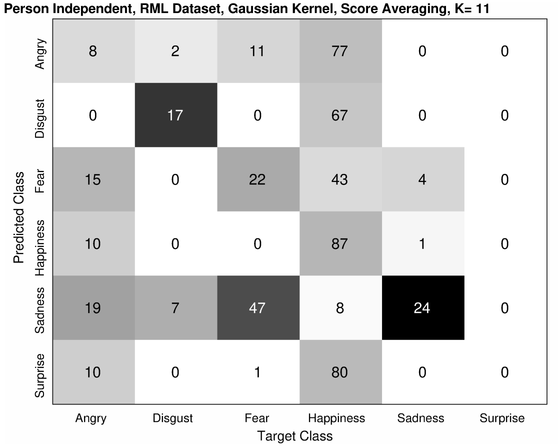
Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 7

	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	8	87	0	0	0	3
Disgust	6	89	0	0	0	3
Fear	23	21	21	7	0	33
Happiness	1	69	0	14	0	0
Sadness	8	81	0	0	0	2
Surprise	13	55	1	0	0	15



Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 9

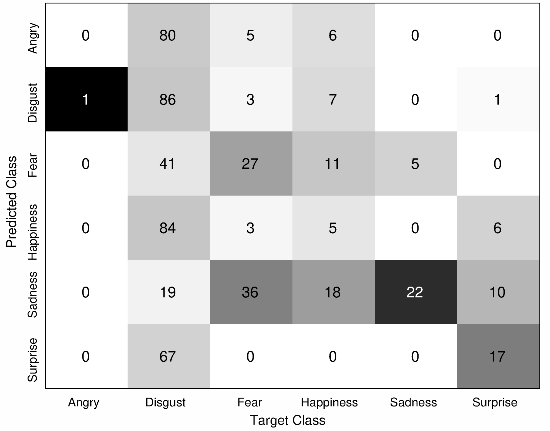
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	90	1	0	7	0	0
Disgust	45	24	0	13	0	2
Fear	74	0	9	1	0	0
Happiness	91	1	0	6	0	0
Sadness	78	4	0	9	0	0
Surprise	11	54	2	25	0	13



Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 11

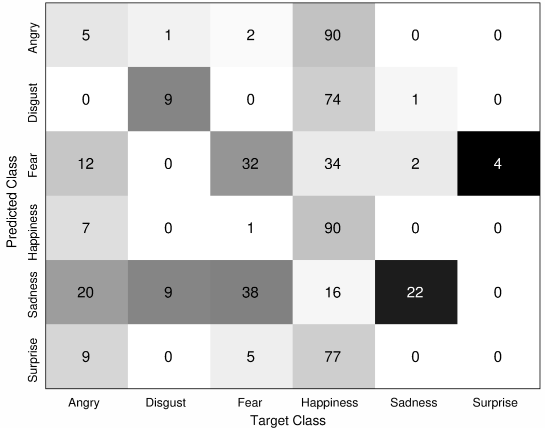
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	8	2	11	77	0	0
Disgust	0	17	0	67	0	0
Fear	15	0	22	43	4	0
Happiness	10	0	0	87	1	0
Sadness	19	7	47	8	24	0
Surprise	10	0	1	80	0	0

Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 13



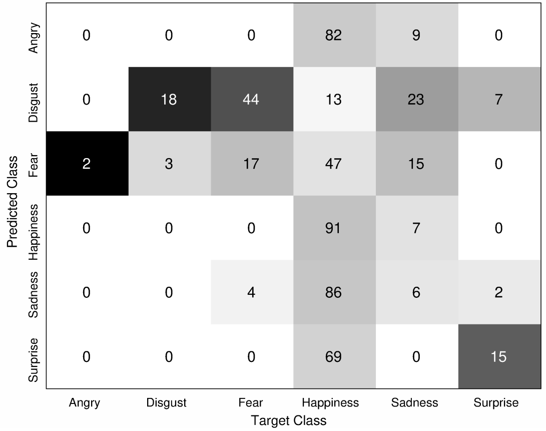
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	0	80	5	6	0	0
Disgust	1	86	3	7	0	1
Fear	0	41	27	11	5	0
Happiness	0	84	3	5	0	6
Sadness	0	19	36	18	22	10
Surprise	0	67	0	0	0	17

Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 15



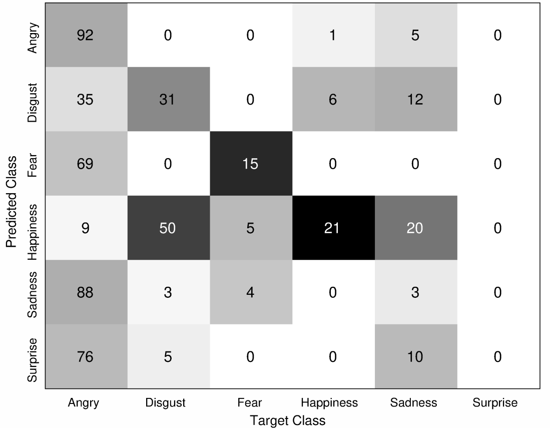
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	5	1	2	90	0	0
Disgust	0	9	0	74	1	0
Fear	12	0	32	34	2	4
Happiness	7	0	1	90	0	0
Sadness	20	9	38	16	22	0
Surprise	9	0	5	77	0	0

Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 17



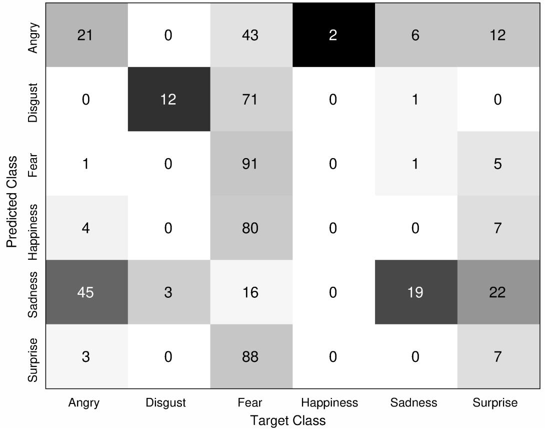
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	0	0	0	82	9	0
Disgust	0	18	44	13	23	7
Fear	2	3	17	47	15	0
Happiness	0	0	0	91	7	0
Sadness	0	0	4	86	6	2
Surprise	0	0	0	69	0	15

Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 19



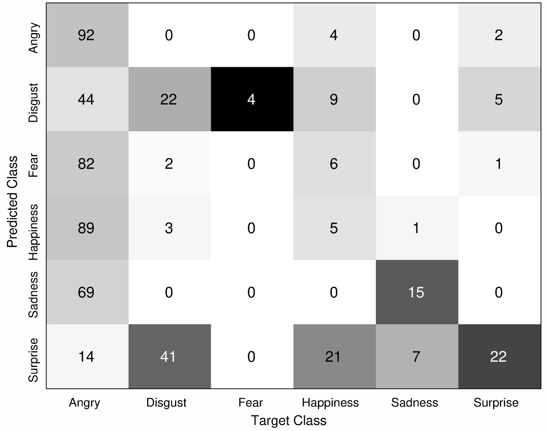
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	92	0	0	1	5	0
Disgust	35	31	0	6	12	0
Fear	69	0	15	0	0	0
Happiness	9	50	5	21	20	0
Sadness	88	3	4	0	3	0
Surprise	76	5	0	0	10	0

Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 21

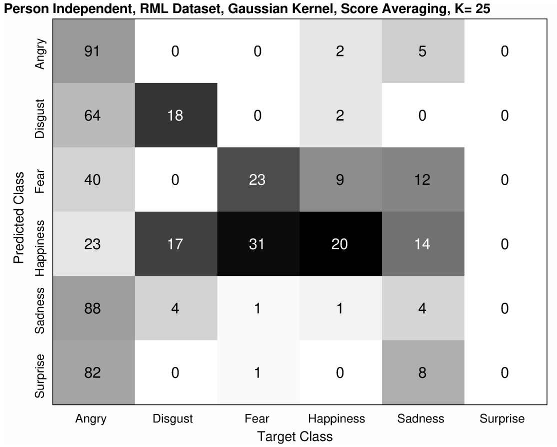


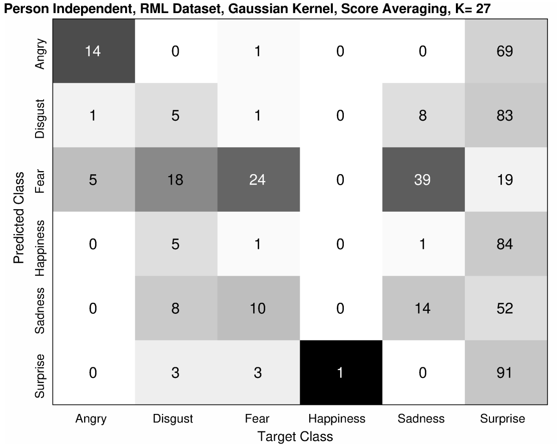
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	21	0	43	2	6	12
Disgust	0	12	71	0	1	0
Fear	1	0	91	0	1	5
Happiness	4	0	80	0	0	7
Sadness	45	3	16	0	19	22
Surprise	3	0	88	0	0	7

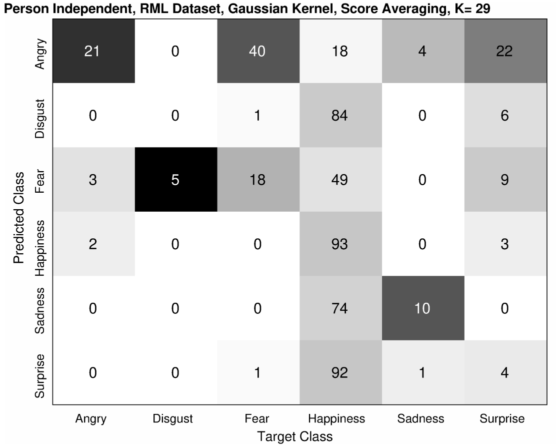
Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 23

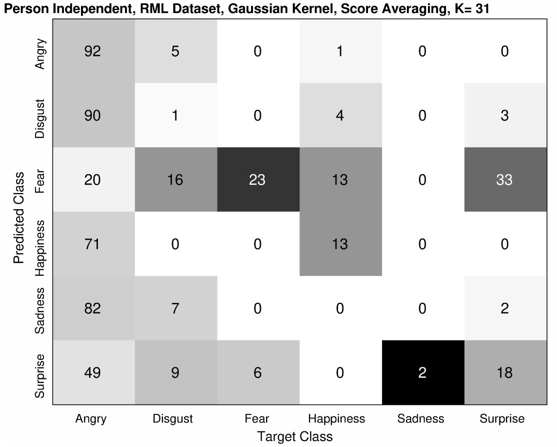


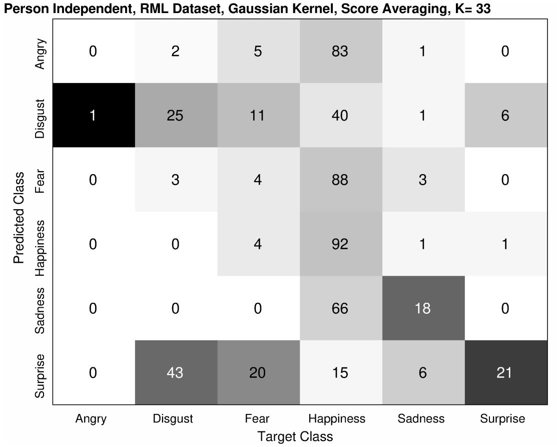
	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	92	0	0	4	0	2
Disgust	44	22	4	9	0	5
Fear	82	2	0	6	0	1
Happiness	89	3	0	5	1	0
Sadness	69	0	0	0	15	0
Surprise	14	41	0	21	7	22

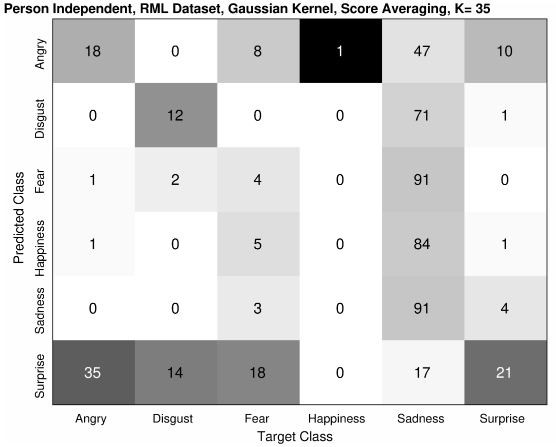

Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 25


Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 27

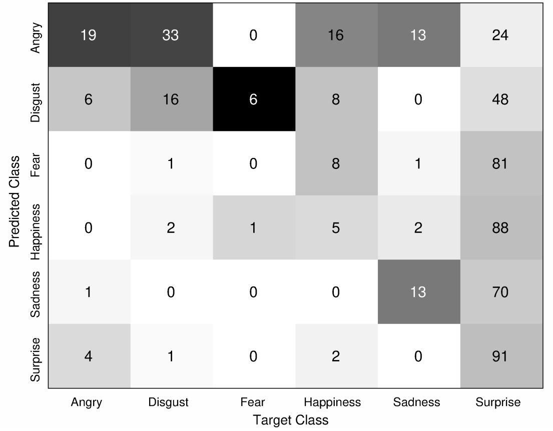

Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 29


Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 31


Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 33

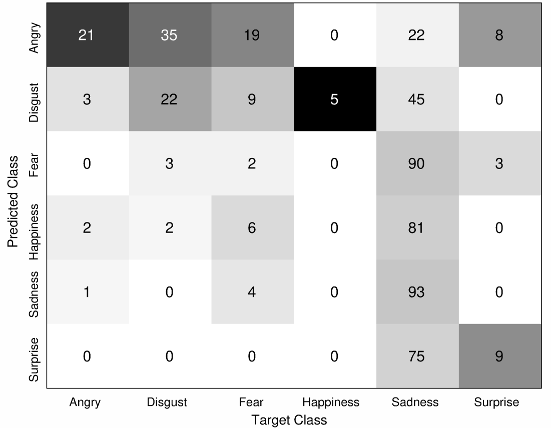

Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 35

Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 37



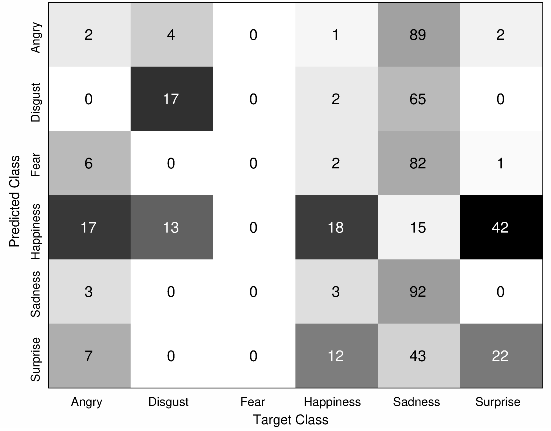
Predicted Class / Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	19	33	0	16	13	24
Disgust	6	16	6	8	0	48
Fear	0	1	0	8	1	81
Happiness	0	2	1	5	2	88
Sadness	1	0	0	0	13	70
Surprise	4	1	0	2	0	91

Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 39



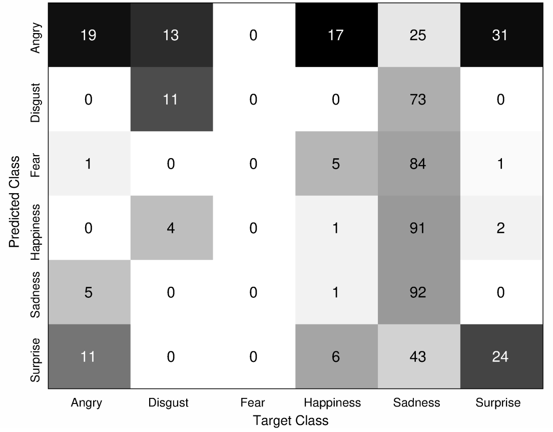
Predicted Class / Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	21	35	19	0	22	8
Disgust	3	22	9	5	45	0
Fear	0	3	2	0	90	3
Happiness	2	2	6	0	81	0
Sadness	1	0	4	0	93	0
Surprise	0	0	0	0	75	9

Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 41



Predicted Class / Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	2	4	0	1	89	2
Disgust	0	17	0	2	65	0
Fear	6	0	0	2	82	1
Happiness	17	13	0	18	15	42
Sadness	3	0	0	3	92	0
Surprise	7	0	0	12	43	22

Person Independent, RML Dataset, Gaussian Kernel, Score Averaging, K= 43



Predicted Class / Target Class	Angry	Disgust	Fear	Happiness	Sadness	Surprise
Angry	19	13	0	17	25	31
Disgust	0	11	0	0	73	0
Fear	1	0	0	5	84	1
Happiness	0	4	0	1	91	2
Sadness	5	0	0	1	92	0
Surprise	11	0	0	6	43	24

Appendix F: ROC Graph (SAVEE, Person-Dependent)

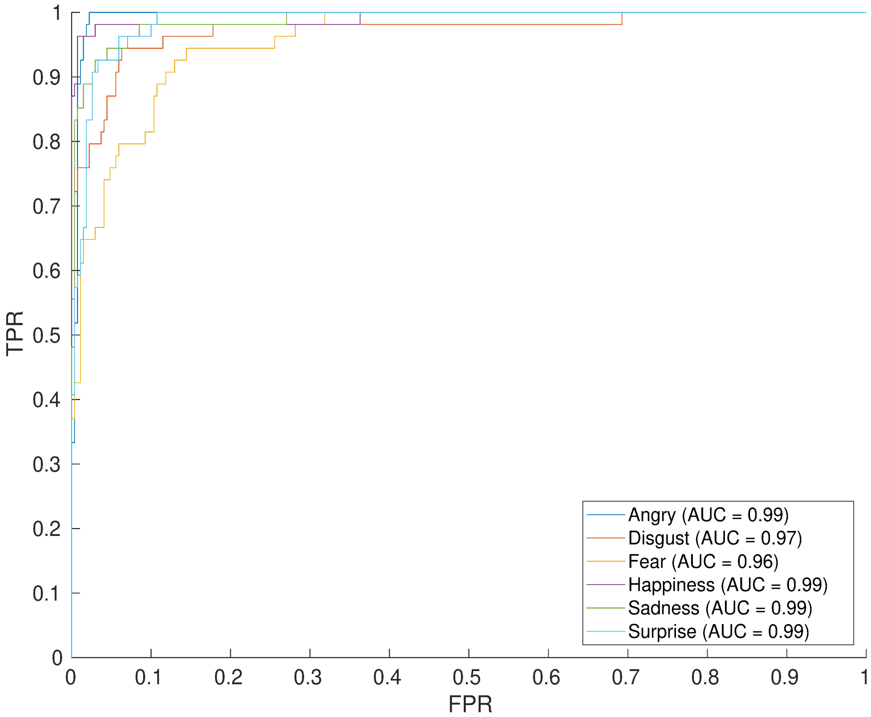


Figure 6.7: Person-dependent ROC graph for SAVEE k = 1.

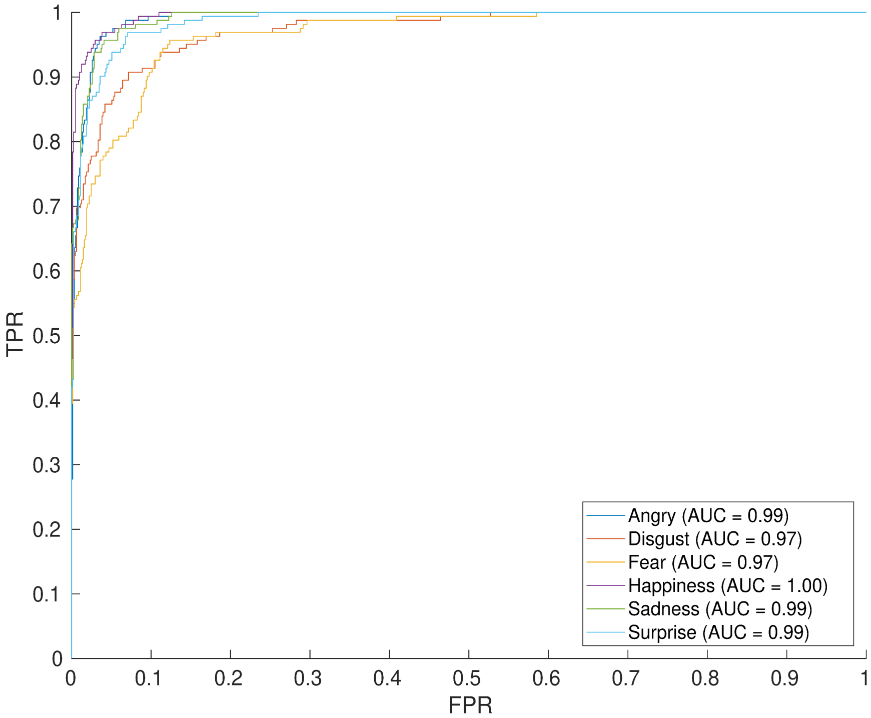


Figure 6.8: Person-dependent ROC graph for SAVEE k = 3.

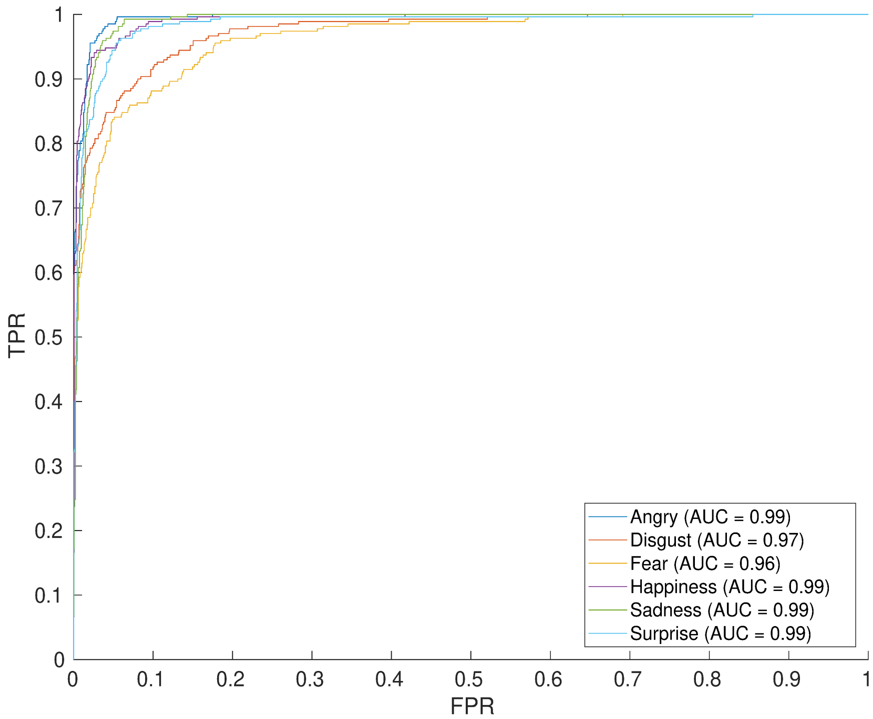


Figure 6.9: Person-dependent ROC graph for SAVEE k = 5.

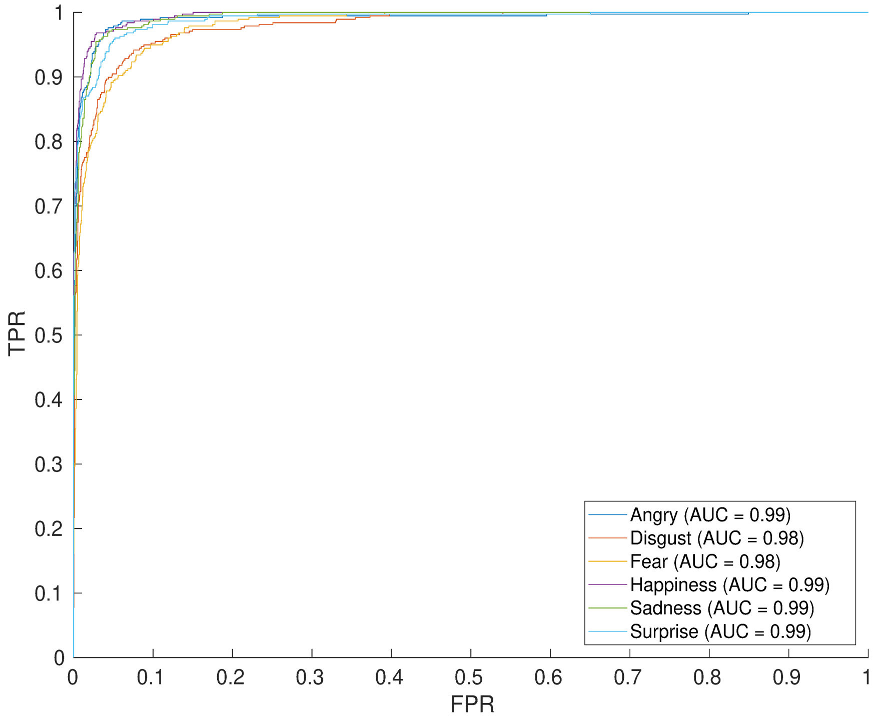


Figure 6.10: Person-dependent ROC graph for SAVEE k = 7.

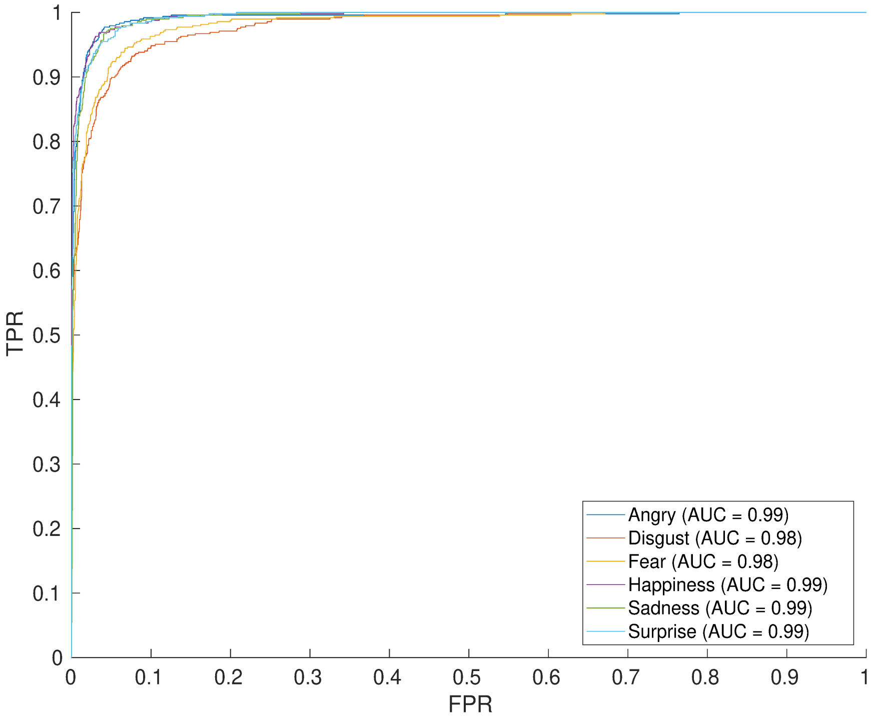


Figure 6.11: Person-dependent ROC graph for SAVEE k = 9.

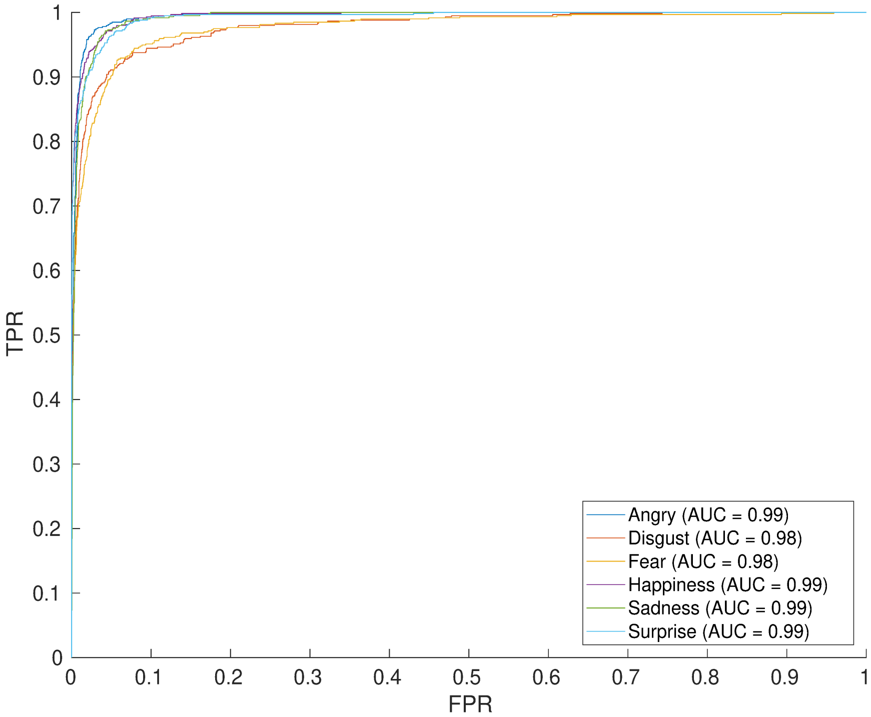


Figure 6.12: Person-dependent ROC graph for SAVEE k = 11.

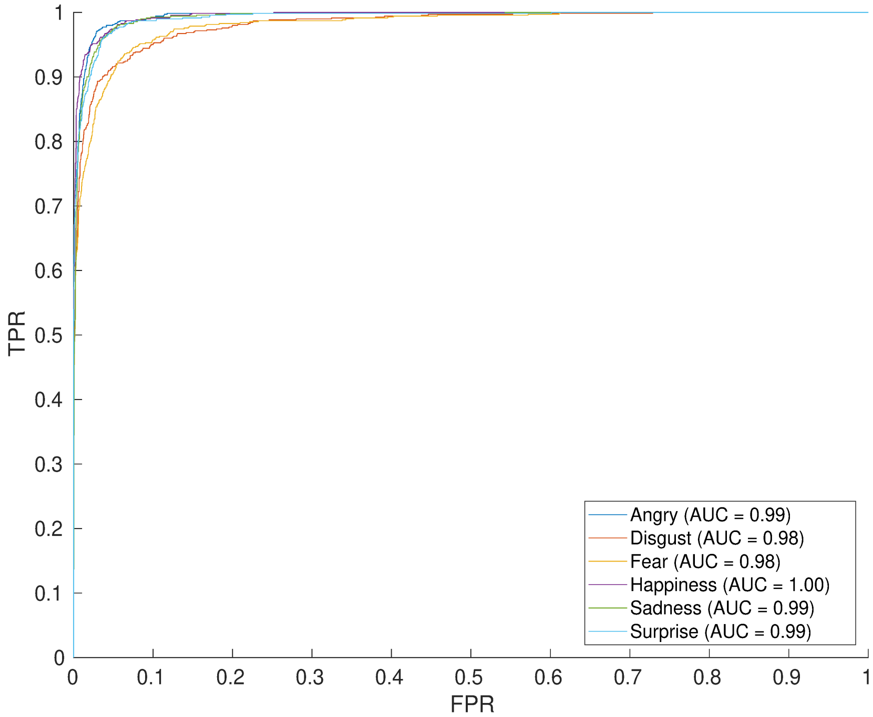


Figure 6.13: Person-dependent ROC graph for SAVEE k = 13.

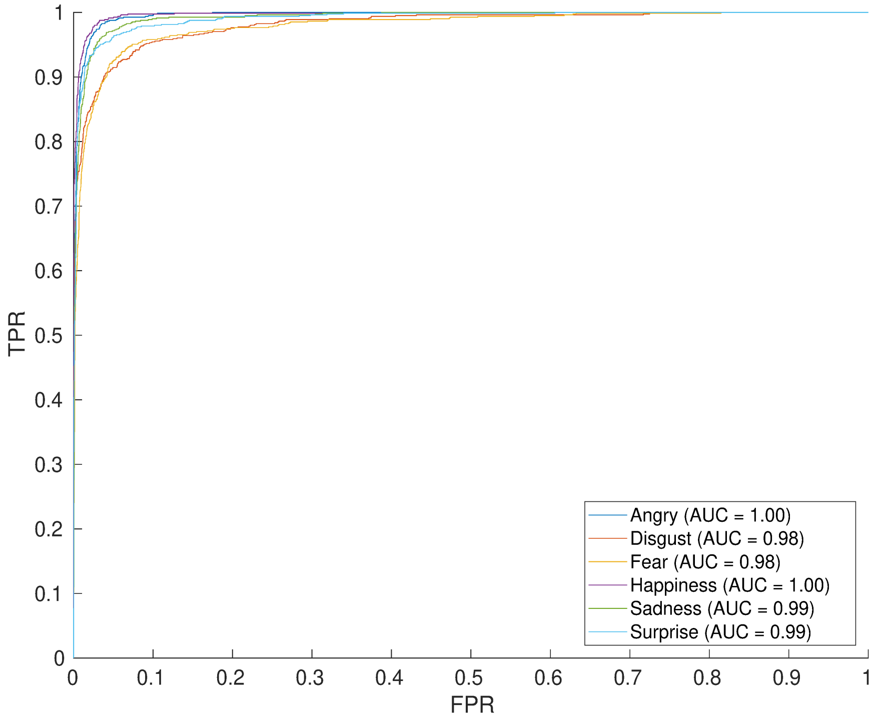


Figure 6.14: Person-dependent ROC graph for SAVEE k = 15.

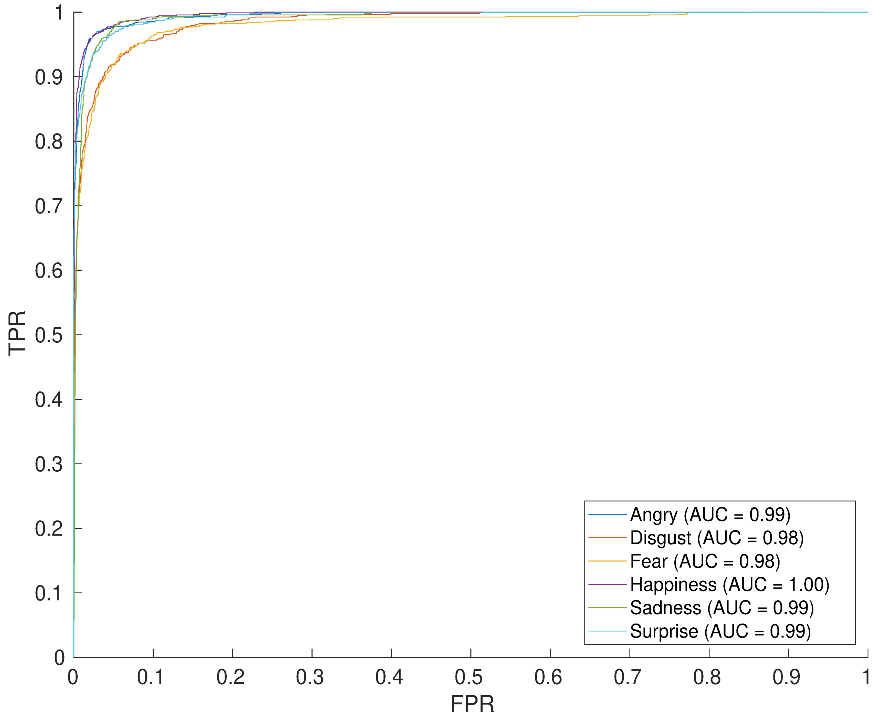


Figure 6.15: Person-dependent ROC graph for SAVEE k = 17.

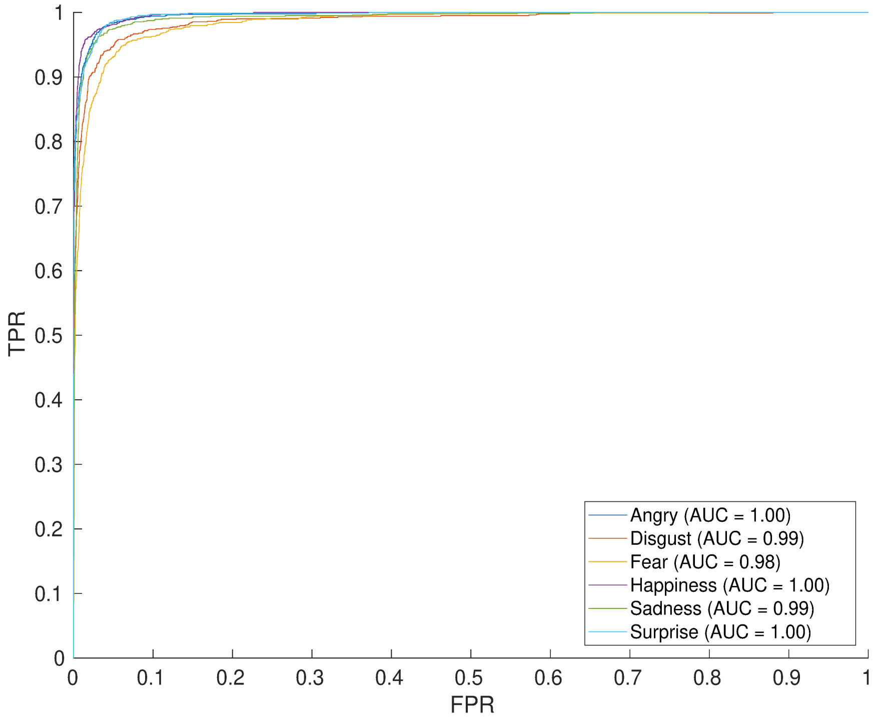


Figure 6.16: Person-dependent ROC graph for SAVEE k = 19.

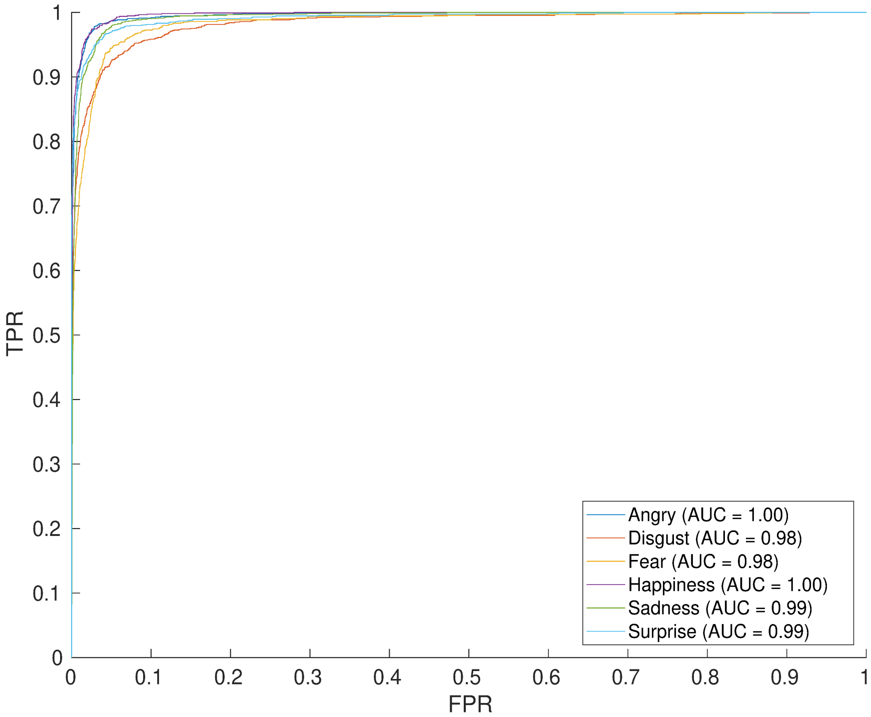


Figure 6.17: Person-dependent ROC graph for SAVEE k = 21.

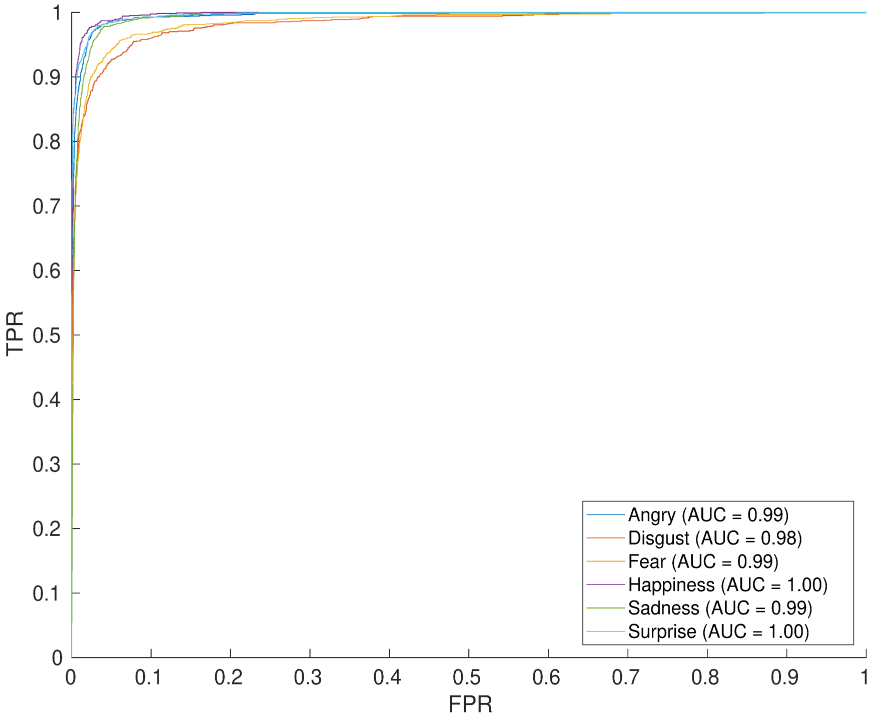


Figure 6.18: Person-dependent ROC graph for SAVEE k = 23.

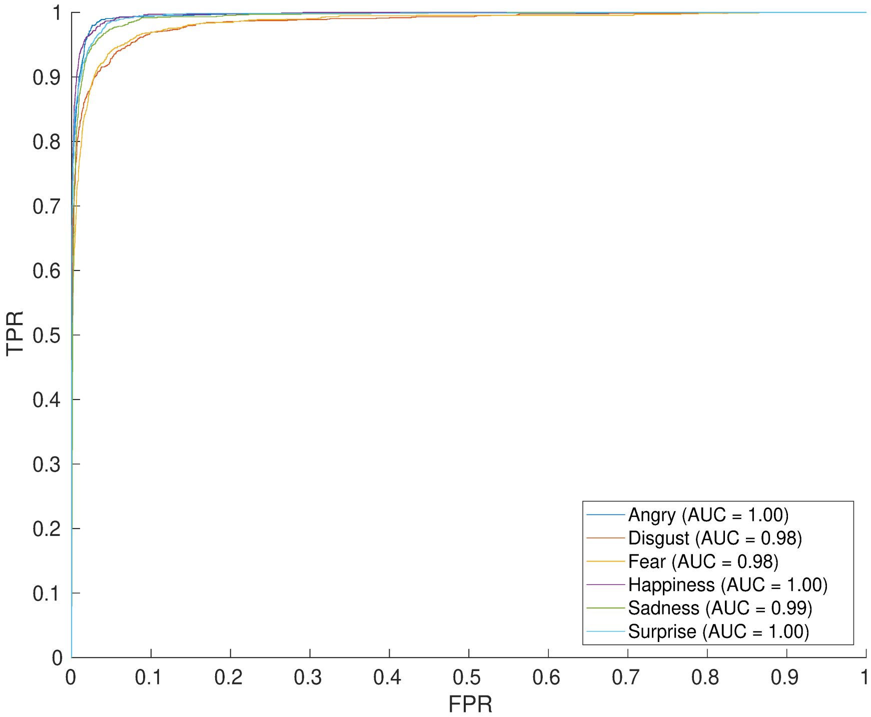


Figure 6.19: Person-dependent ROC graph for SAVEE k = 25.

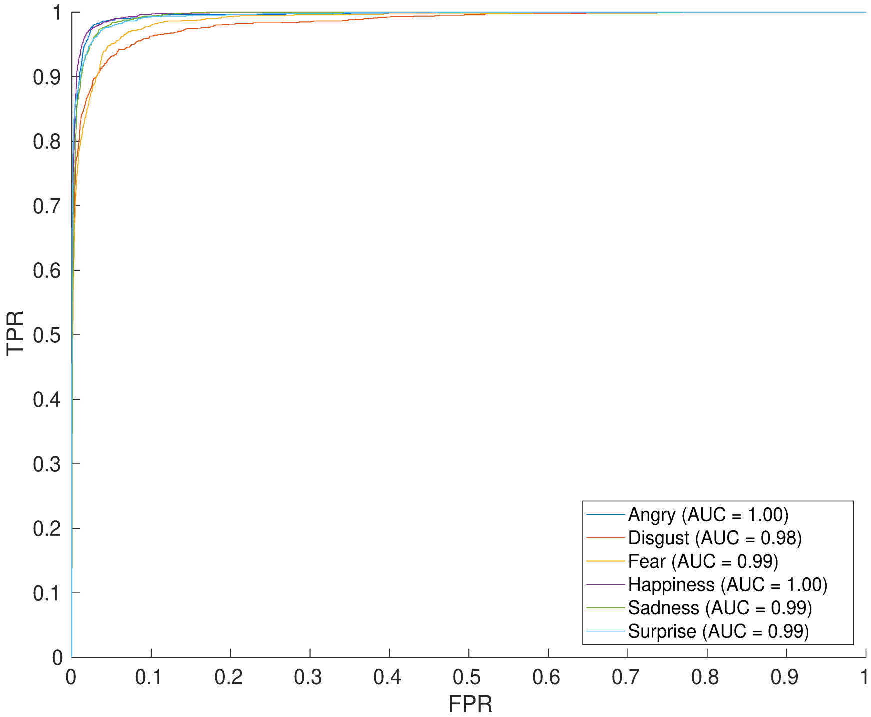


Figure 6.20: Person-dependent ROC graph for SAVEE k = 27.

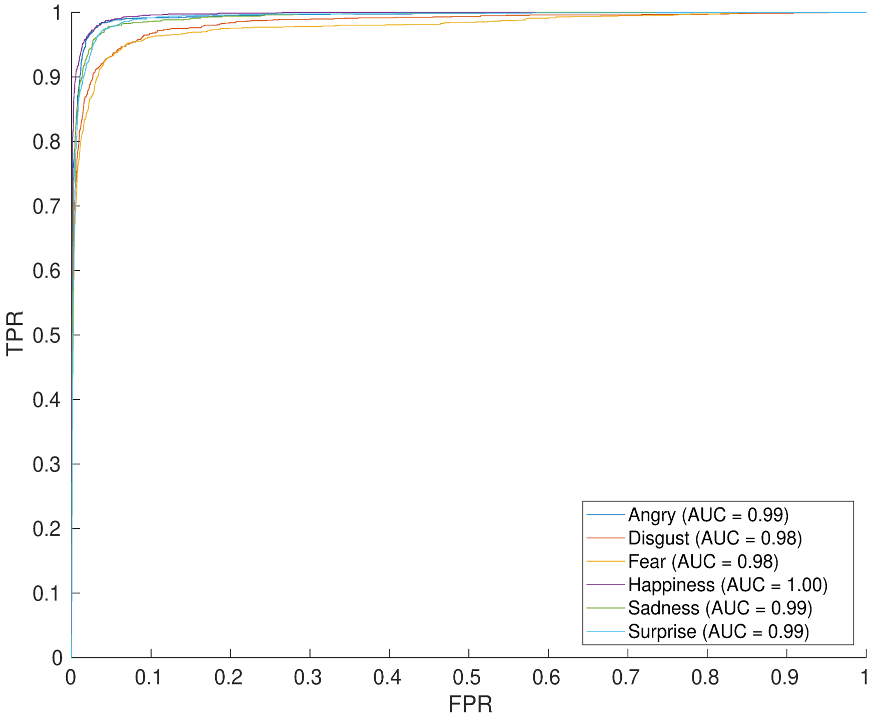


Figure 6.21: Person-dependent ROC graph for SAVEE k = 29.

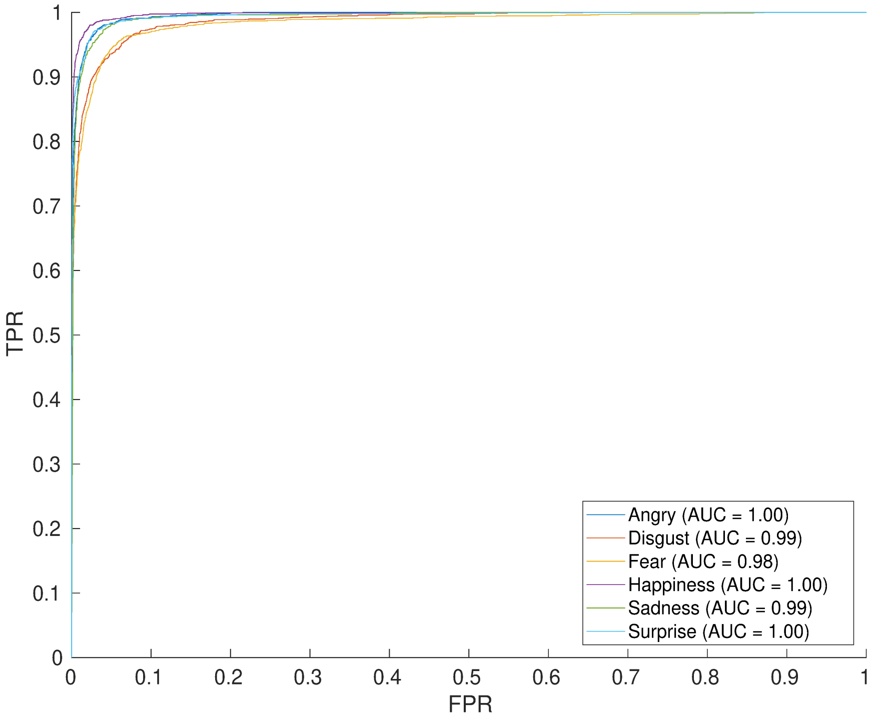


Figure 6.22: Person-dependent ROC graph for SAVEE k = 31.

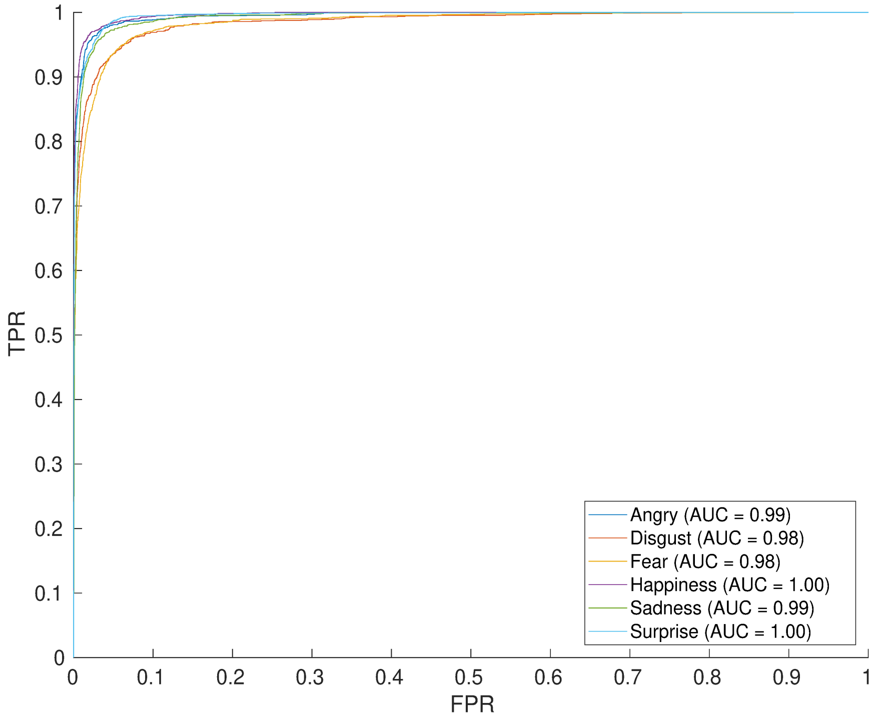


Figure 6.23: Person-dependent ROC graph for SAVEE k = 33.

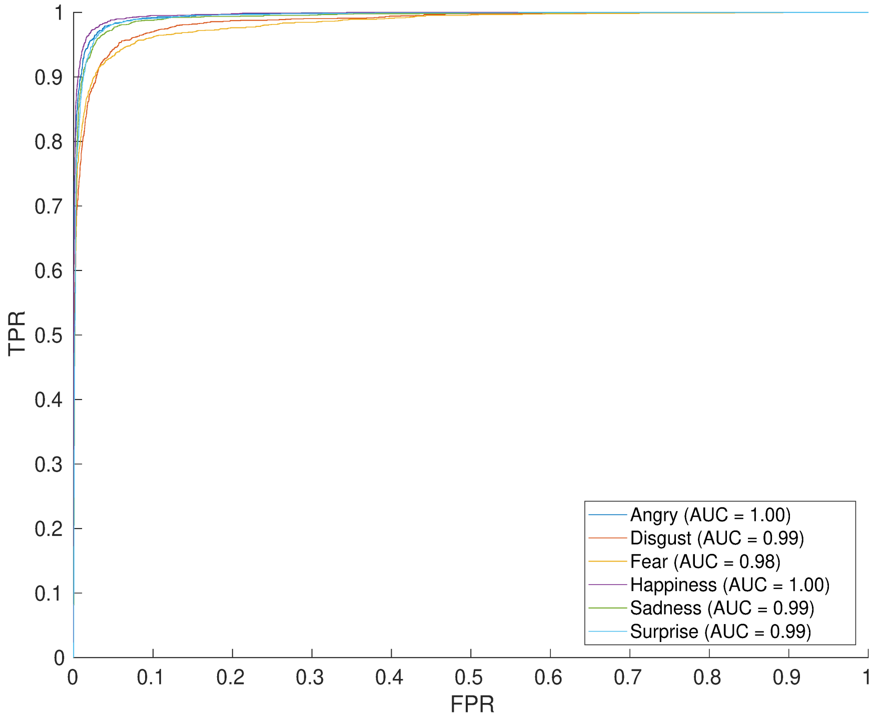


Figure 6.24: Person-dependent ROC graph for SAVEE k = 35.

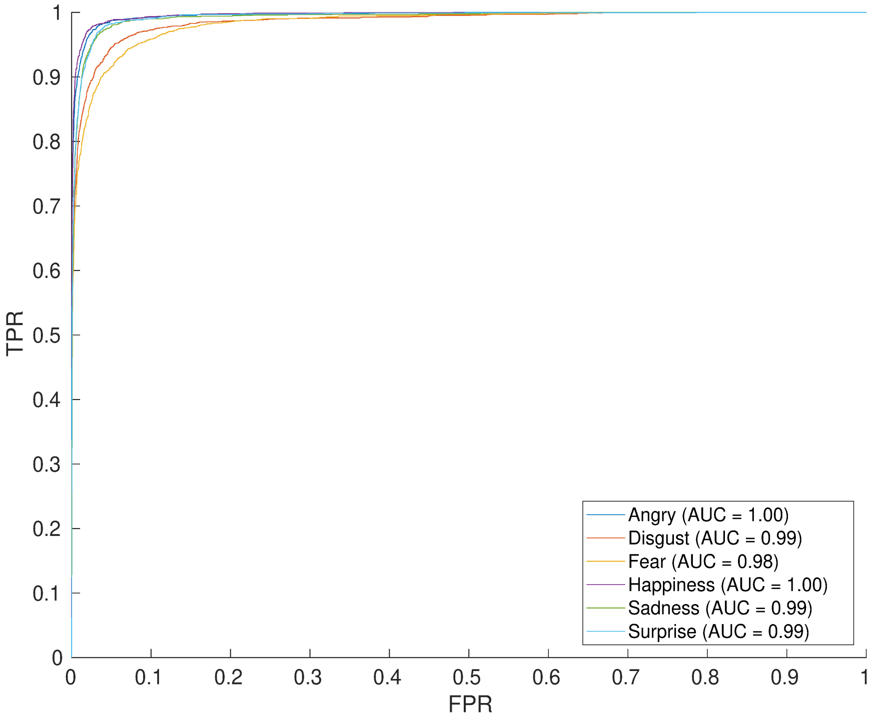


Figure 6.25: Person-dependent ROC graph for SAVEE k = 37.

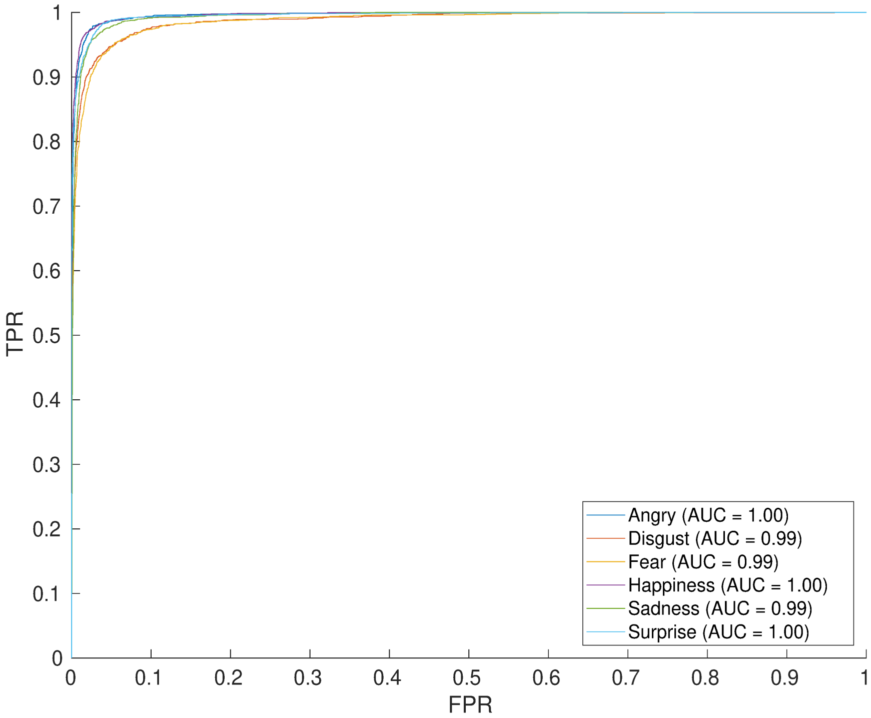


Figure 6.26: Person-dependent ROC graph for SAVEE k = 39.

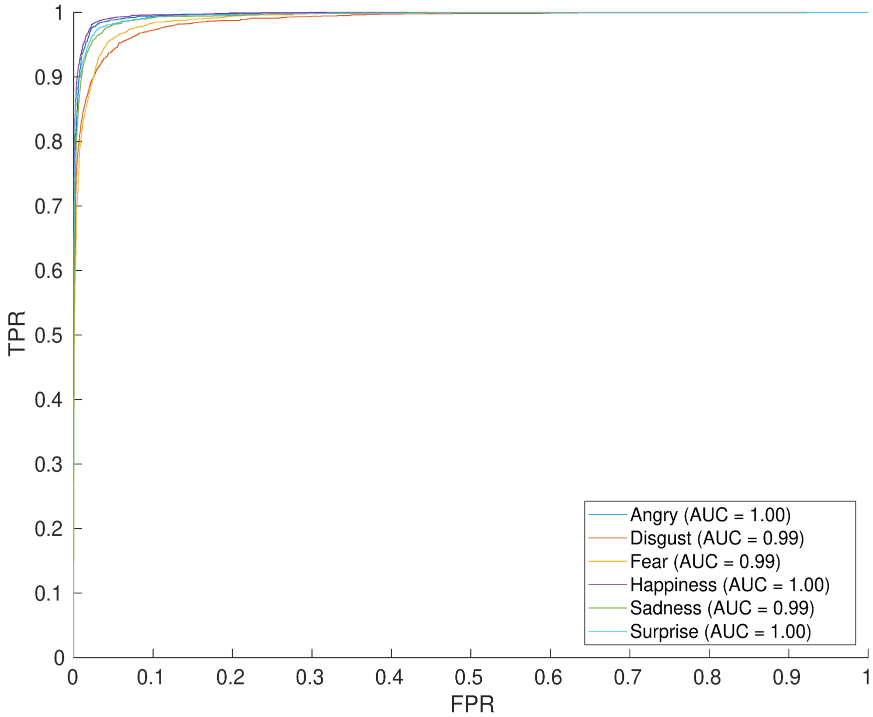


Figure 6.27: Person-dependent ROC graph for SAVEE k = 41.

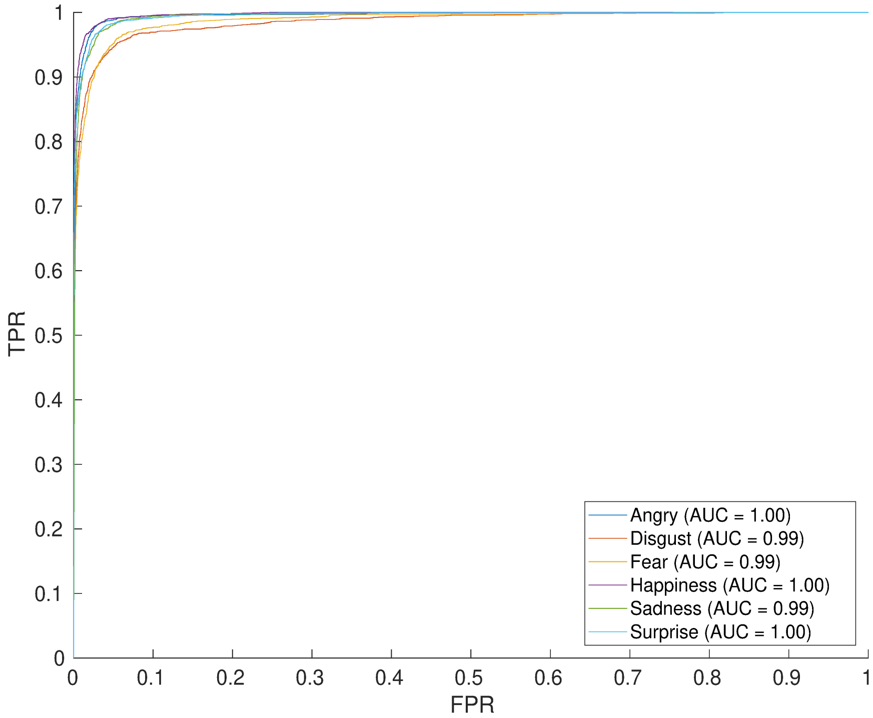


Figure 6.28: Person-dependent ROC graph for SAVEE k = 43.

Appendix G: ROC Graphs (RML, Person-Dependent)

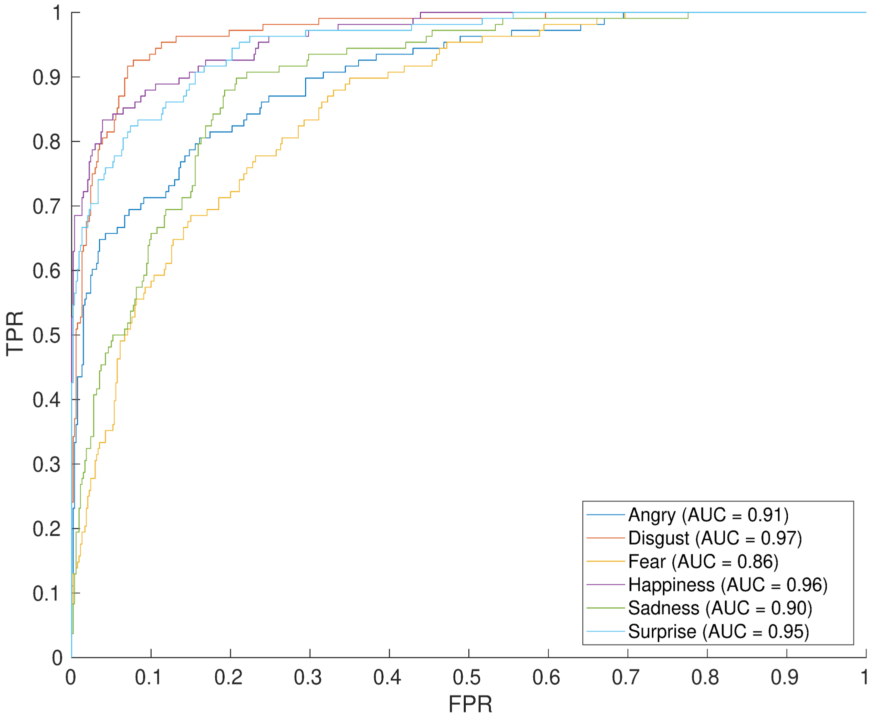


Figure 6.29: Person-dependent ROC graph for RML with k = 1.

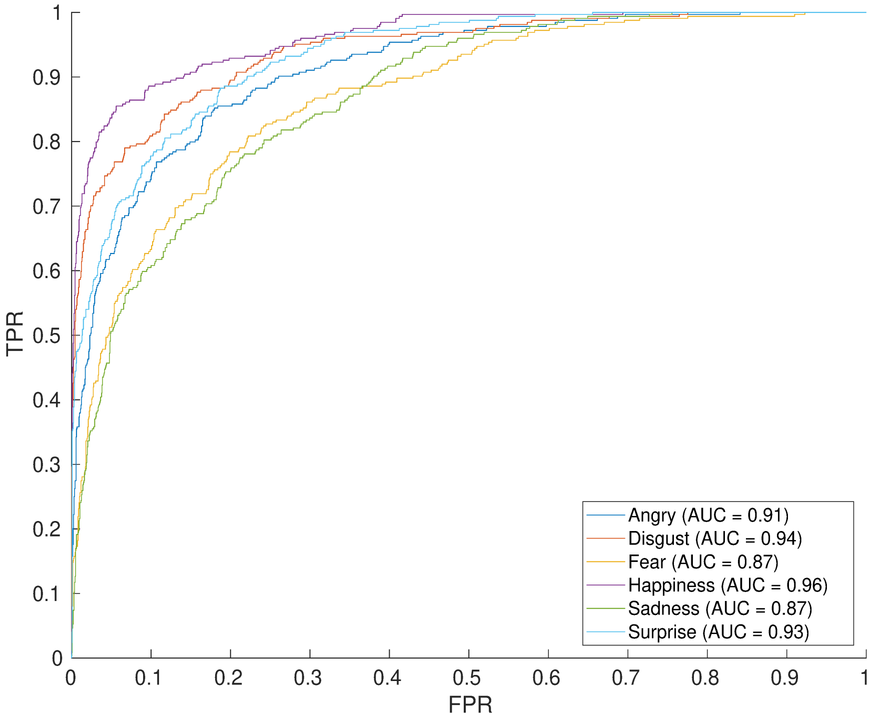


Figure 6.30: Person-dependent ROC graph for RML with k = 3.

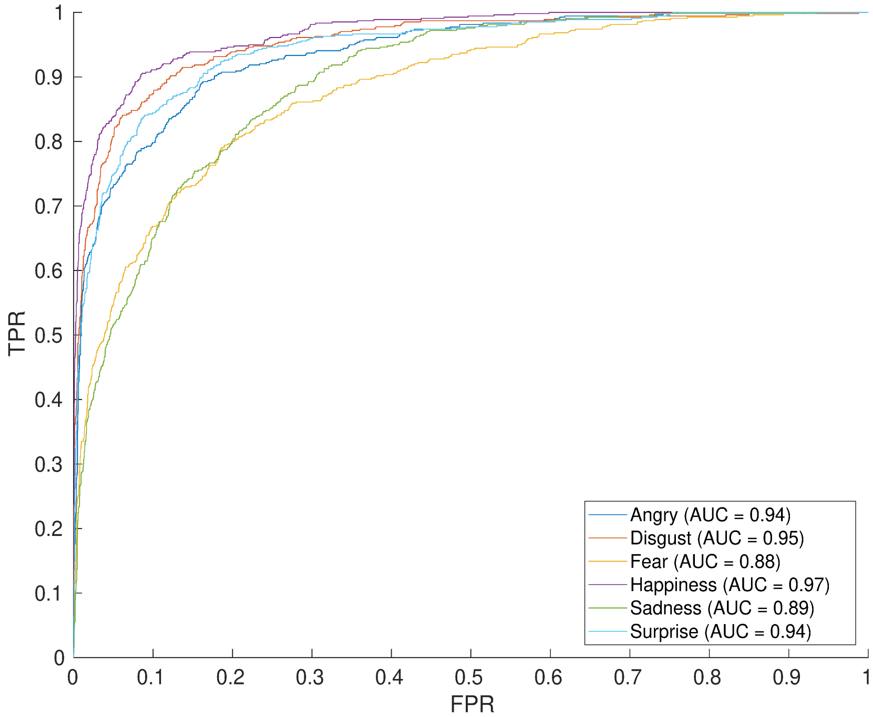


Figure 6.31: Person-dependent ROC graph for RML with k = 5.

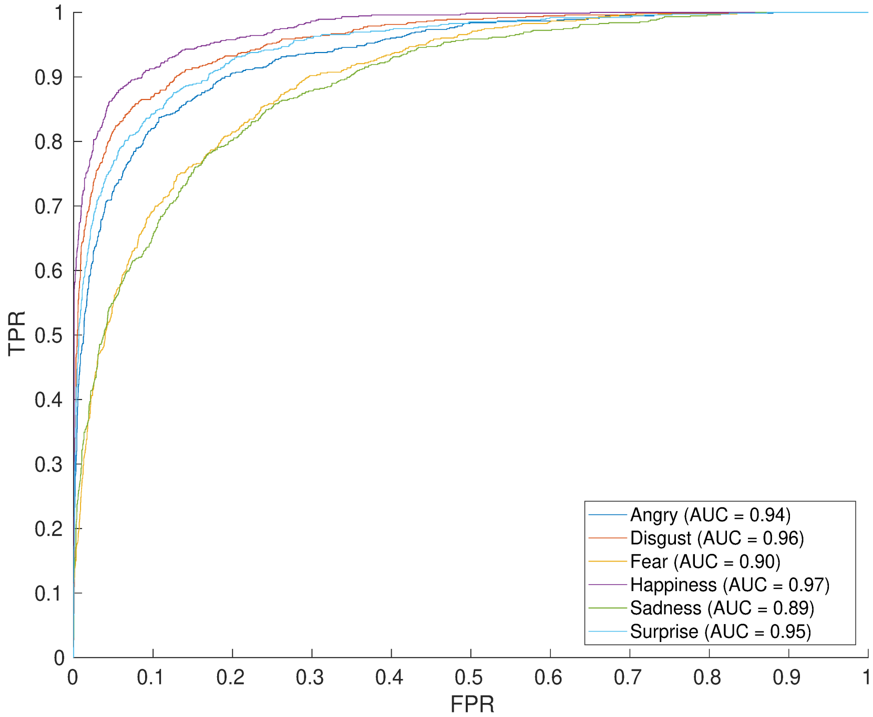


Figure 6.32: Person-dependent ROC graph for RML with k = 7.

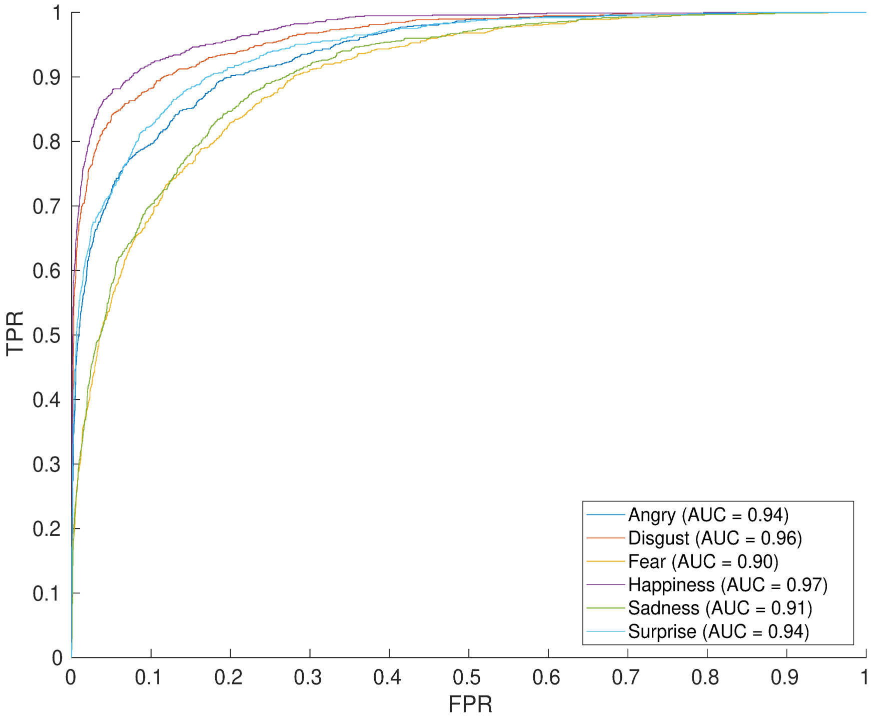


Figure 6.33: Person-dependent ROC graph for RML with k = 9.

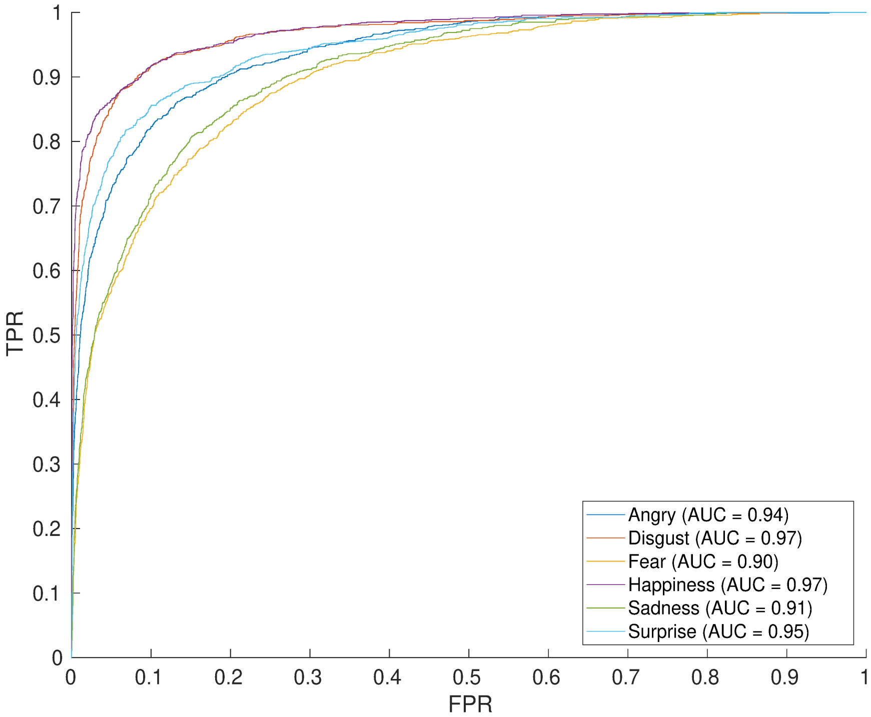


Figure 6.34: Person-dependent ROC graph for RML with k = 11.

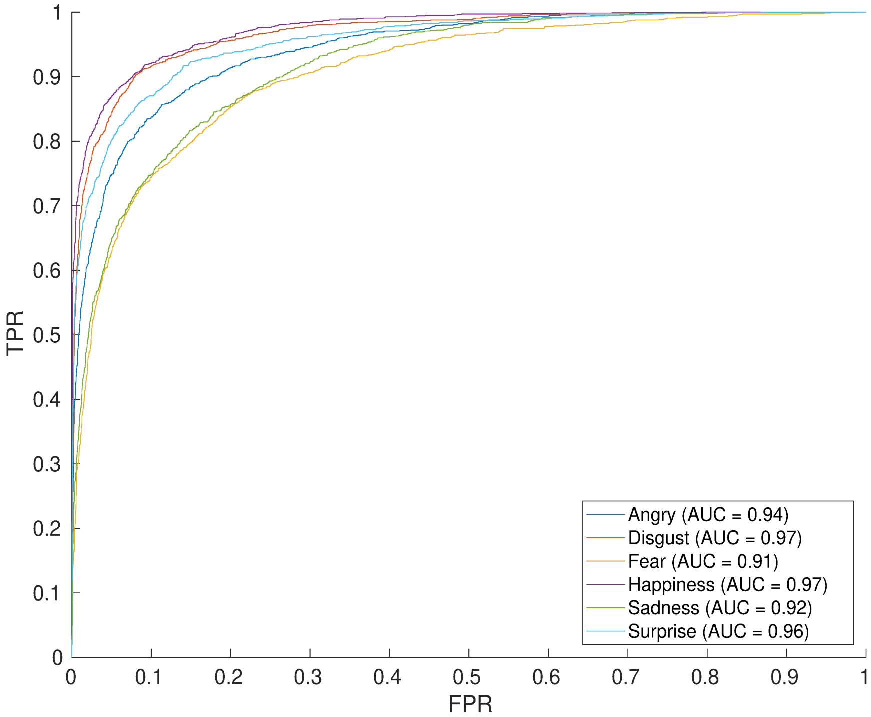


Figure 6.35: Person-dependent ROC graph for RML with k = 13.

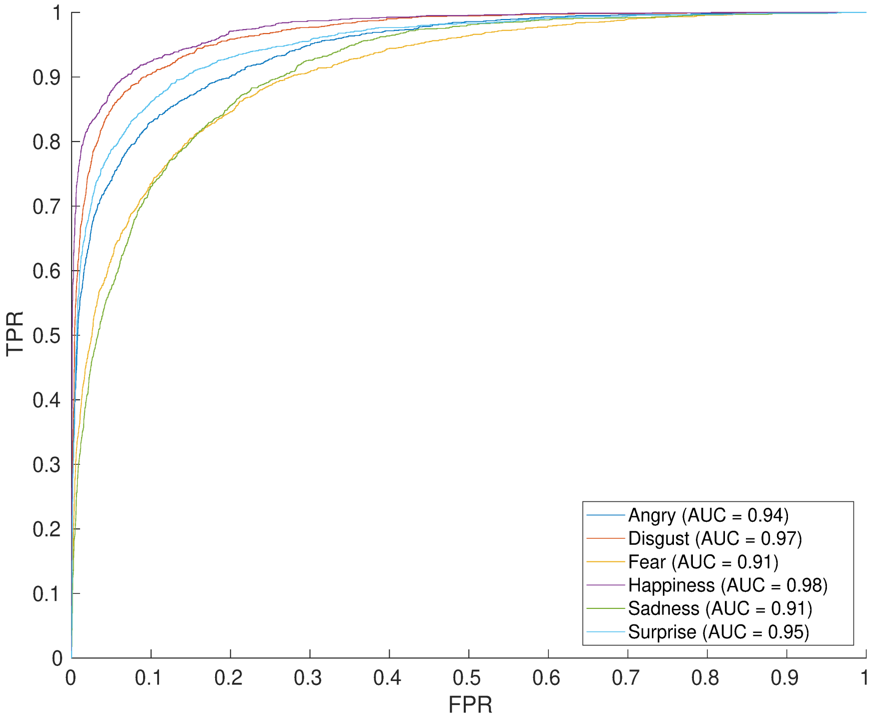


Figure 6.36: Person-dependent ROC graph for RML with k = 15.

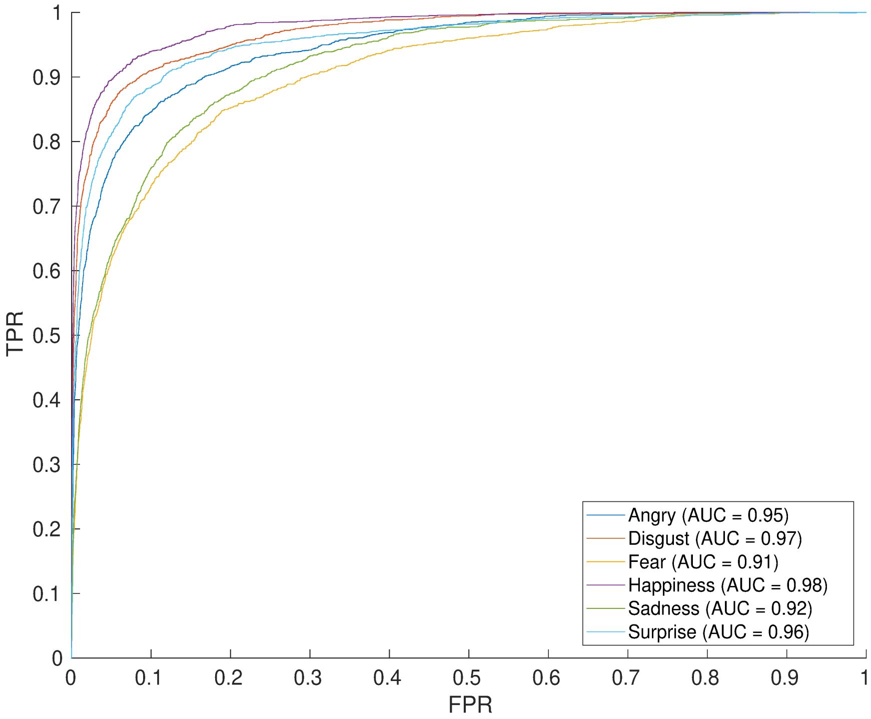


Figure 6.37: Person-dependent ROC graph for RML with k = 17.

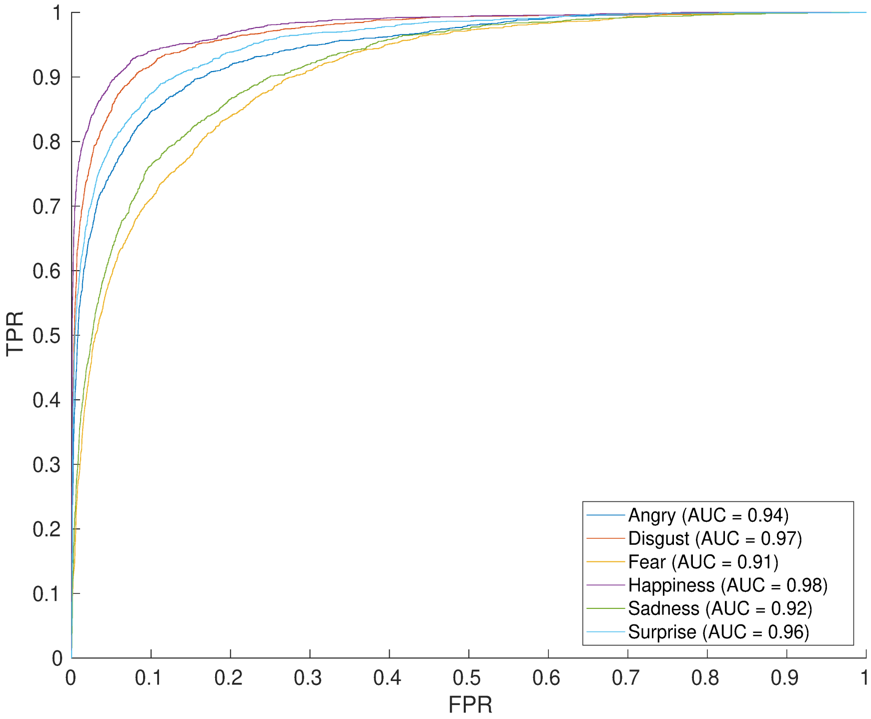


Figure 6.38: Person-dependent ROC graph for RML with k = 19.

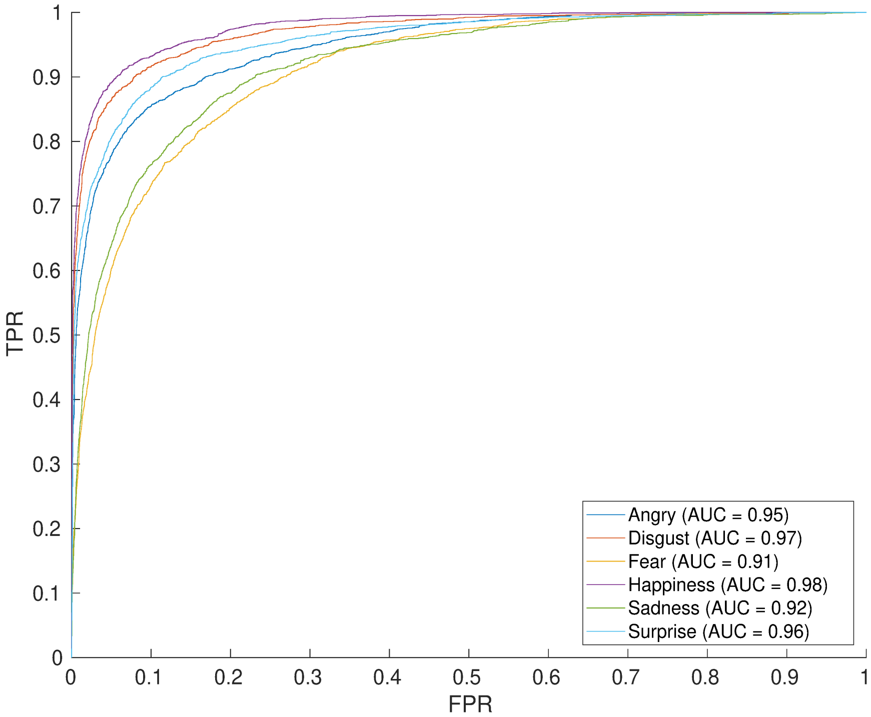


Figure 6.39: Person-dependent ROC graph for RML with k = 21.

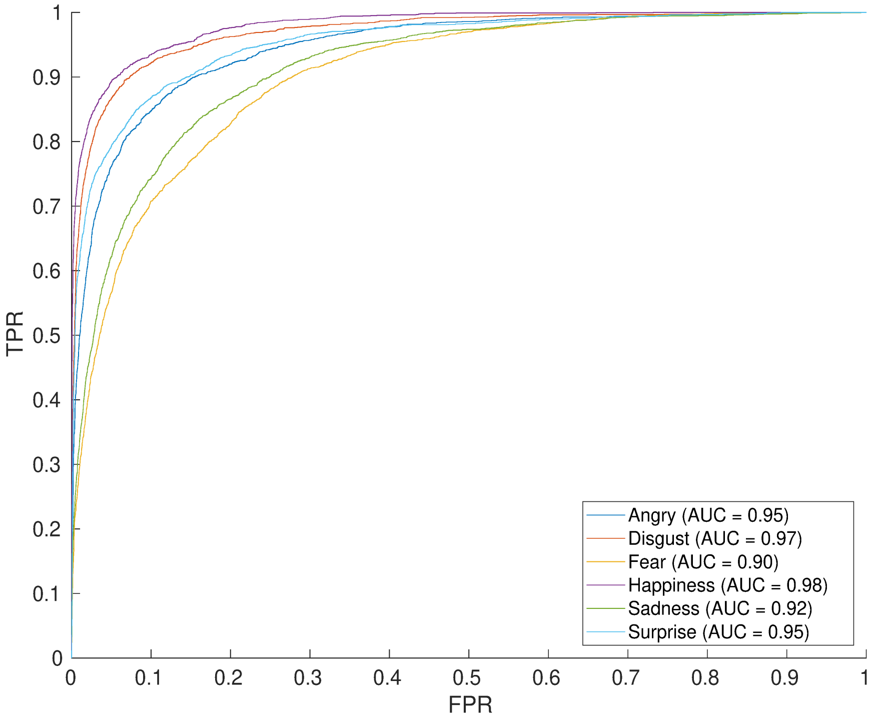


Figure 6.40: Person-dependent ROC graph for RML with k = 23.

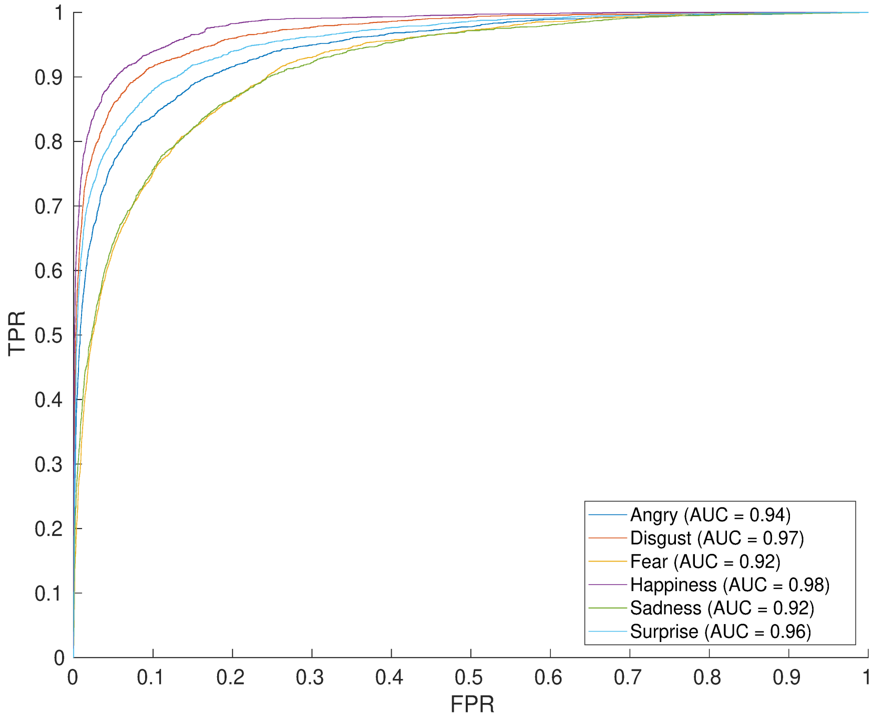


Figure 6.41: Person-dependent ROC graph for RML with k = 25.

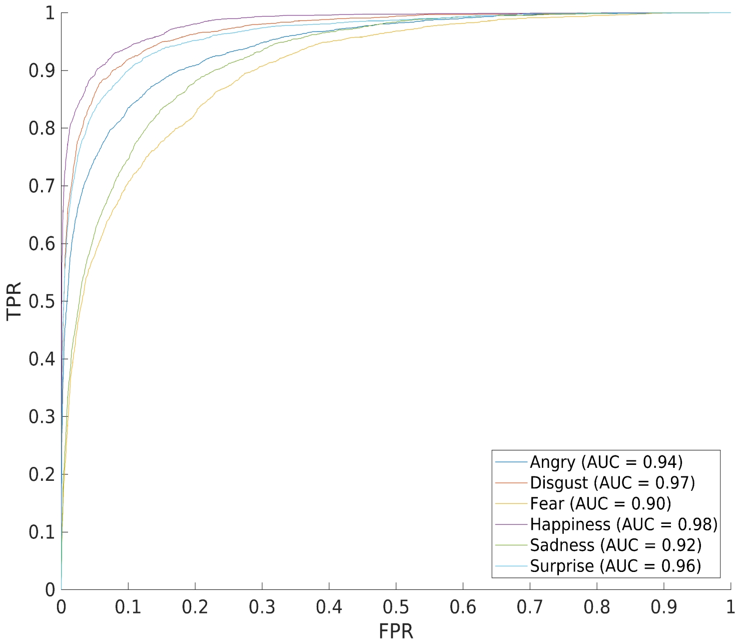


Figure 6.42: Person-dependent ROC graph for RML with k = 27.

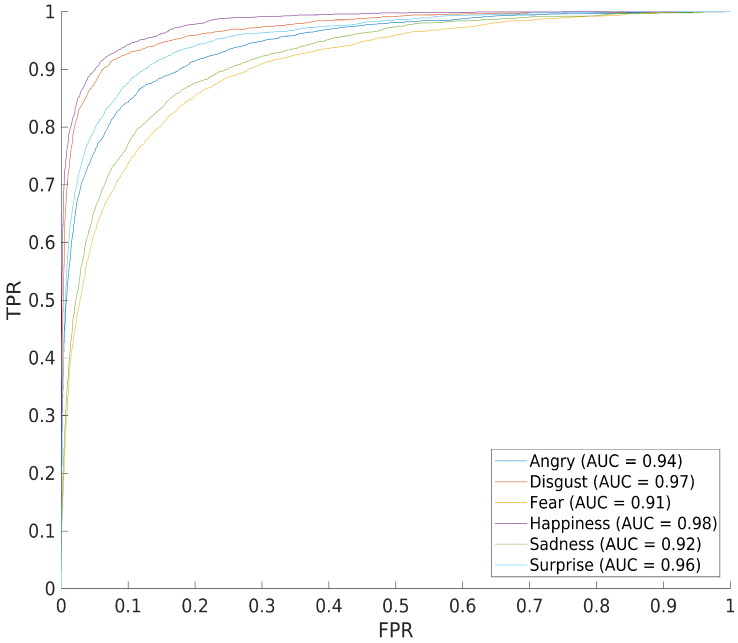


Figure 6.43: Person-dependent ROC graph for RML with k = 29.

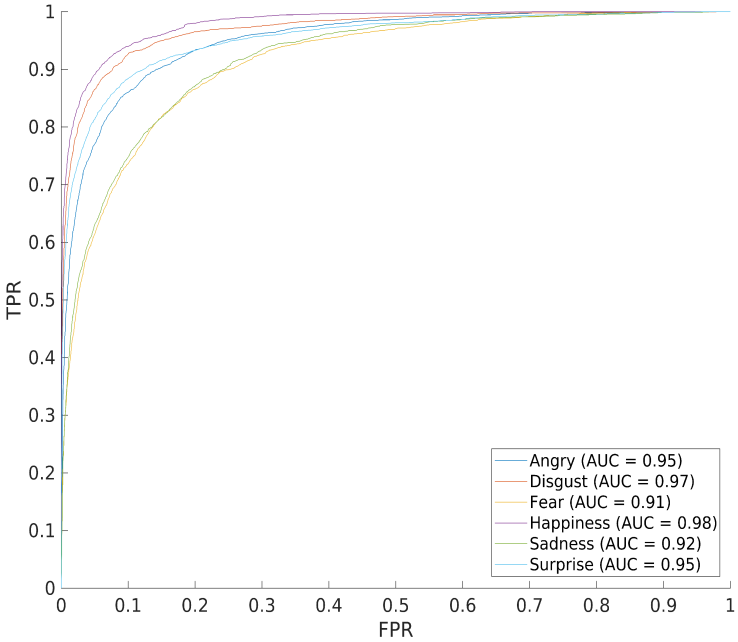


Figure 6.44: Person-dependent ROC graph for RML with k = 31.

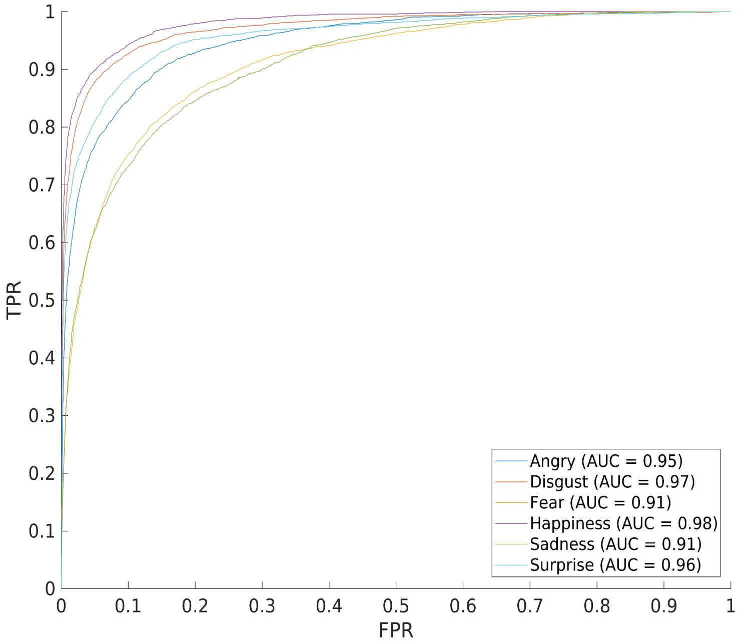


Figure 6.45: Person-dependent ROC graph for RML with k = 33.

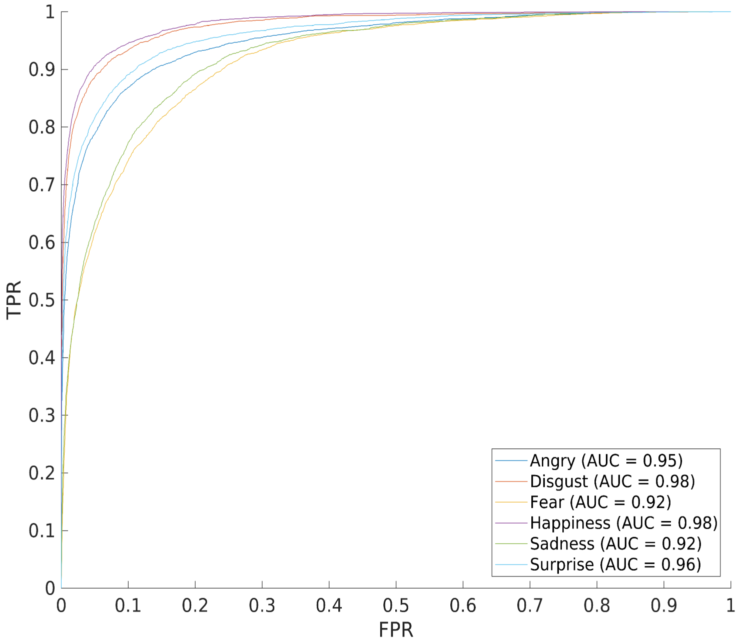


Figure 6.46: Person-dependent ROC graph for RML with k = 35.

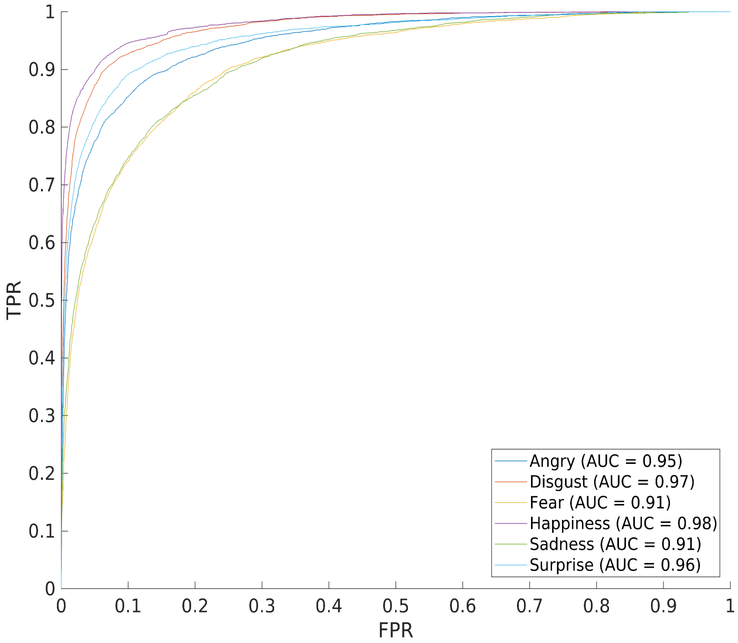


Figure 6.47: Person-dependent ROC graph for RML with k = 37.

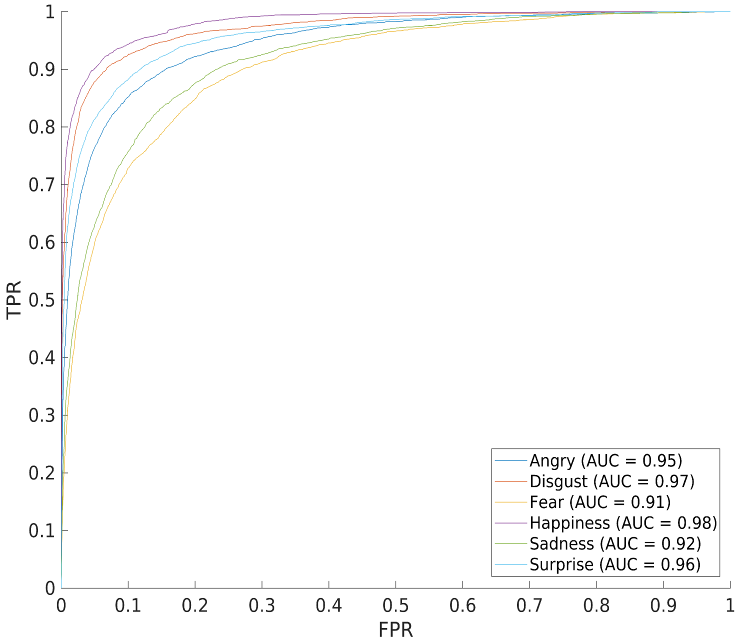


Figure 6.48: Person-dependent ROC graph for RML with k = 39.

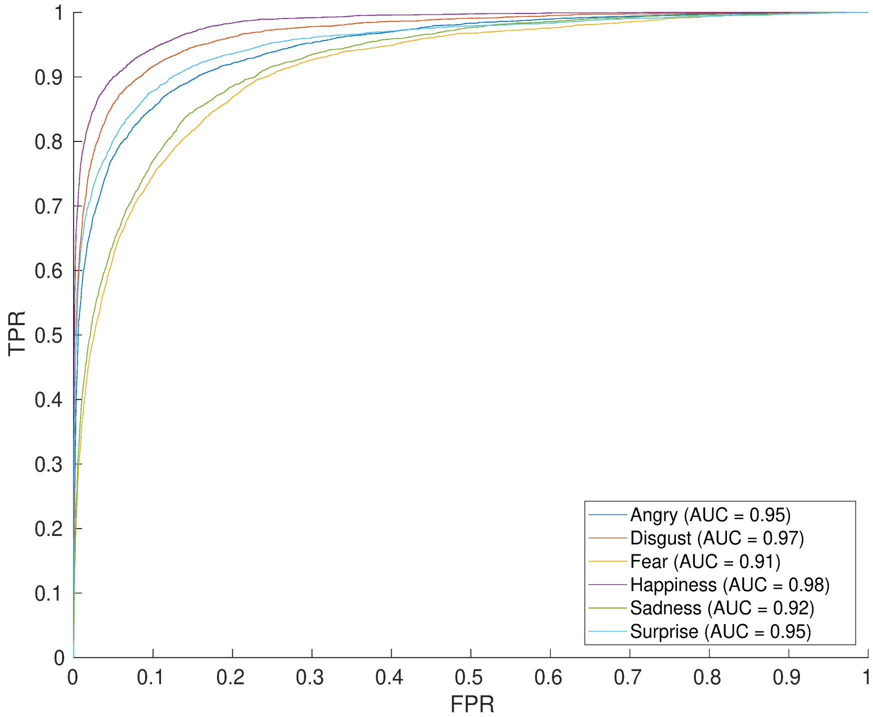


Figure 6.49: Person-dependent ROC graph for RML with k = 41.

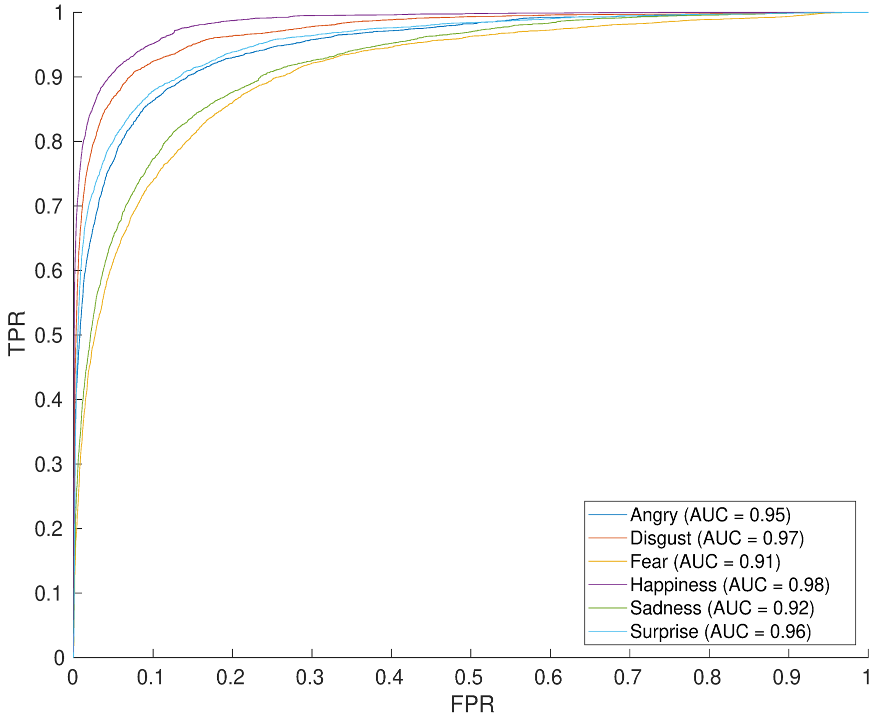


Figure 6.50: Person-dependent ROC graph for RML with k = 43.

Appendix H: ROC Graphs (SAVEE, Person-Independent)

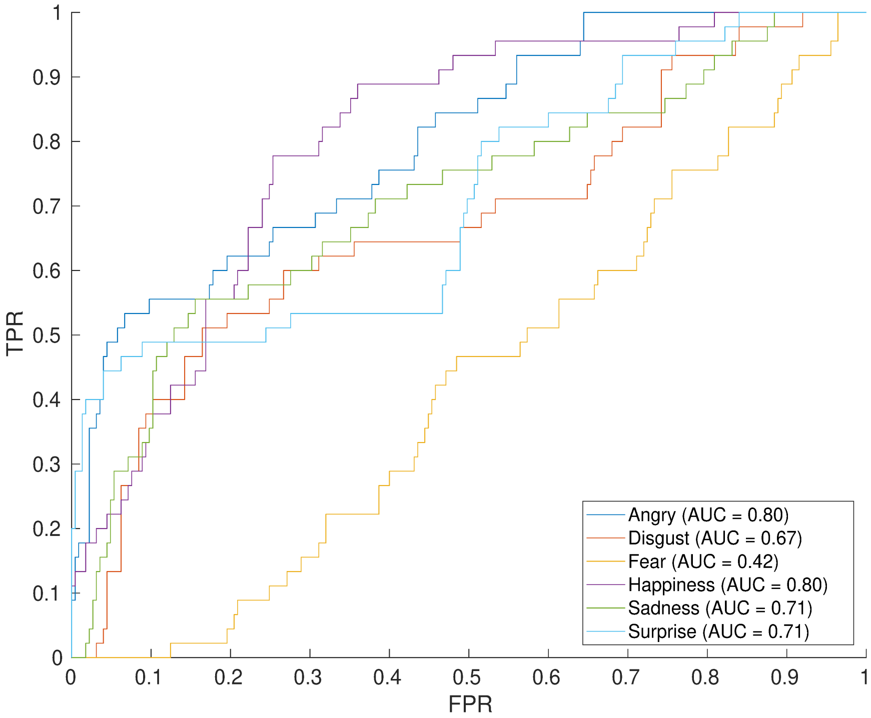


Figure 6.51: Person-independent ROC graph for SAVEE with k = 1.

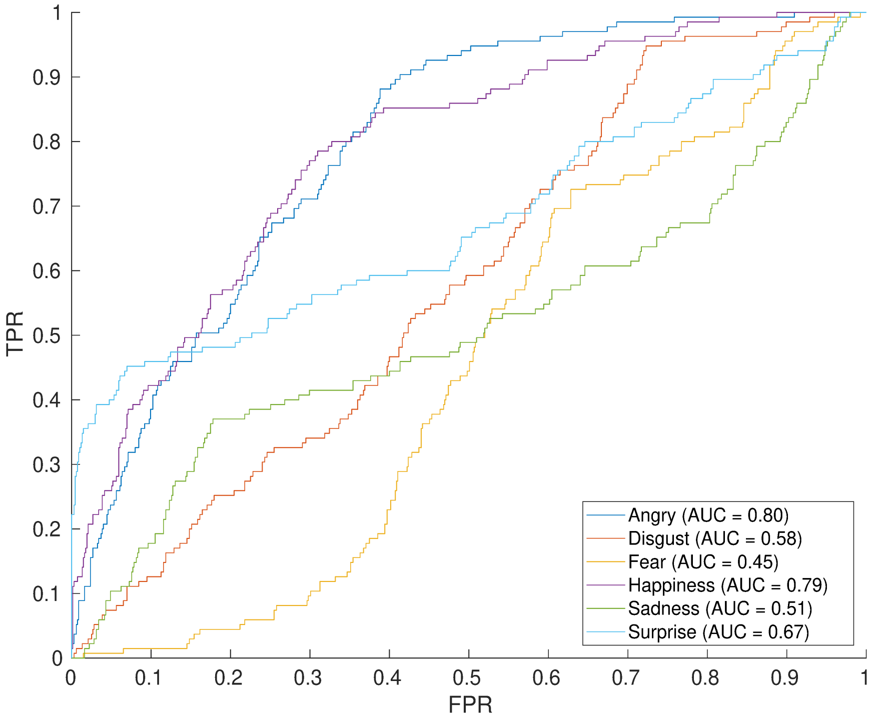


Figure 6.52: Person-independent ROC graph for SAVEE with k = 3.

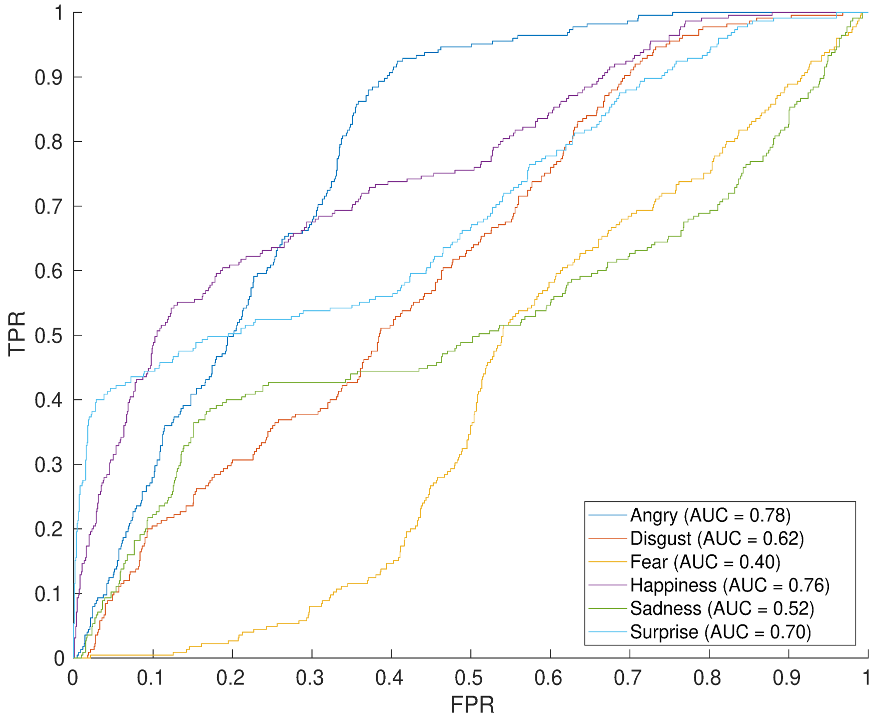


Figure 6.53: Person-independent ROC graph for SAVEE with k = 5.

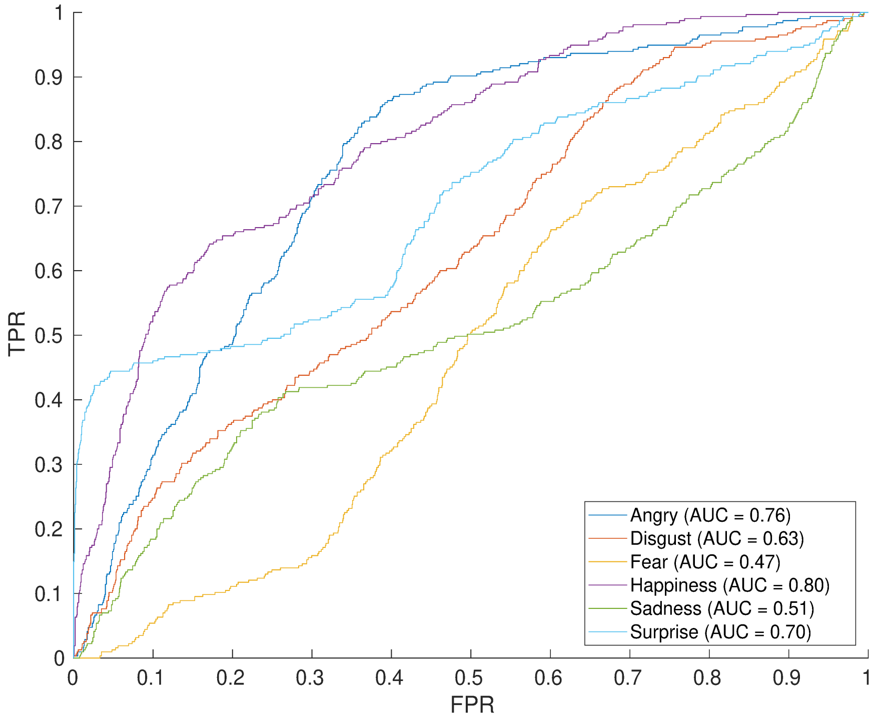


Figure 6.54: Person-independent ROC graph for SAVEE with k = 7.

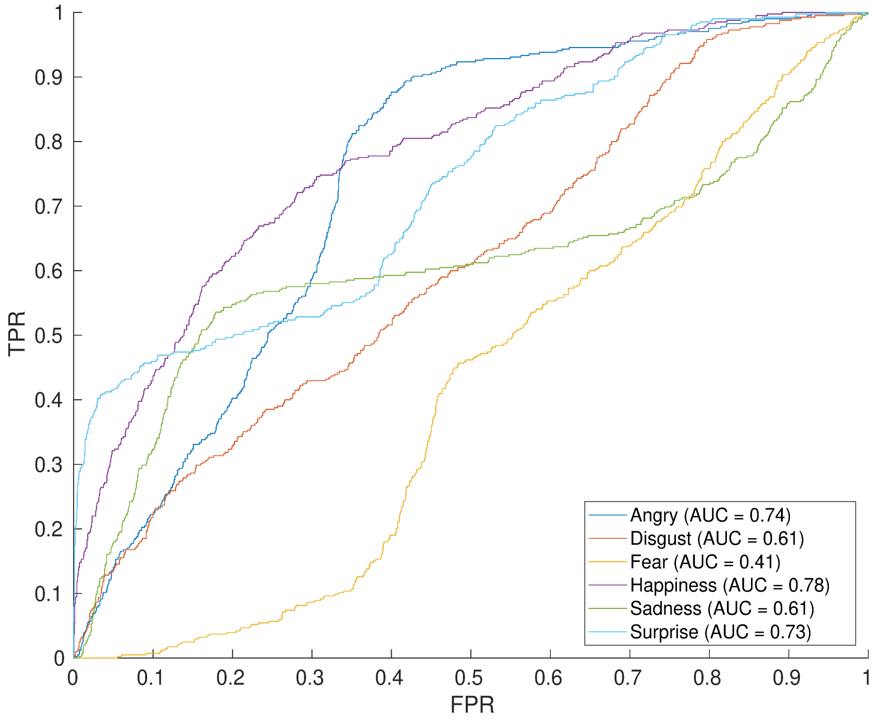


Figure 6.55: Person-independent ROC graph for SAVEE with k = 9.

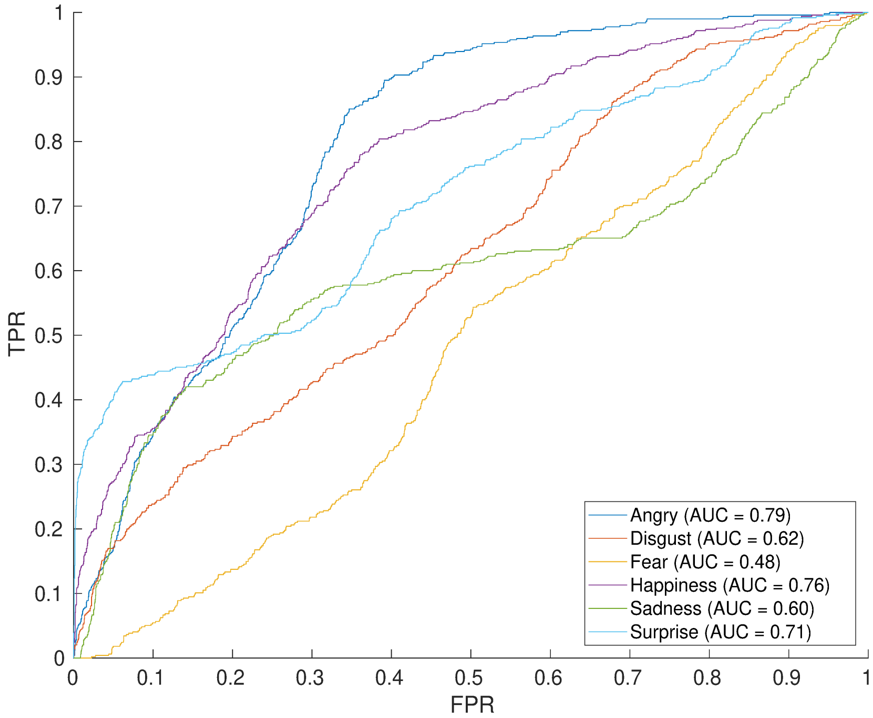


Figure 6.56: Person-independent ROC graph for SAVEE with k = 11.

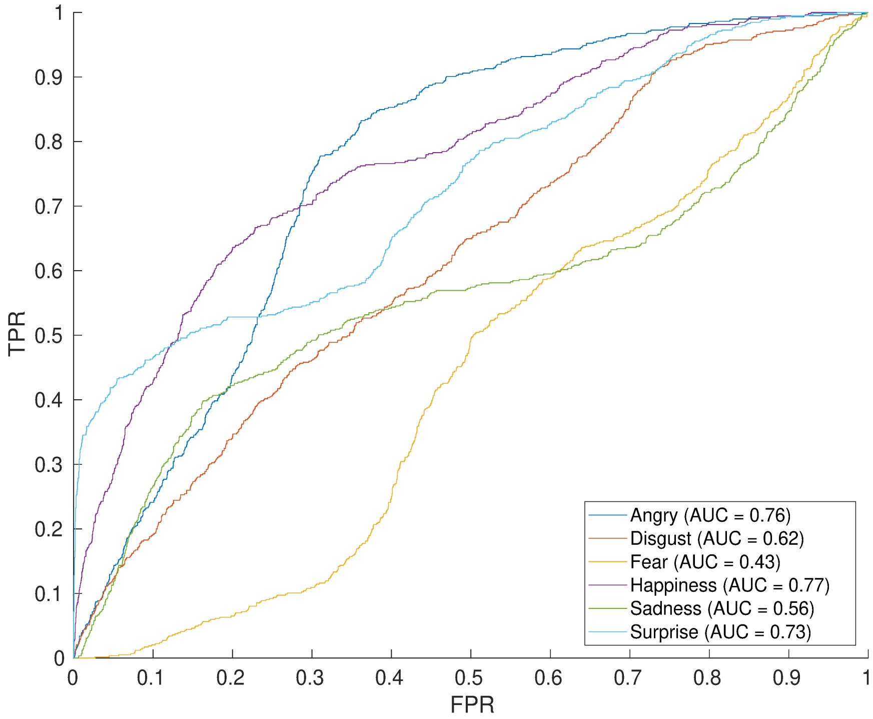


Figure 6.57: Person-independent ROC graph for SAVEE with k = 13.

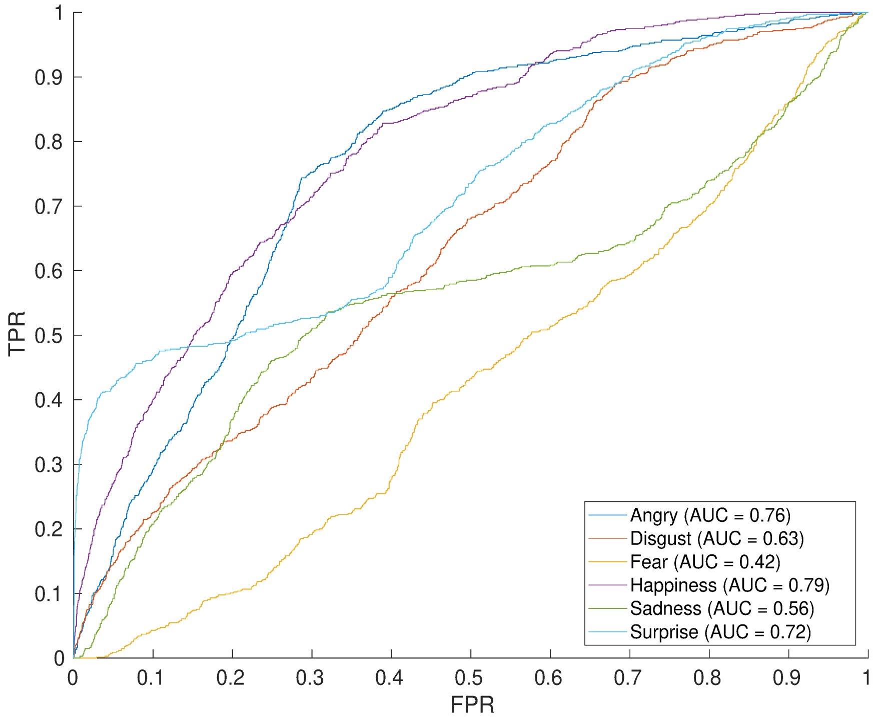


Figure 6.58: Person-independent ROC graph for SAVEE with k = 15.

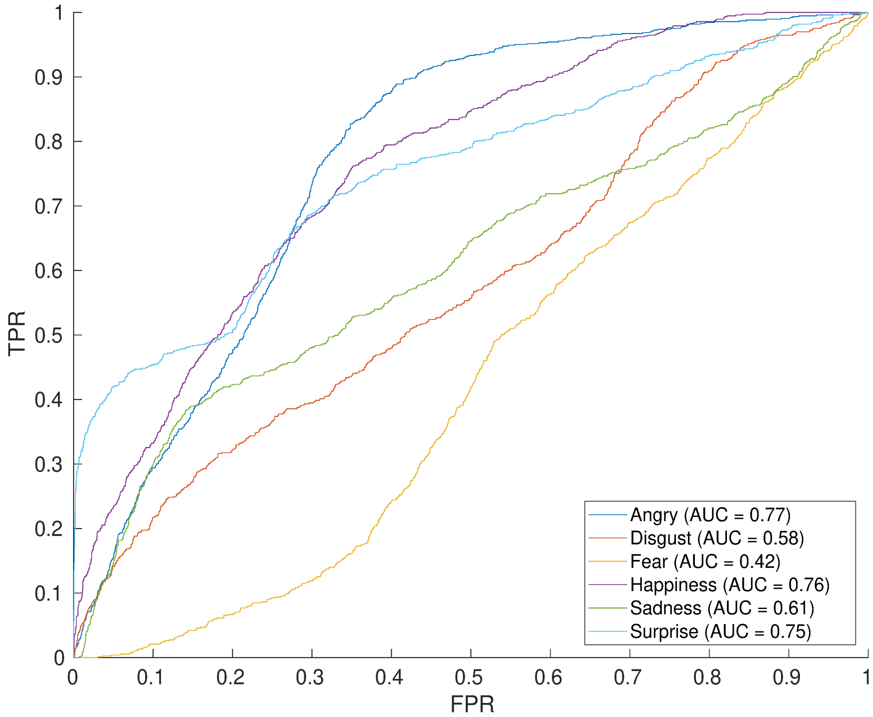


Figure 6.59: Person-independent ROC graph for SAVEE with k = 17.

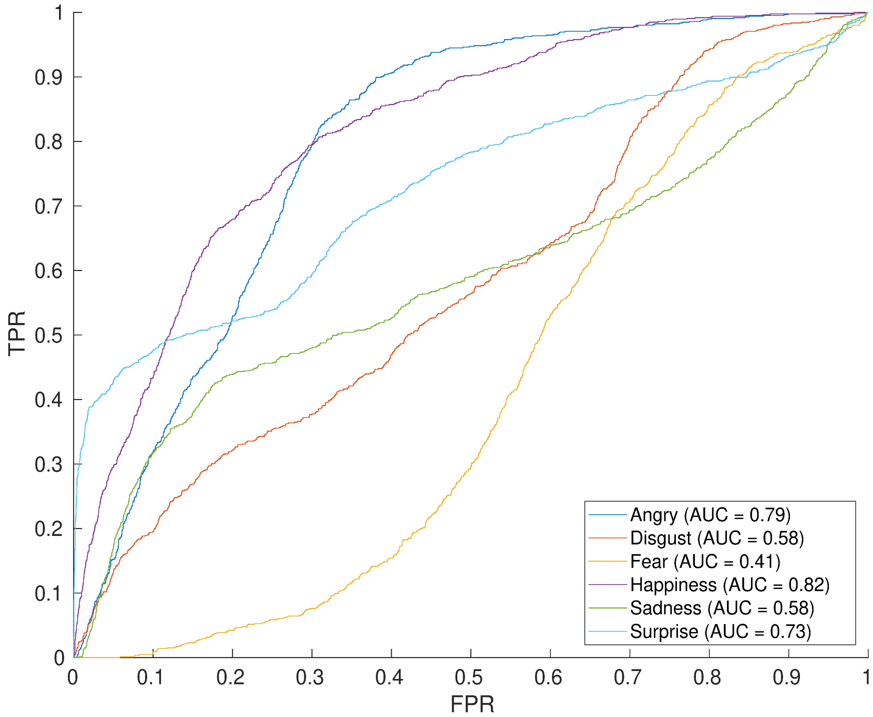


Figure 6.60: Person-independent ROC graph for SAVEE with k = 19.

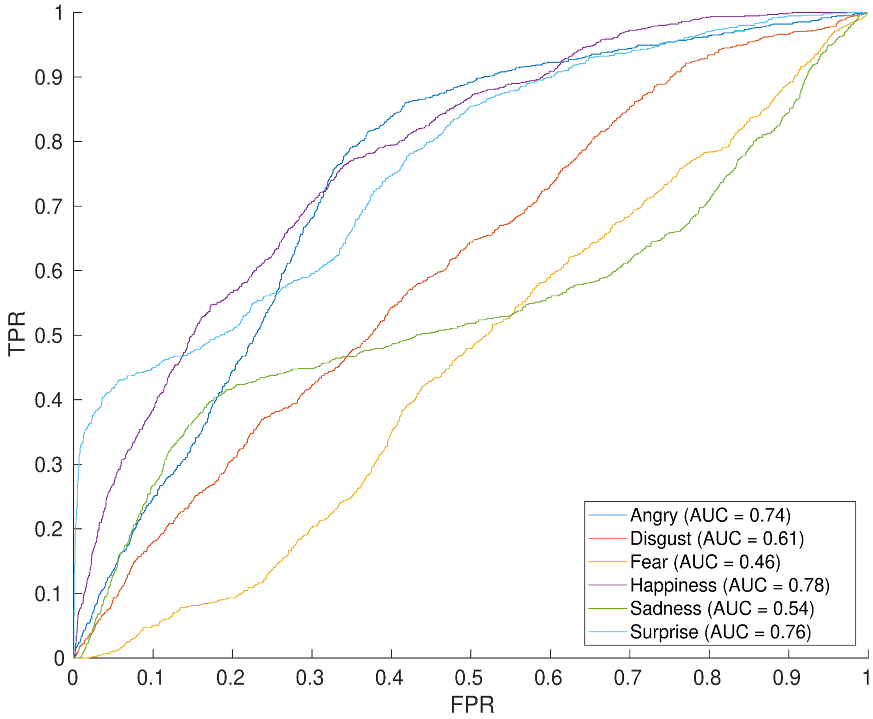


Figure 6.61: Person-independent ROC graph for SAVEE with k = 21.

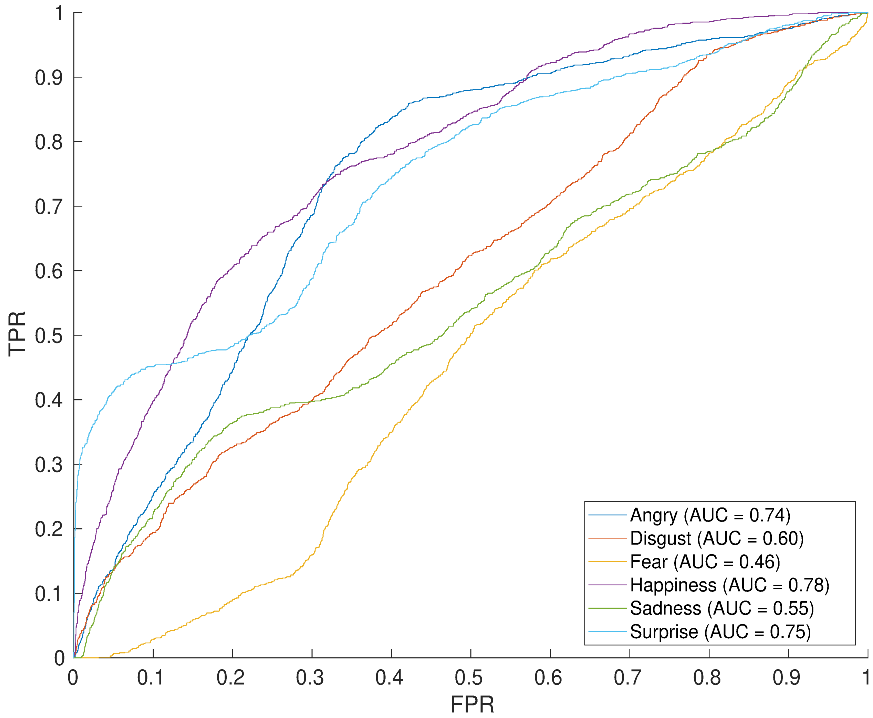


Figure 6.62: Person-independent ROC graph for SAVEE with k = 23.

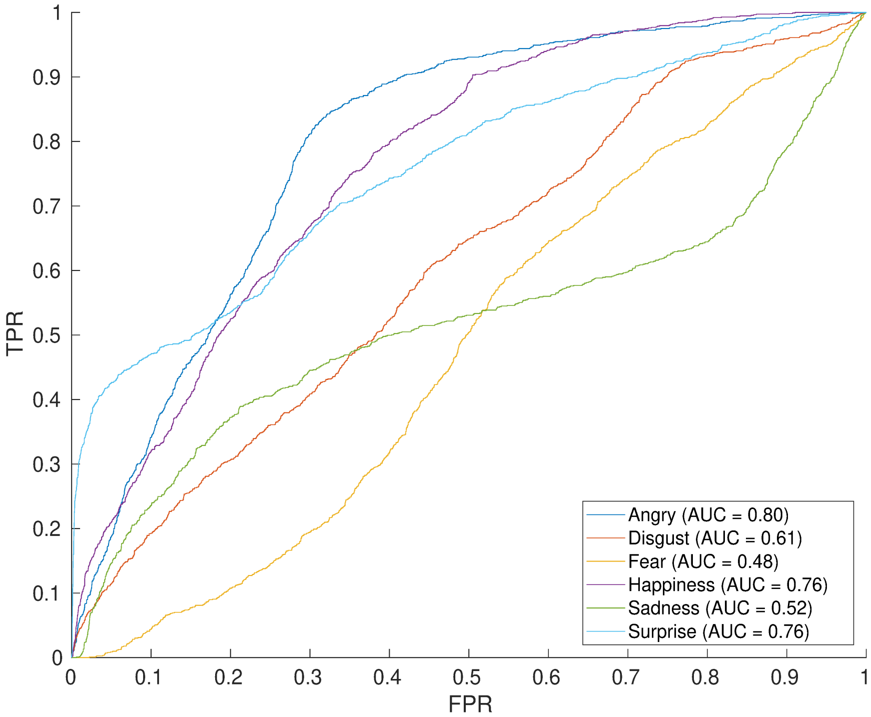


Figure 6.63: Person-independent ROC graph for SAVEE with k = 25.

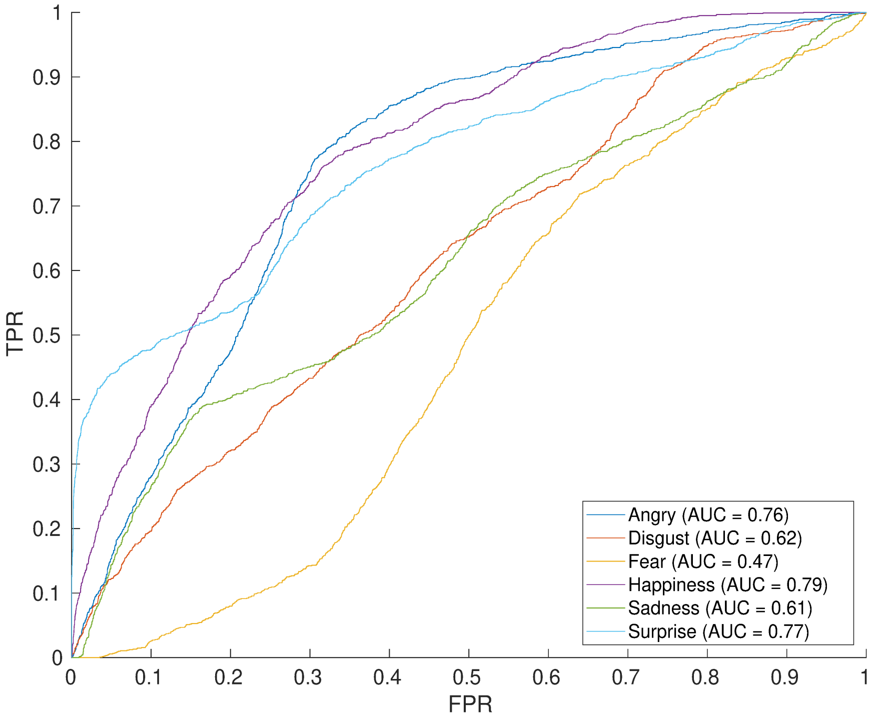


Figure 6.64: Person-independent ROC graph for SAVEE with k = 27.

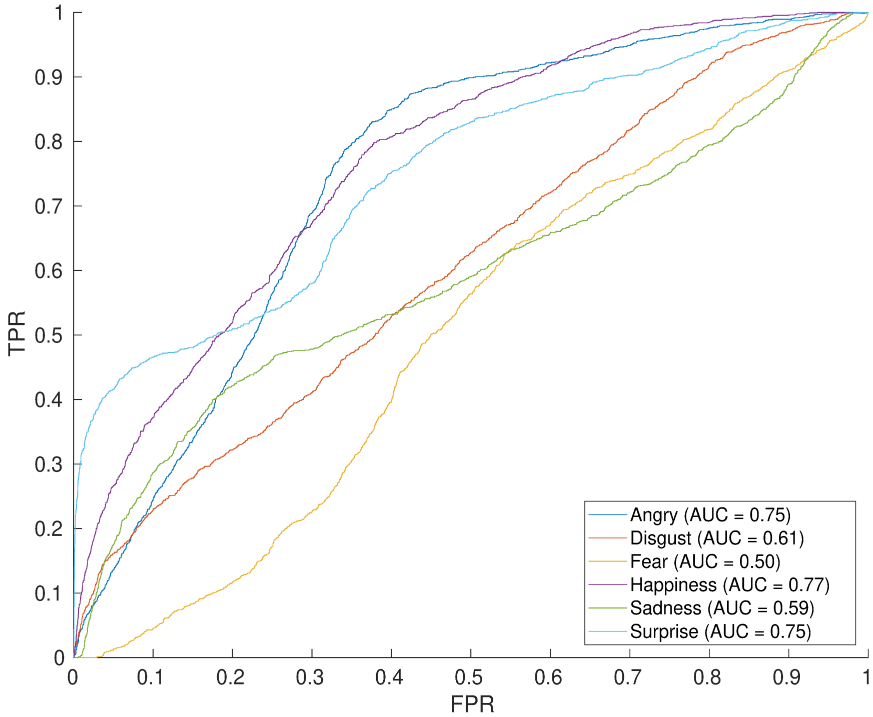


Figure 6.65: Person-independent ROC graph for SAVEE with k = 29.

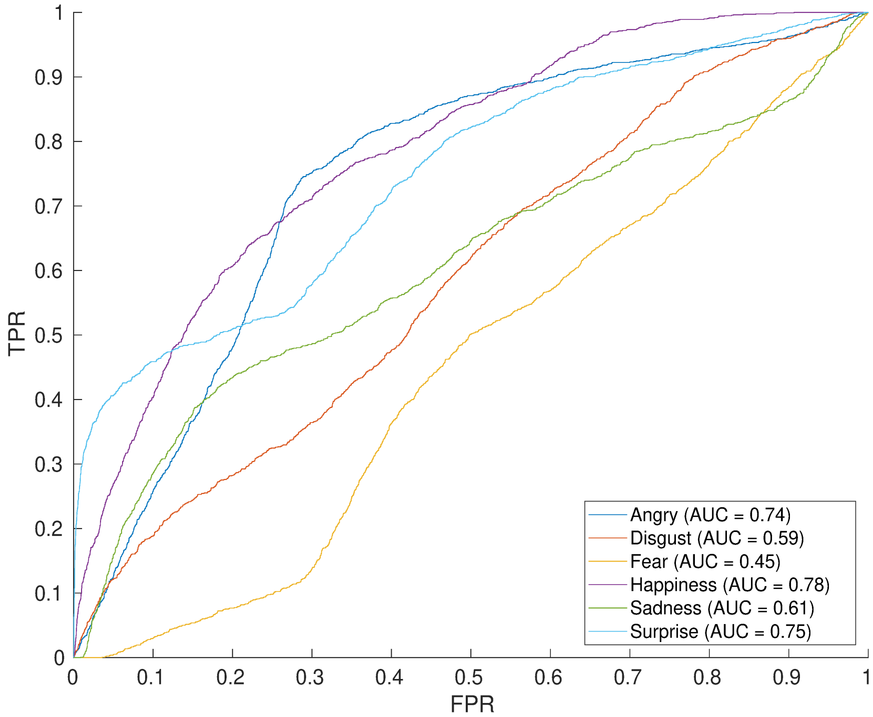


Figure 6.66: Person-independent ROC graph for SAVEE with k = 31.

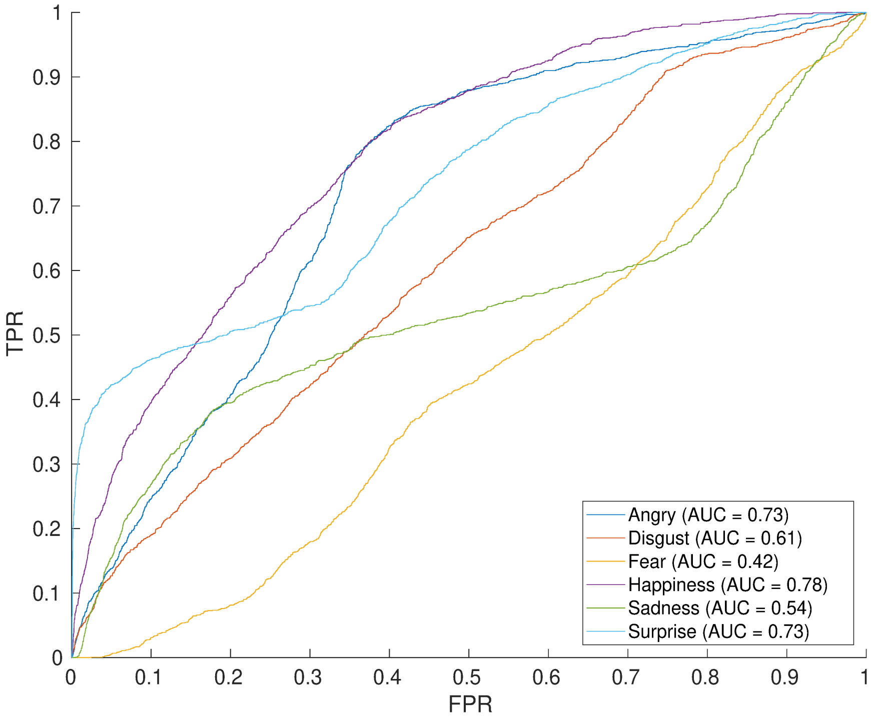


Figure 6.67: Person-independent ROC graph for SAVEE with k = 33.

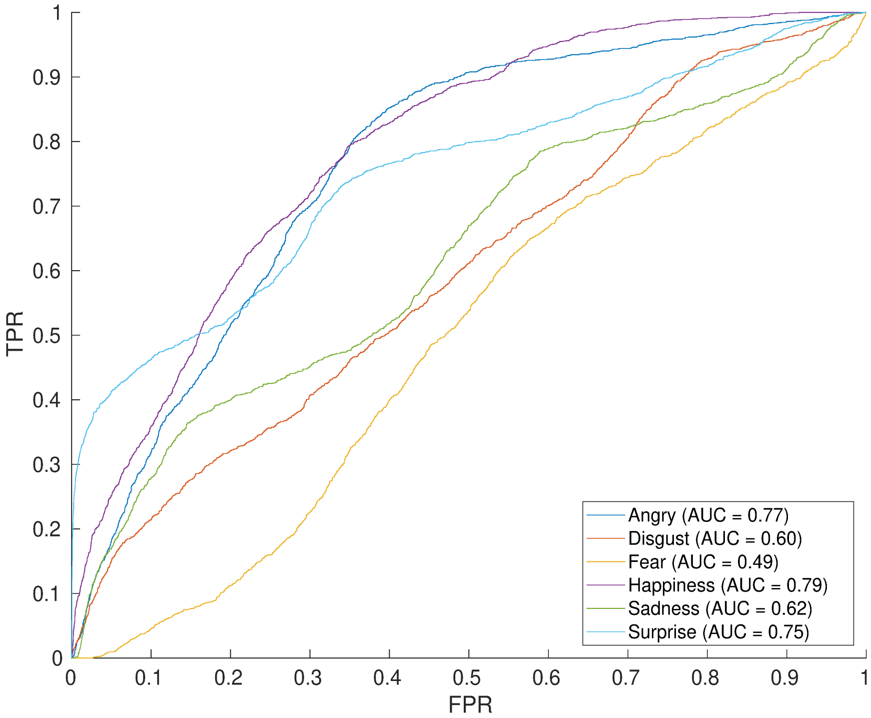


Figure 6.68: Person-independent ROC graph for SAVEE with k = 35.

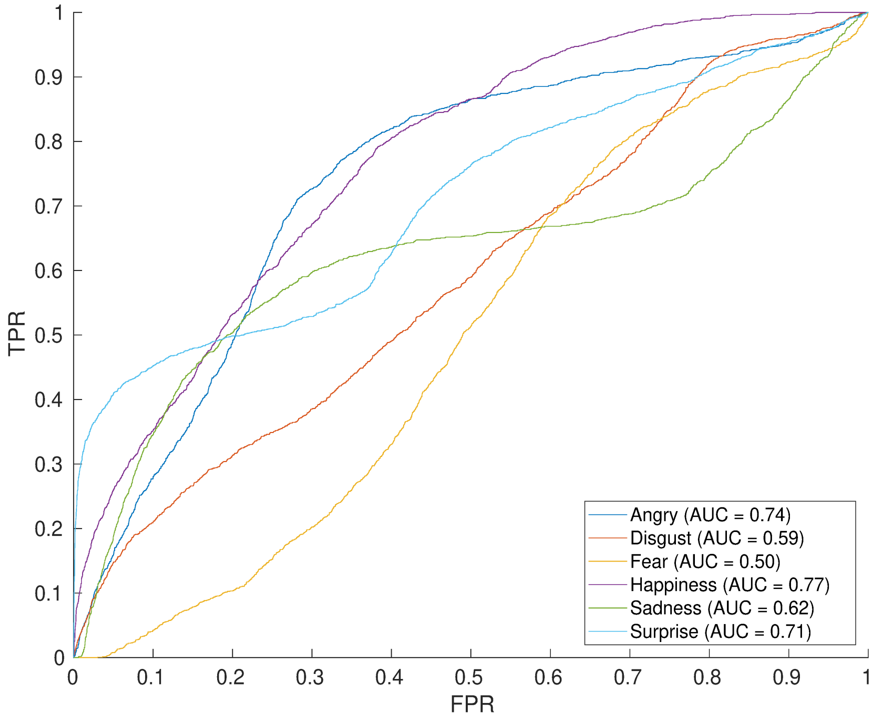


Figure 6.69: Person-independent ROC graph for SAVEE with k = 37.

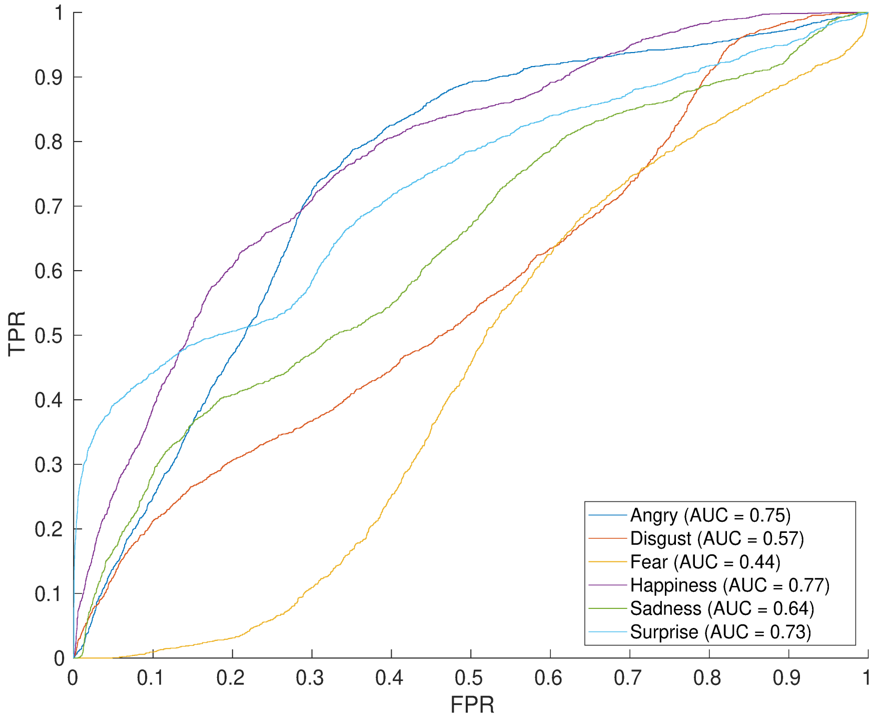


Figure 6.70: Person-independent ROC graph for SAVEE with k = 39.

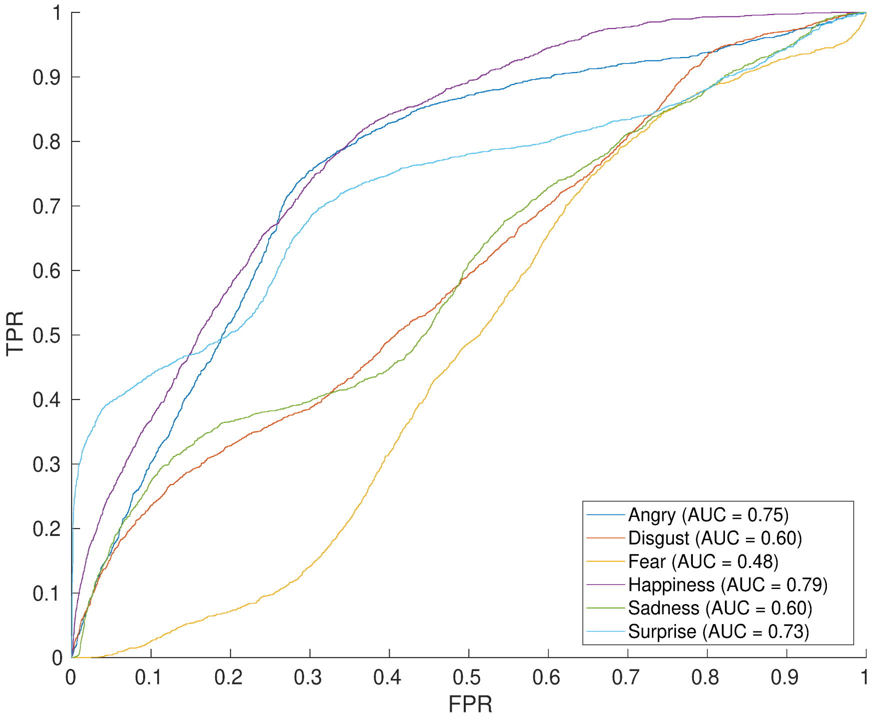


Figure 6.71: Person-independent ROC graph for SAVEE with k = 41.

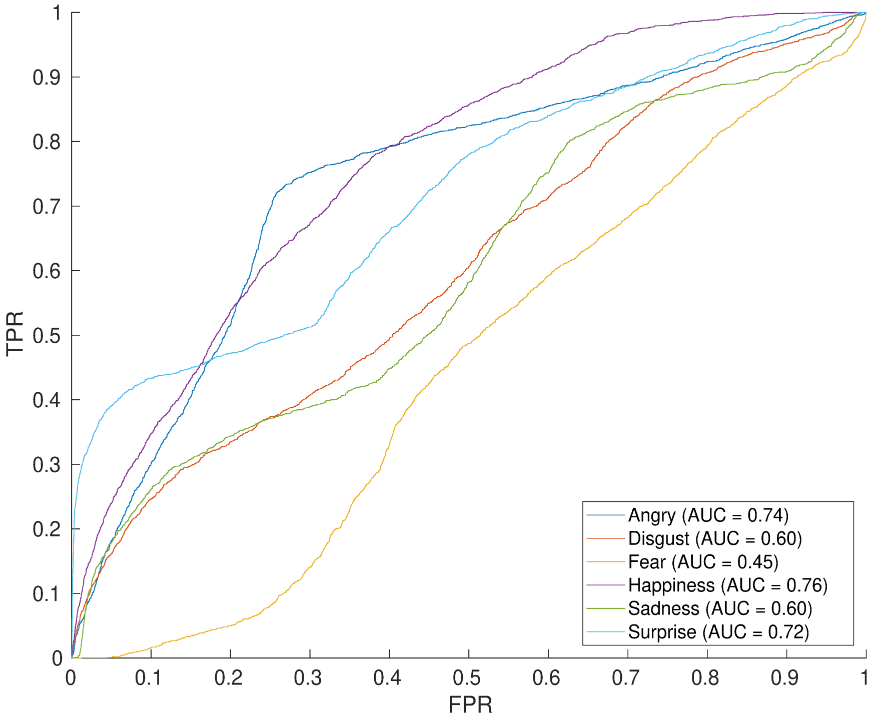


Figure 6.72: Person-independent ROC graph for SAVEE with k = 43.

Appendix I: ROC Graphs (RML, Person-Independent)

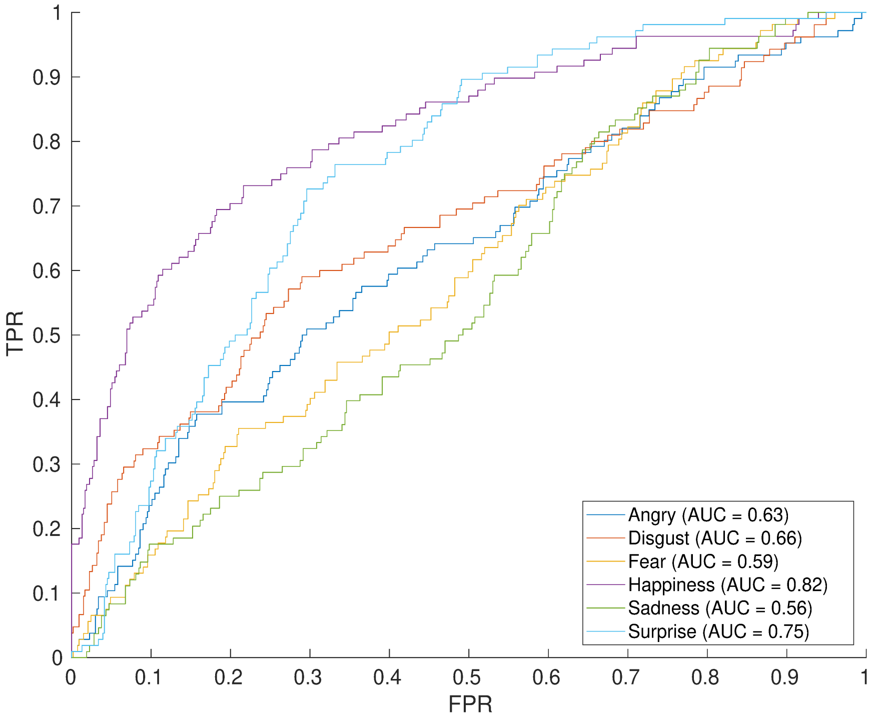


Figure 6.73: Person-independent ROC graph for RML with k = 1.

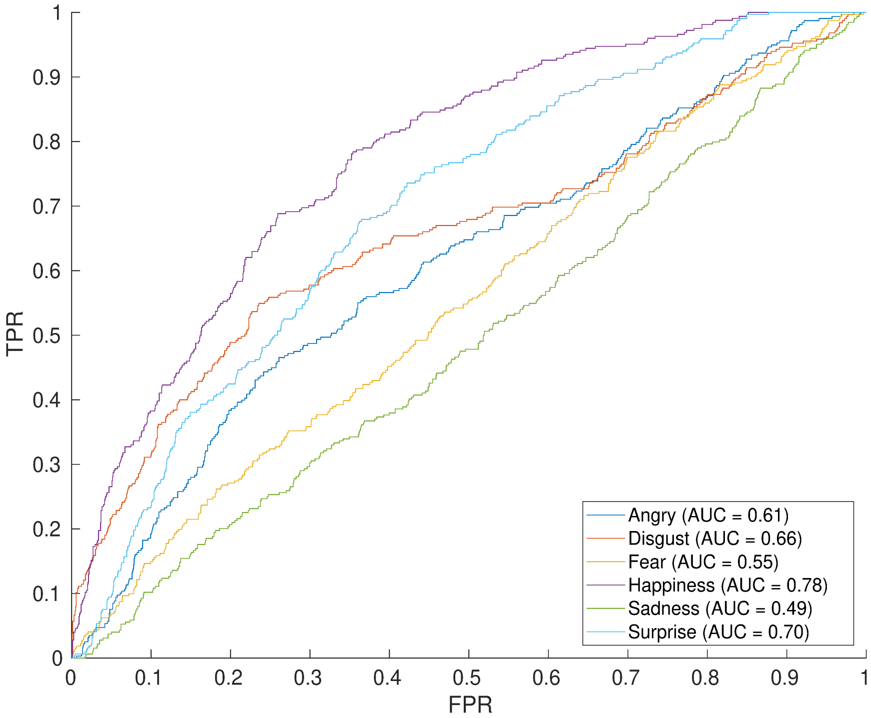


Figure 6.74: Person-independent ROC graph for RML with k = 3.

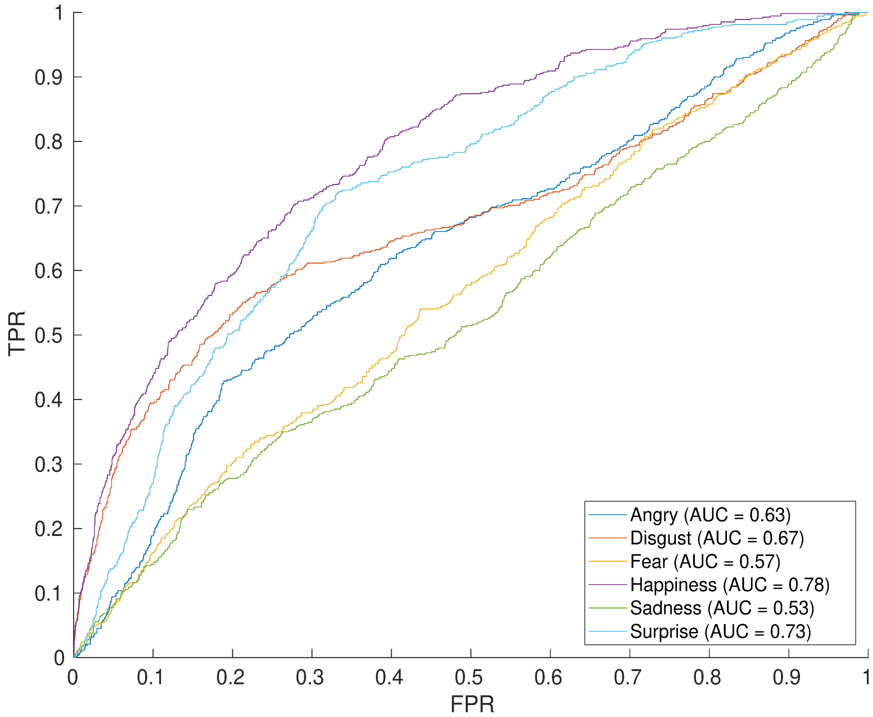


Figure 6.75: Person-independent ROC graph for RML with k = 5.

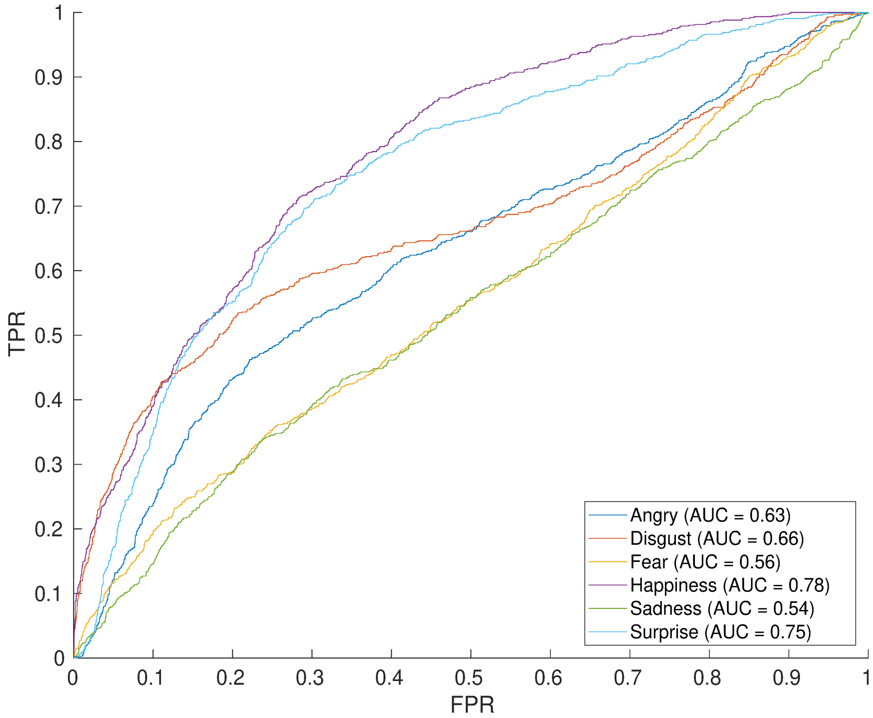


Figure 6.76: Person-independent ROC graph for RML with k = 7.

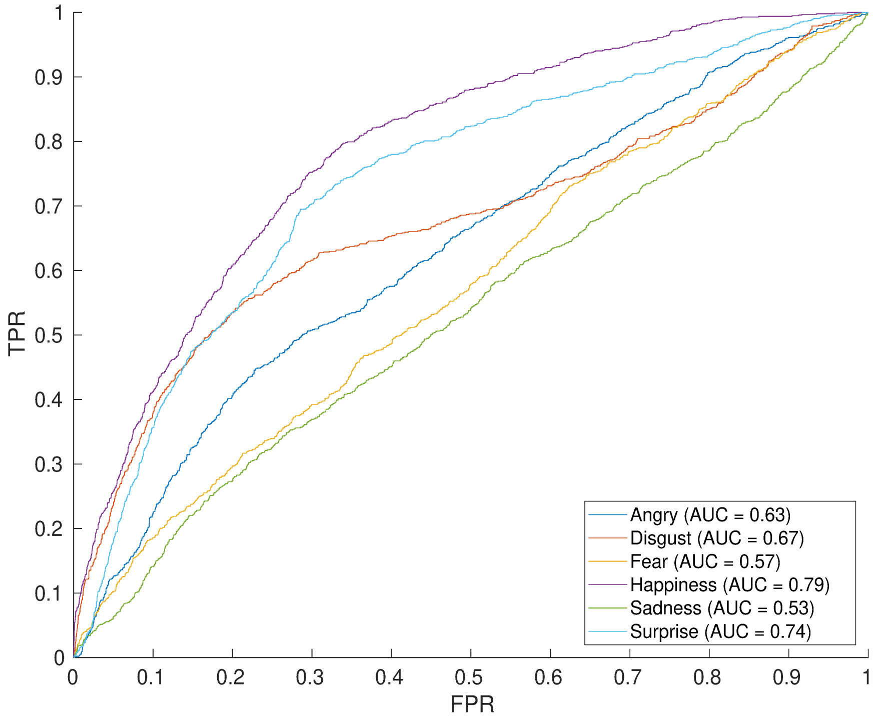


Figure 6.77: Person-independent ROC graph for RML with k = 9.

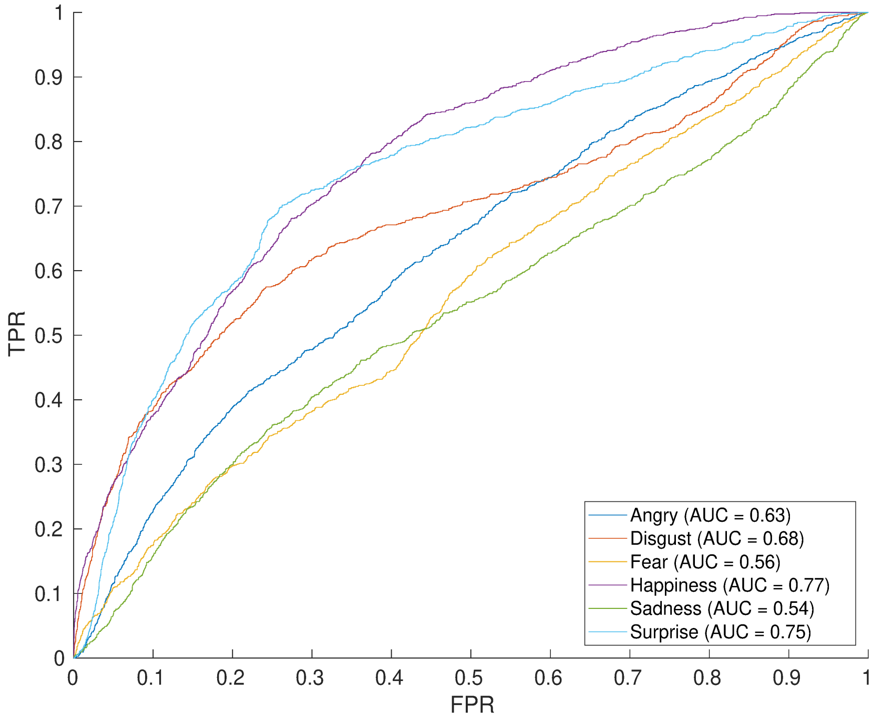


Figure 6.78: Person-independent ROC graph for RML with k = 11.

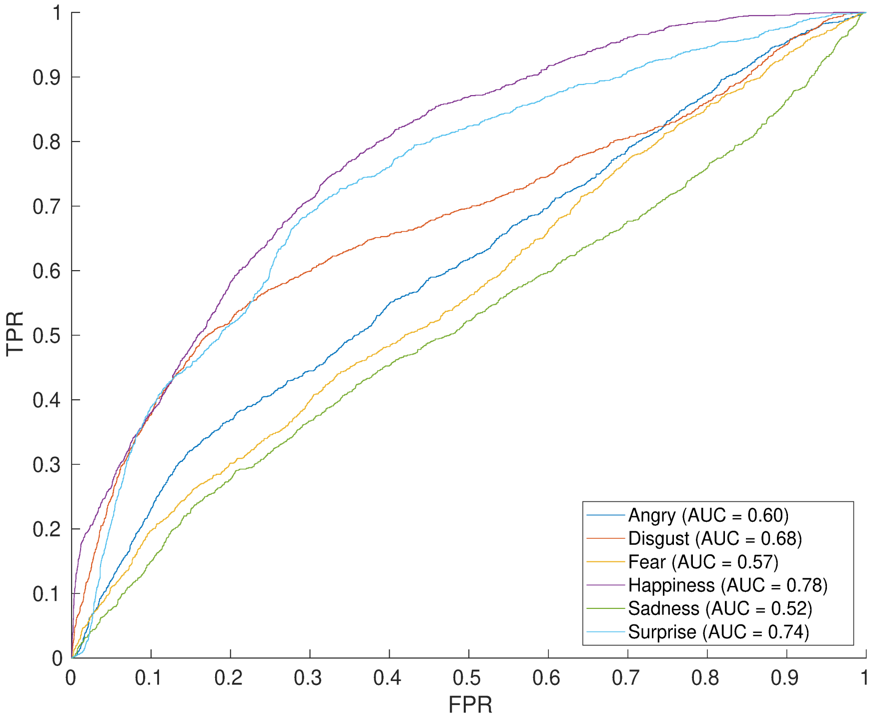


Figure 6.79: Person-independent ROC graph for RML with k = 13.

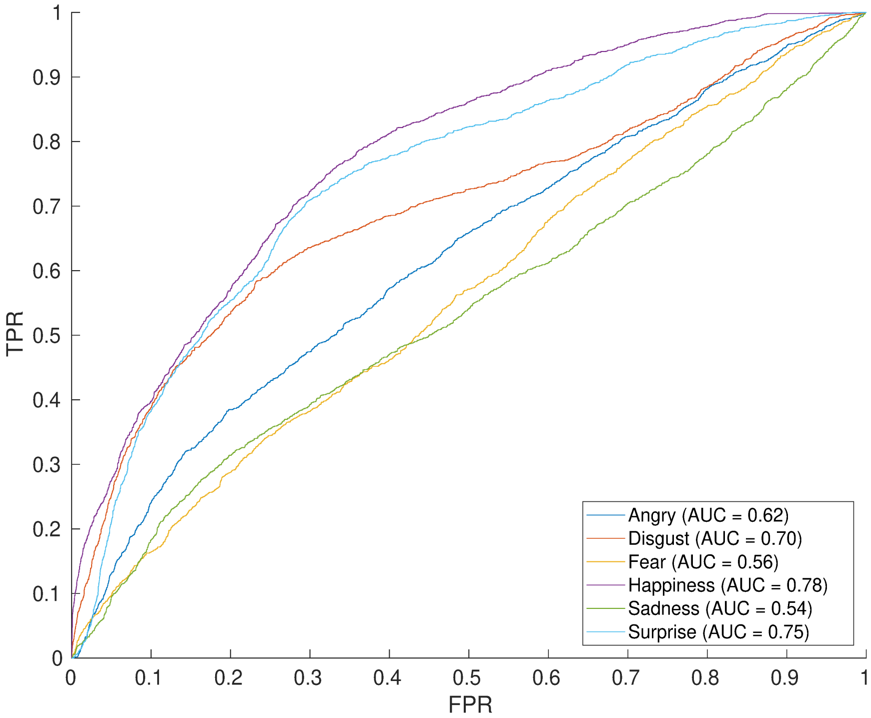


Figure 6.80: Person-independent ROC graph for RML with k = 15.

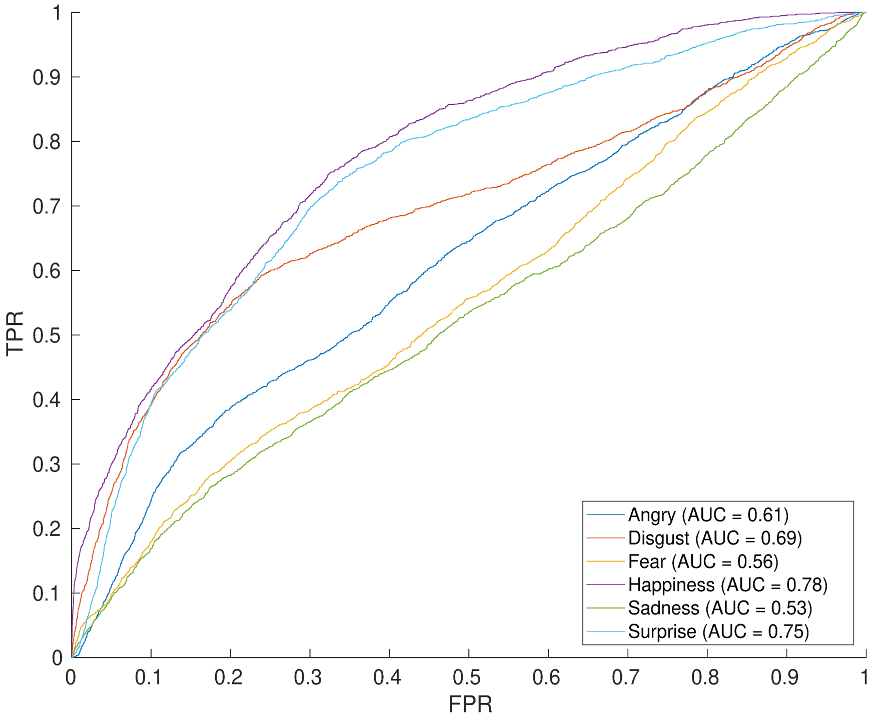


Figure 6.81: Person-independent ROC graph for RML with k = 17.

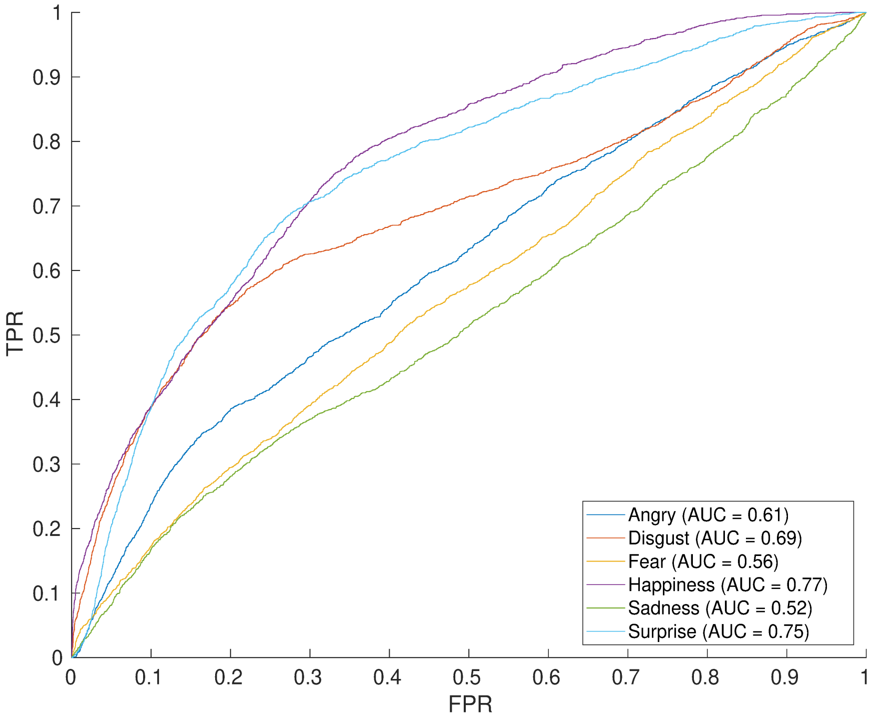


Figure 6.82: Person-independent ROC graph for RML with k = 19.

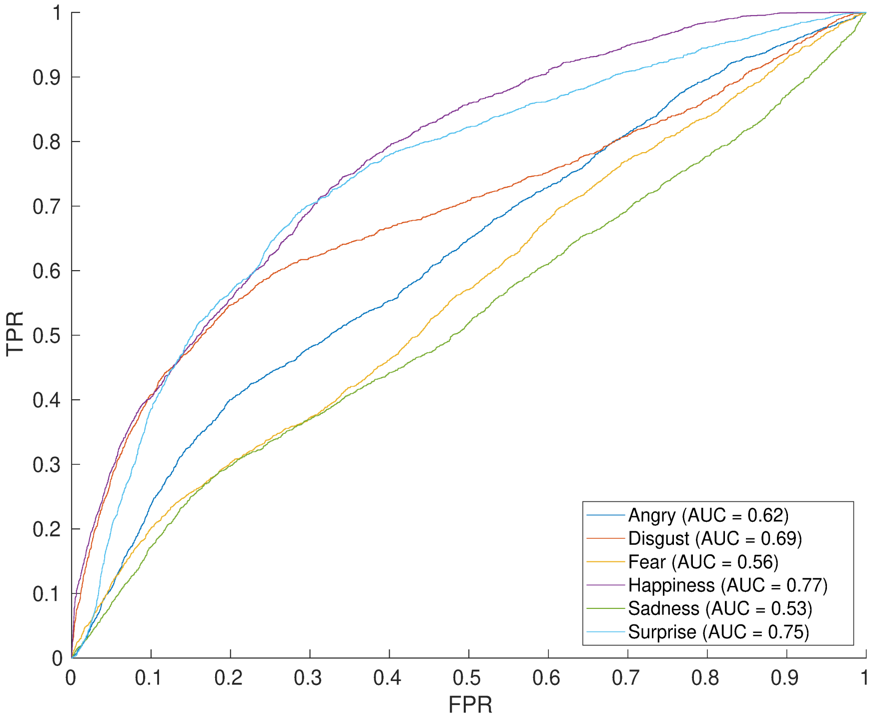


Figure 6.83: Person-independent ROC graph for RML with k = 21.

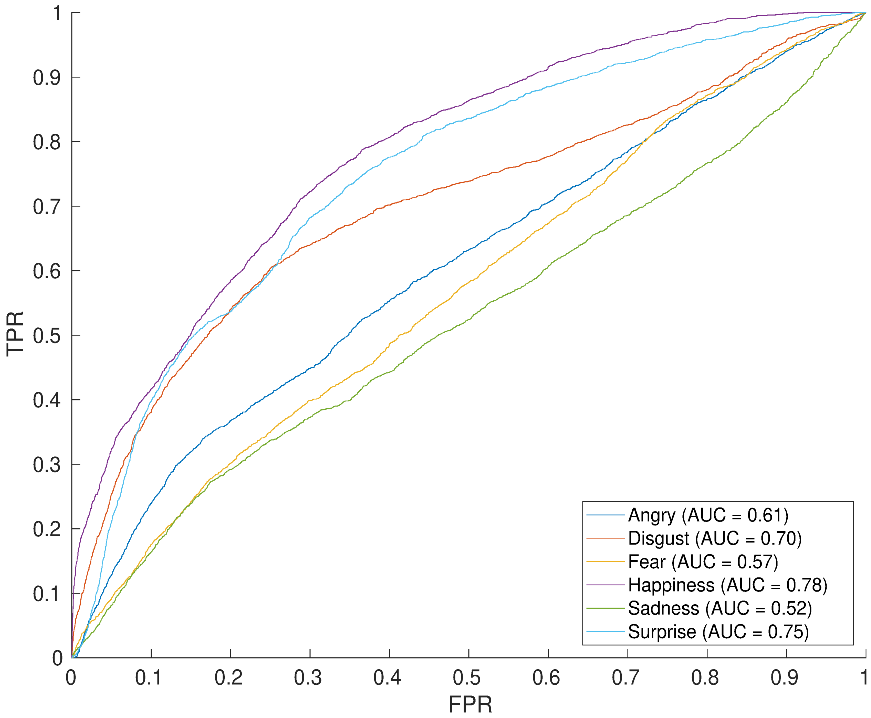


Figure 6.84: Person-independent ROC graph for RML with k = 23.

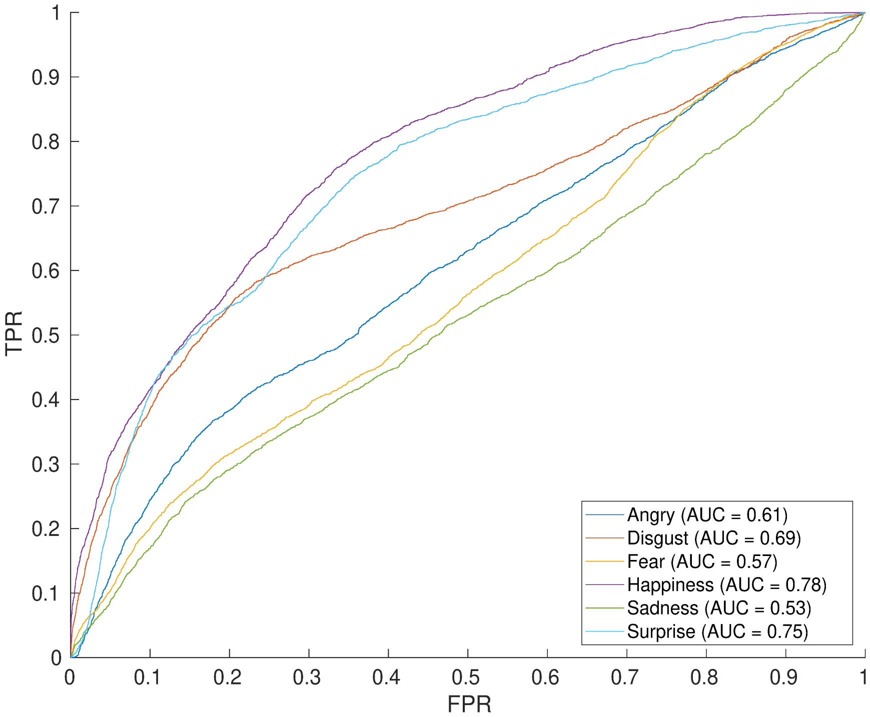


Figure 6.85: Person-independent ROC graph for RML with k = 25.

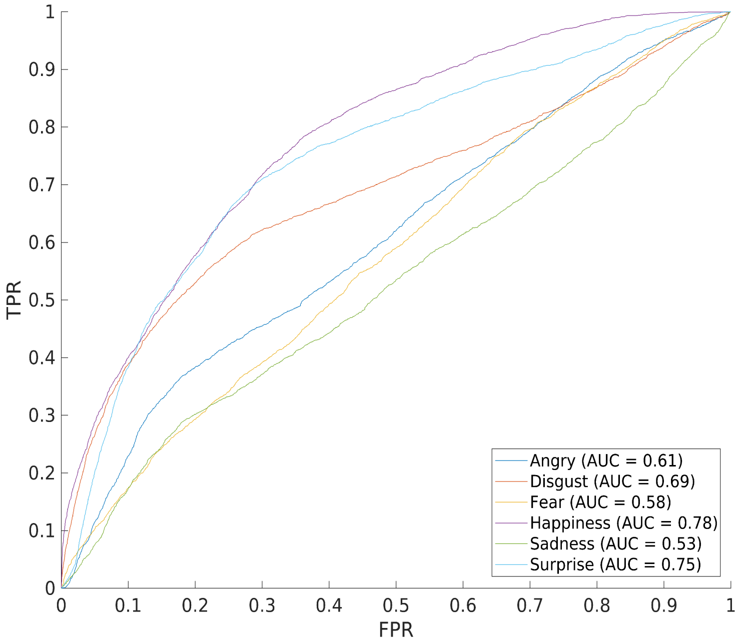


Figure 6.86: Person-independent ROC graph for RML with k = 27.

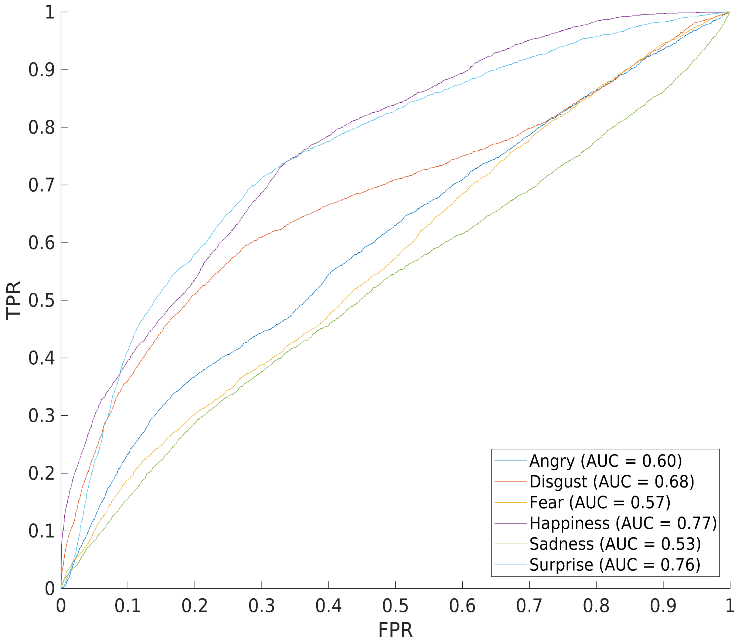


Figure 6.87: Person-independent ROC graph for RML with k = 29.

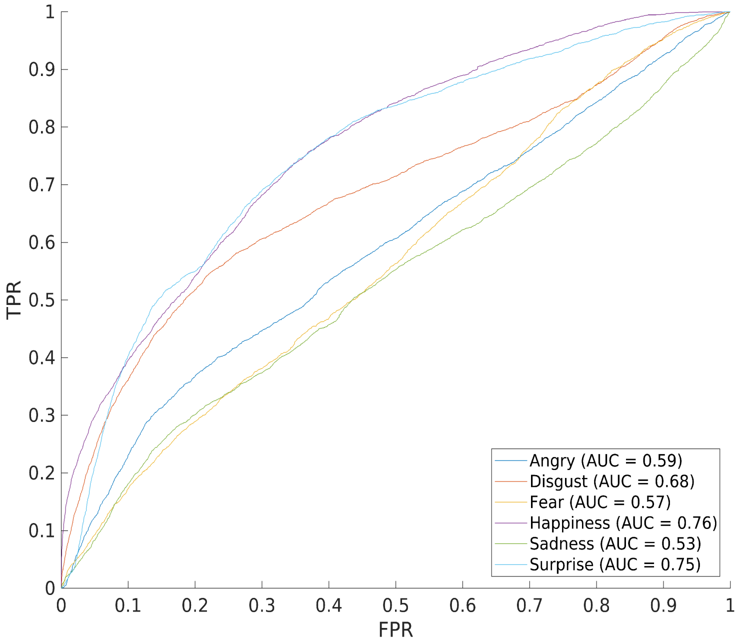


Figure 6.88: Person-independent ROC graph for RML with k = 31.

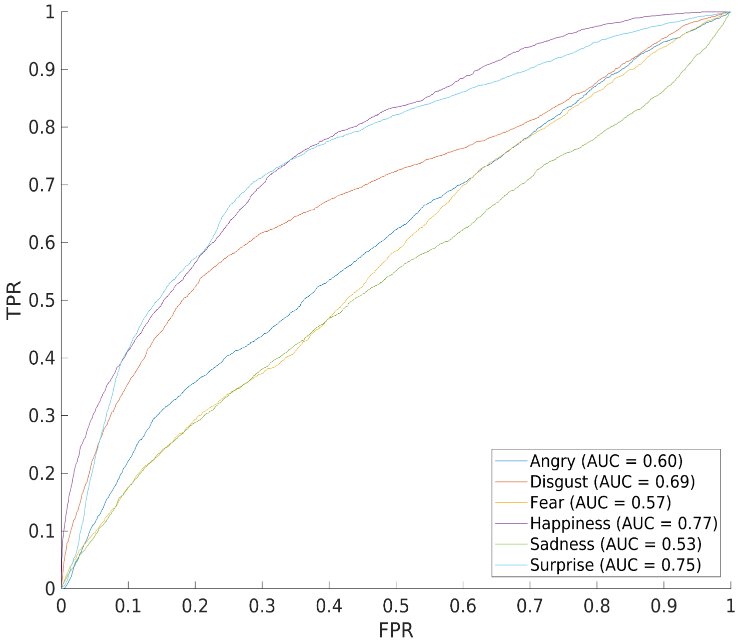


Figure 6.89: Person-independent ROC graph for RML with k = 33.

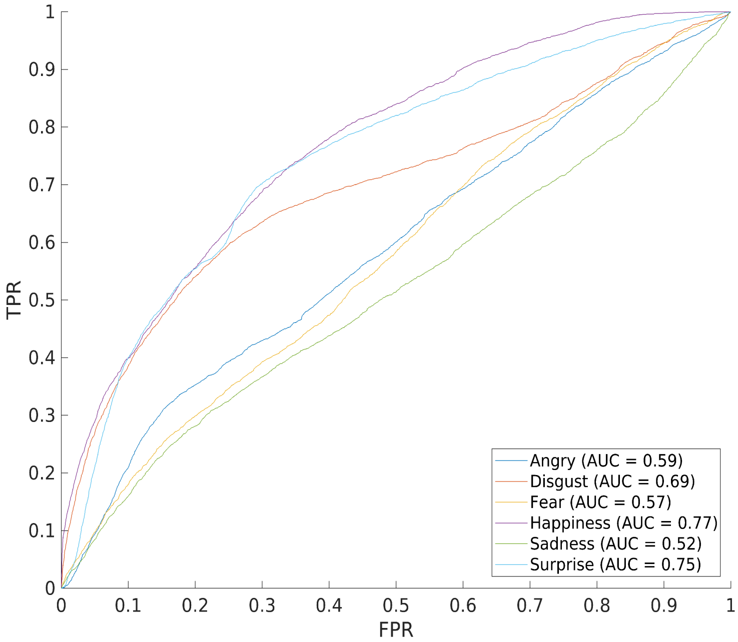


Figure 6.90: Person-independent ROC graph for RML with k = 35.

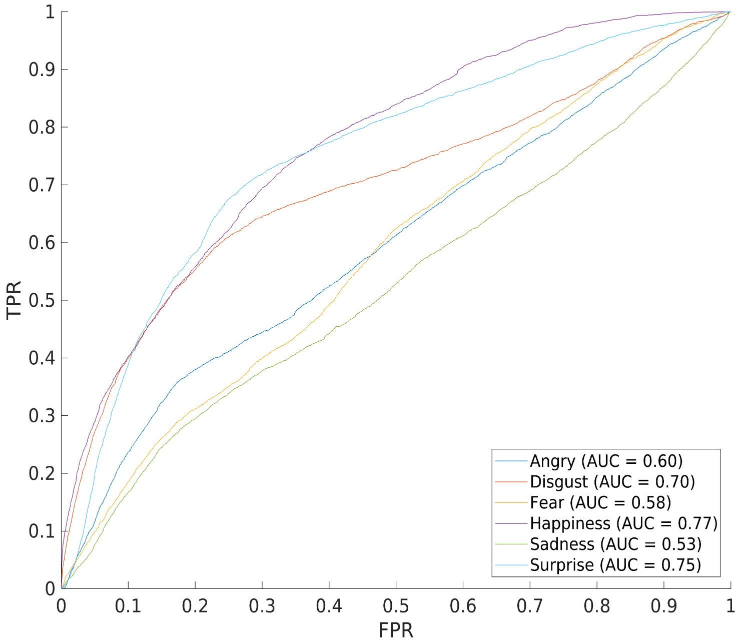


Figure 6.91: Person-independent ROC graph for RML with k = 37.

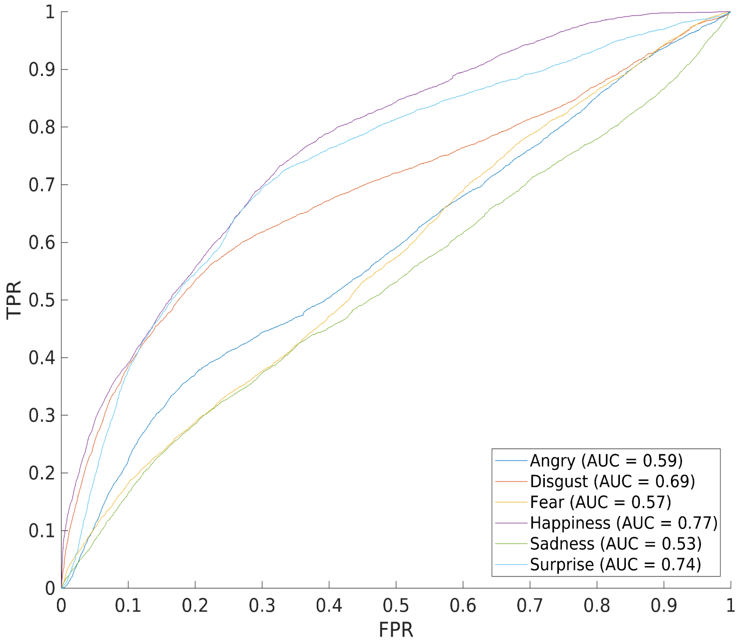


Figure 6.92: Person-independent ROC graph for RML with k = 39.

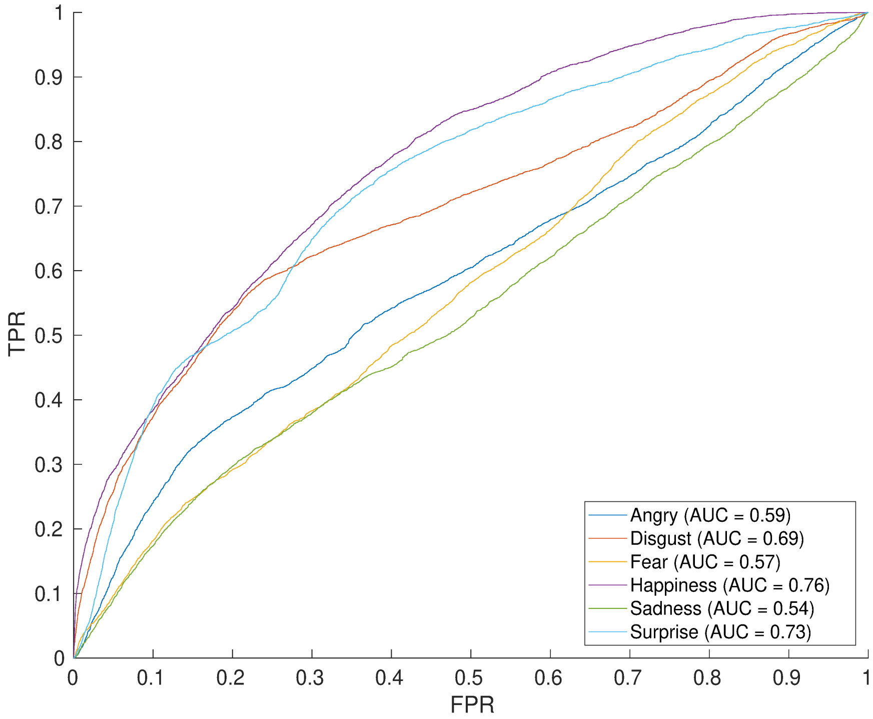


Figure 6.93: Person-independent ROC graph for RML with k = 41.

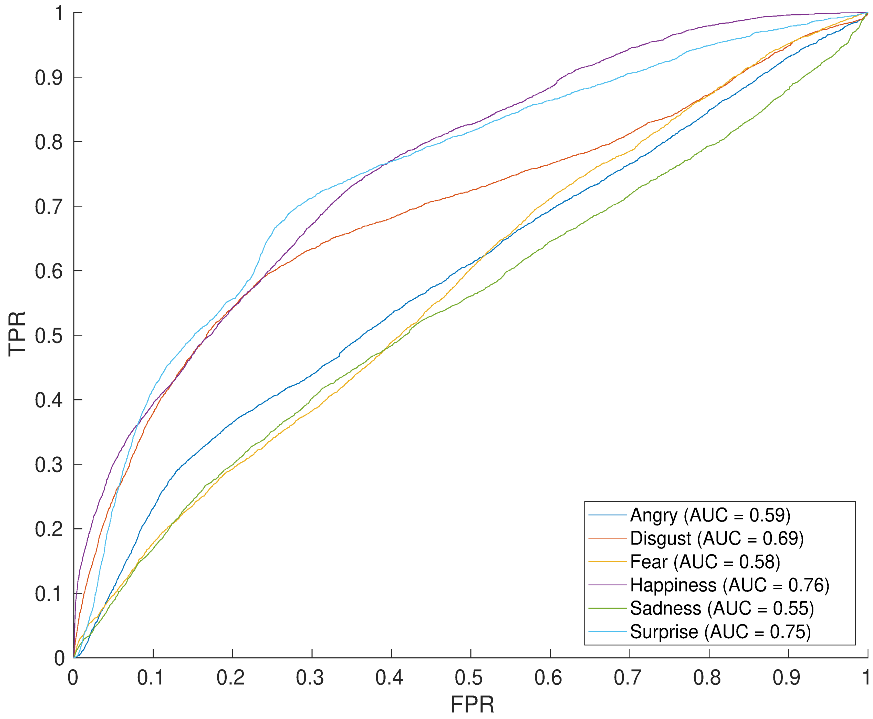


Figure 6.94: Person-independent ROC graph for RML with k = 43.