UNITED KINGDOM · CHINA · MALAYSIA

Feature Extraction via Random Recurrent Deep Ensembles and its Application in Group-Level Happiness Estimation

Submitted April 2017, in partial fulfillment of
the conditions of the award of the BSc (Hons) Computer Science degree.

Yichen PAN

Supervisor: Guoping QIU

The University of Nottingham Ningbo China,
School of Computer Science

I hereby declare that this thesis is all my own work, except as indicated in the text:

Signature: _____

Date: _____/_____/_____

Abstract

In this thesis, we present a novel ensemble framework to extract highly discriminative feature representation of image and its application for group-level happiness intensity prediction in wild. In order to generate enough diversity of decisions, we train n convolutional neural networks by bootstrapping the training set and extract n features for each image from them. A recurrent network is then used to remember which network extracts better feature and generate the final feature representation for one individual image. We use several group emotion models (GEM) to aggregate face features in a group and use a fine-tuned support vector regression (SVR) to get the final results. Through extensive experiments, the great effectiveness of our Random Recurrent Deep Ensembles (RRDE) is demonstrated in both structural and decisional ways. Our best result yields a 0.55 root-mean-square error (RMSE) on validation set of HAPPEI dataset, significantly better than the baseline of 0.78. Meanwhile, I also build an online RESTful API called GREP (GRoup Emotion Parser) and a website for demonstration purpose.

Acknowledgements

First and foremost, I would like to thank my adviser, Dr Guoping QIU, for his mentorship and advice over the past two years. I would also like to thank Shitao TANG for his contribution to this project. The work presented in this thesis is the joint work of our collaboration. Finally, I would like to thank my family for their unconditional support both before and during my undergraduate career.

Contents

List of Figures

List of Tables

Abbreviations

RRDE	Random Recurrent Deep Ensemble
GREP	GRoup Emotion Parser
GEA	Group Emotion Analysis
CNN	Convolutional Neural Network
DCNN	Deep Convolutional Neural Network
LSTM	Long Short-Term Memory
SVM	Support Vector Machine
SVR	Support Vector Machine Regression
RMSE	Root-Mean-Square Error
BDBN	Boosted Deep Belief Network
PPDN	Peak-Piloted Deep Network
CENTRIST	CENsus Transform hISTogram
GPU	Graphics Processing Unit
RNN	Recurrent Neural Network
Bagging	Bootstrap Aggregating
SGD	Stochastic Gradient Descent
GEM	Group Emotion Model
PLS	Partial Least Squares
OLS	Ordinary Least Squares
SMO	Sequential Minimal Optimization
L2-loss	Squared Hinge Loss
RESTful	Representational State Transfer
SPA	Single-Page Application

To the best four years of my life.

Chapter 1

Introduction

1.1 Motivation

With the growing popularity of data sharing and broadcasting websites such as Instagram, YouTube and Flickr, a significant number of images and videos of social events such as a wedding or a party are uploaded to the public internet. These videos and images are recorded in different conditions and often involve multiple subjects. Few attention has been paid to such diverse scenarios in the affective computing community.

Groups are a rich source of emotions. Automatic Group-level Emotion Analysis (GEA) in images is of great importance as it has a wide variety of applications. In e-learning applications, group-level affective computing can be used to adjust the presentation style of a computerized tutor when a learner is bored, interested, frustrated, or pleased. In terms of social monitoring, for example, a car can monitor the emotion of all occupants and engage in additional safety measures, such as alerting other vehicles if it detects the driver to be angry. With regard to management of images, dividing images into albums based on facial emotion is a good solution to searching, browsing and managing images in multi-media systems [3]. It also has practical meaning in key-frame detection and event detection [4].

Estimating the emotion of a person can sometimes be very difficult even for humans due to the subtle differences in expressions between the more nuanced emotions (such as sadness and fear). As a result, finely-tuned and optimized features extracted from images are of great importance for a classifier to make good predictions. Deep learning based approaches, particularly those using convolutional neural networks (CNNs), have been proved to be particularly useful and applicable at image-related tasks recently, due to their ability to extract good representations from data.

1.2 Description of Proposed Approach

In this thesis, we demonstrate the efficacy of our proposed RRDE in group-level emotion recognition sub-challenge based on HAPPEI database. The task is to infer the happiness intensity of the group as a whole on a scale from 0 to 5 from a bottom-up perspective. We first introduce our CNN ensembles to extract several efficiently representative features of each face. Then we conduct feature aggregation on each face using LSTM. The proposed method will selectively memorize (or forget) the components of features which are important (or less important). Then we conduct face-level estimation using our trained SVR on the compact feature representation of each face. Various group emotion models are explored, including mean encoding and weighted fusion framework based on top-down features, such as sizes of faces and the distances between them. Note that the proposed method caters to aggregating information from multiple sources including different number of faces in one image. Our best result exhibits a RMSE of 0.55 on the validation set of HAPPEI database, which compares favorably to the RMSE of 0.78 of the baseline.

The rest of the thesis is organized as follows. Chapter 2 describes related previous work. Chapter 3 describes in detail our RRDE framework together with the underlying principles. Chapter 4 discusses the experiments and evaluates the results, and Chapter 5 concludes the thesis by giving the summary and indicating the direction of future work.

As a final note, the construction of the whole group-level happiness intensity estimation framework is done in collaboration with Shitao TANG and advised by Professor Guoping QIU. Thus, in describing the system, I generally refer to it as “our system and “our work to reflect the joint nature of this work.

Chapter 2

Background and Related Work

Automatic facial expression recognition has attracted much attention since nineteenth century, while Darwin demonstrated already in 1872 the universality of facial expressions and their continuity in human and animals [5]. In the decades ago, facial expression recognition and analysis was primarily a research subject of psychology. However in 1978, because of its various applications, and the development in related research areas such as face tracking and face detection, as well as the recent evolution in computational power, automatic facial expression recognition has gained more and more interests in terms of Compute Science and Pattern Recognition. Given the fact that much progress has been made in automatic facial expression recognition, more research starts to focus on expanding it into emotion recognition and analysis. However, most of work focused on emotion analysis in only limited on individual-level.

GEA is a multi-dimensional problem under various conditions. There are two recent surveys which give detailed summary of related methodologies and the state-of-the-art in affect analysis [6][7]. In terms of the automatic affect recognition in diverse conditions, the EmotiW challenge series is an authoritative organization which focused on offering a benchmark for researcher to evaluate the performance of their methods on 'in the wild' data. 'In the wild' here refers to the real-life, uncontrolled conditions such as diverse background situations (indoor/outdoor), illumination conditions, head motion occlusion, multiple people in an image, spontaneous expression and etc. Image based facial expression recognition sub-challenge of Emotion Recognition in the Wild Challenge 2015 (EmotiW 2015) [8], and Group-level Emotion Recognition (GReco) sub-challenge of EmotiW 2016 [9] provide good benchmark for our research in GEA.

As discussed before, GEA is an intersection of many research areas, including face detection, facial expression recognition and analysis, and affective computing. A rough literature review has been conducted and discussed as follows.

2.1 Individual-level Emotion Analysis

Individual-level emotion analysis normally adopts a three-stage training process comprised of feature learning, feature selection, and classifier construction [10]. These features can be either hand-designed or learnt from training images. The most effective extracted features of images are selected so as to accelerate the classification process and increase the generalization capability. Finally, a classifier is constructed upon the selected feature representations of each face. Recently, there has been much evidence showing that emotion analysis can benefit from performing these three stages together with certain multi-layer (i.e., deep) neural network architectures.

One of the top three submissions in 2015 EmotiW competition [8] is [11]. They use a transfer learning approach for DCNN architectures. Their proposed method uses two different DCNN architectures that are pre-trained for the task of generic object detection (i.e., AlexNet [12] and VGG-CNNM-2048). The DCNNs are trained on the ImageNet database. During the first-stage fine-tuning, the FER 2013 database is used. A second-stage fine-tuning is based only on the training dataset of the EmotiW database. By doing this, it helps adapt the network weights to the characteristics of the SFEW sub-challenge. Both architectures are found to be improved after the fine-tuning stages. The best architecture achieves a 55.6% recognition rate, more than 15% (in absolute terms) better than the baseline.

Inspired by the effectiveness of the so-called multi-column DCNN (MCDNN) architecture [13] in multiple visual classification tasks, the MCDNN is applied for facial expression recognition "in-the-wild" in [14]. The standard MCDNN is made of a group of DCNNs with a simple averaging decision rule. Various network architectures, input normalization and random weight initialization are tested to find the optimal way to introduce diversity. Finally, in order to train more diverse decisions, an ensemble rule based on an exponentially-weighted decision fusion is applied. The best architecture achieves a recognition rate of around 57% and wins the EmotiW 2015.

The majority of deep learning techniques applied for emotion analysis "in-the-wild" revolve around learning static discriminative feature representations via CNNs and using score aggregation for final prediction [15]. [10] implements the three training stages iteratively in a unified loopy framework through a novel Boosted Deep Belief Network (BDBN). More recent studies adopt CNN architectures that permit feature extraction and recognition in an end-to-end framework. For instance, [16] employs a multiple deep CNN ensemble. [17] uses three inception structures proposed by [18] in convolution for facial expression recognition. [19] introduces the Peak-Piloted Deep Network (PPDN) to implicitly learn the evolution from non-peak to peak expressions.

2.2 Non-face Features: Body and Context

Recently, body expressions, in the form of body movement and postures, have received more and more attention from the affective computing field [20]. They have shown to be particularly effective for predicting the level of emotion [21].

Researches in the affective computing field have shown that the displayed emotion is heavily dependent on context, such as the location of the person and his connection with the outside world [22]. Thus, except for the face and body information, context information is becoming increasingly popular for automatic affect recognition [23]. Especially, in a group setting where multiple people are inherently involved, more complex contextual situations exist. This complexity does not only result from each individuals identity, location and task but also from interpersonal dynamics. For instance, either who the person is with or what others are doing at that time, will play an important role when analyzing the group emotion. Once the contextual information is extracted from the image, it can be directly used as a cue for affect analysis or fused with other attributes, such as facial expressions and body motion.

2.3 Group-level Emotion Analysis

Social psychology studies suggest that group emotion can be conceptualized in different ways. Generally, group emotion analysis problems can be solved using two approaches. The firs is *top-down* approach, where emotion emerges at the group level and follows by individual participants of the group [24]. The second is *bottom-up* approach, where overall emotion of group is assumed to consist of emotion expression of each individual member [25].

[26] proposes a hybrid approach, which combines top-down and bottom-up components, where top-down information is extracted using scene descriptors and bottom-up information is extracted by analyzing the face of the members of a group. Later, [27] comes up with a multi-modal method combining face, upper-body and context information, where face/upper-body is viewed as the bottom-up component and scene as top-down component. For representing an image, information aggregation is proposed to encode multiple peoples information for group-level image.

In the recent GReco Sub-challenge of EmotiW 2016, the baseline feature used is the CENsus TRansform hISTogram (CENTRIST) descriptor [28]. [29] proposes a framework

based on ensemble of features in LSTM and ordinal regression, which wins the sub-challenge. The second place is the method from [30], which aims at exploring geometric features extracted from face images. They also use partial least square regression to infer the group-level happiness intensity. The method of the third place [31] is about adopting a LSTM based approach and fine-tuning the AlexNet model [12] based on the HAPPEI database [1] and Static Facial Expressions in the Wild (SFEW) database [32].

2.4 Database

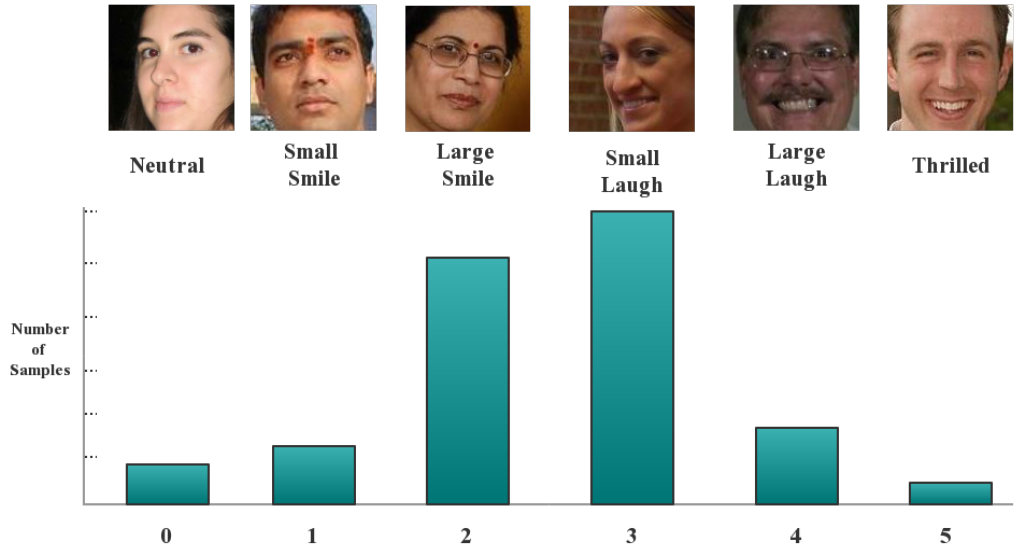
The database used in the experiment is the HAPpy PEople Images (HAPPEI) database [1], which is a fully annotated database (by human labelers) for the task of group happiness level prediction. Images in this database are collected from Flicker by using a Matlab based program that automatically searches particular images associated with groups of people and predefined events. For those downloaded images, a Viola-Jones object detector trained on different data is executed on the images. Only images containing more than one subject were kept. Figure 2.1 shows a collage of images from the HAPPEI database.

FIGURE 2.1: **A collage of images from the HAPPEI database** [1]

Every image in the database is given a label between $[0, 5]$. These six discrete numbers correspond to six stages of happiness: Neutral, Small Smile, Large Smile, Small Laugh, Large Laugh and Thrilled, as shown in Figure 2.2. During the manual labelling stage, if the teeth of a member of a group were visible, the face was often labelled as a *Laugh* (happiness intensity ranges from 3 to 4). If the mouth was open wide, the face was labelled as *Thrilled* (of happiness intensity 5). A face with a closed mouth was assigned

the label *Smile* (happiness intensity ranges from 1 to 2). [1] The LabelMe [33] based annotation tool was also used for labelling.

FIGURE 2.2: **Samples in HAPPEI database**



The HAPPEI database is a highly unbalanced database, with most of the faces/groups labelled as 3, but very few of them labelled as 0 or 5.

2.5 Convolutional Neural Networks

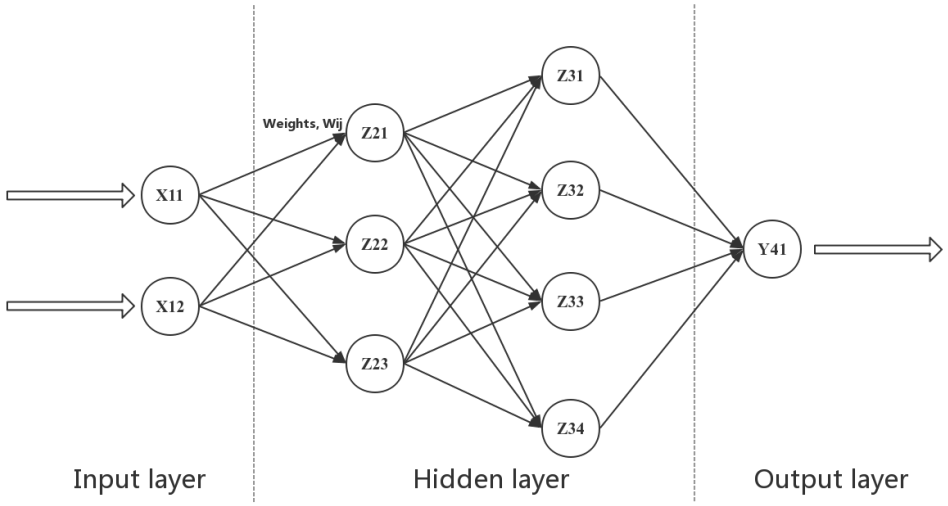
In the system, the first step is to extract representative feature vector for each face image. One prerequisite is that feature representations are not those clever and hand-designed features or extracted from very intricate models based on large amounts of prior knowledge. In our work, we base on CNN to build the whole feature extraction framework. This section provides a brief overview of both basic feed-forward neural networks as well as CNN.

2.5.1 Feed-Forward Neural Networks

It is necessary to describe the basic feed-forward neural network before investigating into the more complex convolutional neural network. A detailed discussion of feed-forward neural network could be found in [34]. Given a supervised learning scenario and a set of labeled data $(x^{(i)}, y^{(i)})$ where $x^{(i)}$ and $y^{(i)}$ are the feature and label of training sample i, a neural network can be seen as a nonlinear function $h_W(x)$ of the input vector variable

x. A weights matrix W is used to parameterize the function $h_W(x)$ and can be tuned to fit the input data. Figure 2.3 shows a simple neural network architecture with two input units or neurons, denoted x_{11} and x_{12}, and one output unit $y = h_W(x_{11}, x_{12})$.

FIGURE 2.3: **Typical feed-forward neural network architecture**



A feed-forward network generally has three layers: the input layer, hidden layer and output layer, as shown in Figure 2.3. Neurons (or perceptrons) are arranged in layers, with the first layer taking in inputs and the last layer producing outputs. The middle layers have no connection with the external world, and hence are called hidden layers. Each neuron in one layer is connected to every neuron in the next layer so that information is constantly "fed forward" from one layer to the next, and this is why these kinds of networks are called feed-forward networks.

Each neuron is a computational unit that takes in the combinations of values of the neurons from the preceding layer as input. For example, the input to the neuron z_{21} is the computed combination of x_{11} and x_{12} denoted as a_{21}. In detail, let $x_1, x_2, ..., x_n$ denote the inputs to a neuron z_j, the combined input to z_j is:

$$a_j = \sum_{i=1}^{n} w_{ji} x_i + b_j \tag{2.1}$$

where w_{ji} describing the weight between z_j and the input neuron x_i. The b_j is a bias associated with neuron z_j. The final value of neuron z_j is the result of applying a nonlinear activation function [35] to a_j:

$$z_j = h(a_j) = h(\sum_{i=1}^{n} w_{ji} x_i + b_j) \tag{2.2}$$

where h is the nonlinear activation function, which often includes the sigmoid and the hyperbolic tangent functions.

The weights w and biases b in the neural network are the main components of the model. Given a labeled training set $(x^{(i)}, y^{(i)})$, the goal of training process is to learn the weights w and biases b so that the loss or objective function is minimized. The standard approach to learning the parameters w and b is the error back-propagation algorithm [34].

2.5.2 Convolutional Neural Network

A convolutional neural network is a type of feed-forward neural network where the connectivity between neurons conforms to the organization of the animal visual cortex. There are three essential characteristics that capture the most distinguished difference of the CNN from the simple feed-forward neural networks: local receptive fields, weight sharing, and spatial pooling or subsampling layers [36]. The following discussion of these three characteristics is based on the context of a computer vision problem, where the input to the CNN consists of a single N-by-N image. For example, this input is 32×32 pixel values.

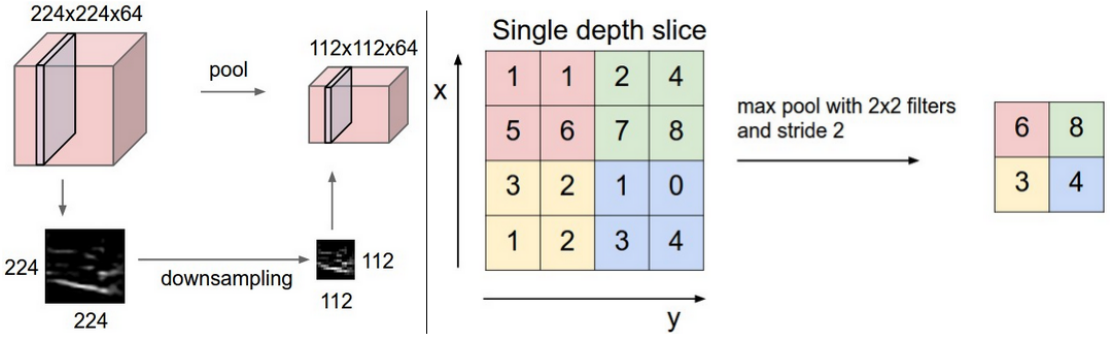
In the basic neural networks mentioned above, each neuron is fully connected to the subsequent layer, which means the value of each neuron fully depends on the whole input image. However, in terms of visual recognition, it often makes more sense to make use of the local substructure within the image. For example, the up-left part of the image tends to be weekly related to the bottom-right part. Inspired by the unbalanced correlation nature of the image, the CNN architecture exploits the local substructure within the image by forcing each neuron to depend only on a spatially local subset of the variables in the previous layer [36]. The set of nodes in the input layer that have influence on the activation of a neuron is referred to as the neuron's *receptive field*. The receptive fields could be explained as the part of the image that the neuron "sees" in a more intuitive sense. Since the layers are not fully connected to each other, the CNN architecture has a sparser set of edges.

Weights in the connected edges are shared among various neurons in the hidden layers. In terms of a simple feed-forward neural network, each neuron in the network is the computation result of a weighted linear combination of its inputs, which can be treated as evaluating a linear filter over the input values. By contrast, the shared weights and local receptive fields make each neuron evaluate the same filter over multiple sub-windows of the input image. In this case, the CNN can be regarded as effectively learning a set of filters $F = \{F_i | i = 1, ..., n\}$, each of which is applied to all of the sub-windows within

the input image. By adopting the same set of filters over the entire image, it manages to constrain the network to learn a general representation of the input data. Another benefit of weight sharing is a more efficient training stage by substantial reduction in the number of free parameters in the CNN.

The last property that characterizes CNNs is the subsampling or pooling layers. Among those typical non-linear functions for pooling, the max pooling is the most common approach. It works by partitioning the input image into several smaller rectangles that are not overlapped with each other, and for each such sub-region, outputting the maximum value. This is shown in Figure 2.4. The intuition behind this is that the exact location of a feature is less important than its rough location relative to other features. Pooling operation brings extra benefits for two perspectives: reduce computation for upper layers and bring a small degree of translational invariance into the model. More specifically, the pooling layer could help gradually reduce the spatial size of the representation. Reduction of the feature size will directly lead to reduction of the number of parameters and amount of computation in the network, and hence finally prevent overfitting.

FIGURE 2.4: **Typical max pooling layer in CNN**



Left: the input volume of size $224 \times 224 \times 64$ is pooled with filter size 3, stride 2, into volume of $112 \times 112 \times 64$. Right: Max pooling with a stride of 2

A typical CNN is often comprised of multiple layers, alternating between pooling and convolution layers. For example, a convolution-pooling layer can be stacked on top of the outputs of another convolution-pooling layer. By stacking convolution-pooling layers, a multilayered or deep architecture can be constructed. Finally, after several convolutional and max pooling layers, the high-level reasoning in the neural network is done via fully connected layers. Neurons in a fully connected layer have full connections to all activations in the previous layer. Their activations can hence be computed with a matrix multiplication followed by a bias offset. From an intuitive perspective, the low-level convolutional filters, such as those in the first convolutional layer, can be treated as low-level encoding of the input data. In terms of the image as input, these low-level filters are likely to work as simple edge filters. As for those higher layers, the CNN has the ability to learn more and more complicated structures. With multiple layers and

a large number of filters, the CNN is capable of extracting powerful representation of input data. To train a CNN, the standard technique of error back-propagation is used [34].

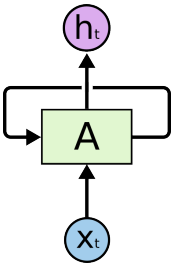
CNNs have wide applications in many tasks such as handwriting recognition [37] and visual object recognition [38]. With the rapid advancements in distributed system and GPU (graphics processing units) computation, much deeper and more powerful CNNs achieve state-of-the-art performance on standard benchmarks [12]. Therefore, by exploiting the representational capacity of CNNS together with ensemble learning, we are able to construct simple, but powerful and robust framework for feature extraction.

2.6 Long-Short Term Memory

2.6.1 Recurrent Neural Network

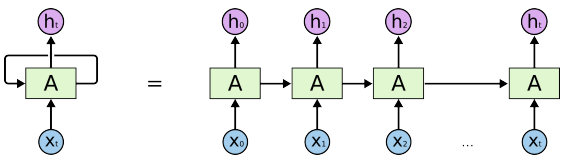
Based on the intuition that human thoughts have persistence, recurrent neural network (RNN) allows information to persist by nesting loops in the architecture. As shown in Figure 2.5, a loop allows information to be passed from one step of the network to the next.

FIGURE 2.5: **Loops in recurrent neural networks**



A recurrent neural network can be viewed as multiple copies of the same network as shown in Figure 2.6 [39]. With this chain-like nature, RNNs are particularly suitable for sequential tasks such as speech recognition, language modeling, translation, image captioning and etc.

FIGURE 2.6: **An unrolled recurrent neural network**



Basic RNNs work well when the gap between previous information and the present task is small. For example, building a language model to predict the next word based on the previous ones. However, when the task needs more context, where the gap becomes much larger, basic RNNs tend to be incapable of learning to connect the information. The underline principle is illustrated in Figure 2.7 and 2.8.

FIGURE 2.7: **Basic RNNs work well with short-term dependencies**

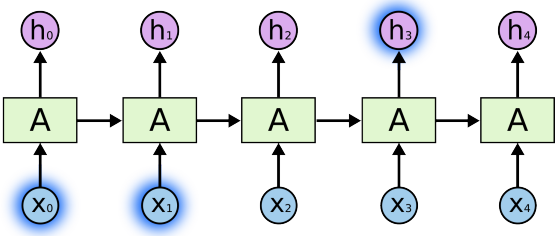
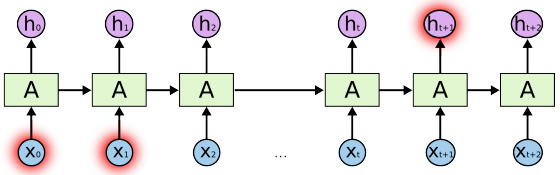


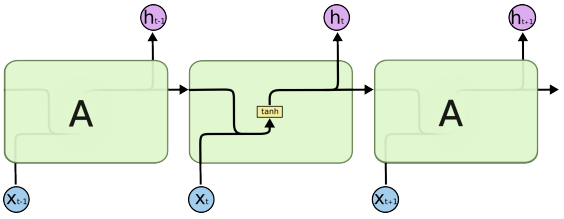
FIGURE 2.8: **Basic RNNs become incapable with long-term dependencies**



2.6.2 Long-Short Term Memory

Long short-term memory (LSTM) is a special kind of RNN architecture proposed in 1997 [40]. LSTMs are explicitly designed to deal with the long-term dependency problem. All RNNS have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module has a very simple structure, such as a single tanh layer.

FIGURE 2.9: **The single-layer repeating module in a standard RNN**



By contrast, LSTMs have a different repeating module structure that contains four interacting neural network layers, as shown in Figure 2.10.

The key to LSTMs is the cell state, which is represented by the horizontal line on the top of the diagram.

FIGURE 2.10: **The four-layer repeating module in a LSTM**

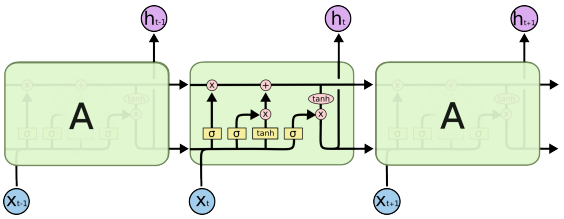
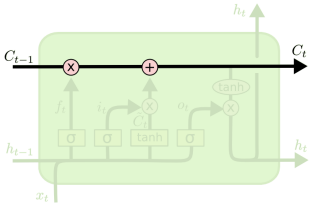
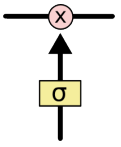


FIGURE 2.11: **The cell state in a LSTM**



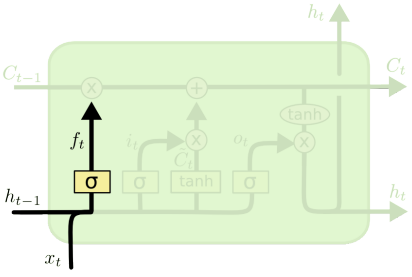
Gates are a way to control the information flow among the cell state. They consist of a point-wise multiplication operation and a sigmoid neural net layer. The output from the sigmoid layer is a number between zero and one, indicating how much of each component should be let through. A value of zero forbids everything going through while a value of one allows everything going through.

FIGURE 2.12: **The gate in a LSTM**



A standard LSTM first begins with the step to forget information, deciding what information should be excluded from the cell state. This is decided by a sigmoid layer called the *forget gate layer*. It computes a number between zero and one for each number in the cell state C_{t1} based on h_{t1} and x_t.

FIGURE 2.13: **Forget gate layer in a LSTM**

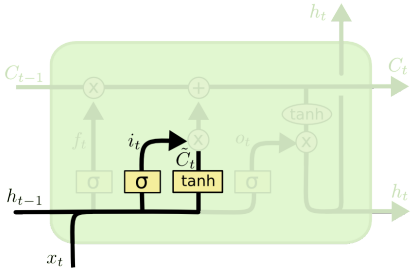


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

The second step is to decide what new information should be stored in the cell state. This action involves two function layers. The first is a sigmoid layer called the *input*

gate layer which decides which values will be updated. The second is a tanh layer which creates a vector of new candidate values, $\tilde{C}_t$, that could be added to the state.

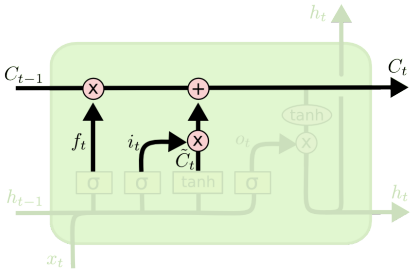
FIGURE 2.14: **Input gate layer and tanh layer in a LSTM**



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

The next step is to update the old cell state, $\tilde{C_{t-1}}$, into the new cell state C_t. First, the old state is multiplied by f_t to forget the things decided to forget earlier. Then, $i_t * \tilde{C}_t$ is added to the new candidate values, scaled by how much is decided to update each state value.

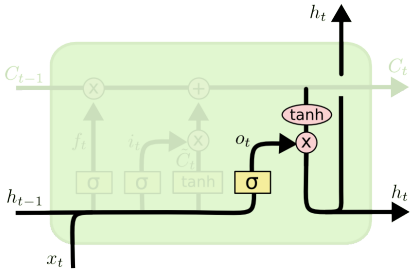
FIGURE 2.15: **Cell state update in a LSTM**



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Finally, the output is decided by a sigmoid layer and a tanh layer.

FIGURE 2.16: **Final output of a repeating module in a LSTM**



$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

2.7 Ensemble Learning

Ensemble learning methods are focused on obtaining better predictive performance by using multiple learning algorithms. Common types of ensembles include bootstrap aggregating (Bagging) [41], boosting [42], and stacking [43].

Considering the combination of deep learning and ensemble learning, there have not been many applications so far. Basically, one intuitive way is to use a deep neural network (DNN) as if they were a single classifier and combine them with other classifiers to build an ensemble. Another more complicated way is to use a DNN to extract features and use the output of the n-1 layer as input to an ensemble of classifiers. Particularly, [44] proposes a mixture of deep convolutional neural networks (DCNN), called MixDCNN, to perform classification. More recently, [45] and [46] explore using 'generalist network performance statistics to inform the design of ensemble-of-expert architectures for classification. [47] generalizes the Multiple Choice Learning [48] [49] paradigm to jointly learn ensembles of deep networks that minimize the oracle loss directly.

Chapter 3

Proposed Approach and Methodology

3.1 Face Detection

For HAPPEI database, faces are detected using the approach proposed by [50]. However, we find each face only contains face bounding box without other information such as head-pose estimations of yaw and pitch, facial points and face confidence score. In this case, we conduct face detection by ourselves using OpenCV's Viola-Jones frontal face detector [51]. We set the minimum face size to 25×25 pixels, and use the Intraface library [52] to detect 49 facial points. We discard false positive faces based on the score of the alignment model provided by Intraface; any detection with a score lower than 0.3 is considered a non-face.

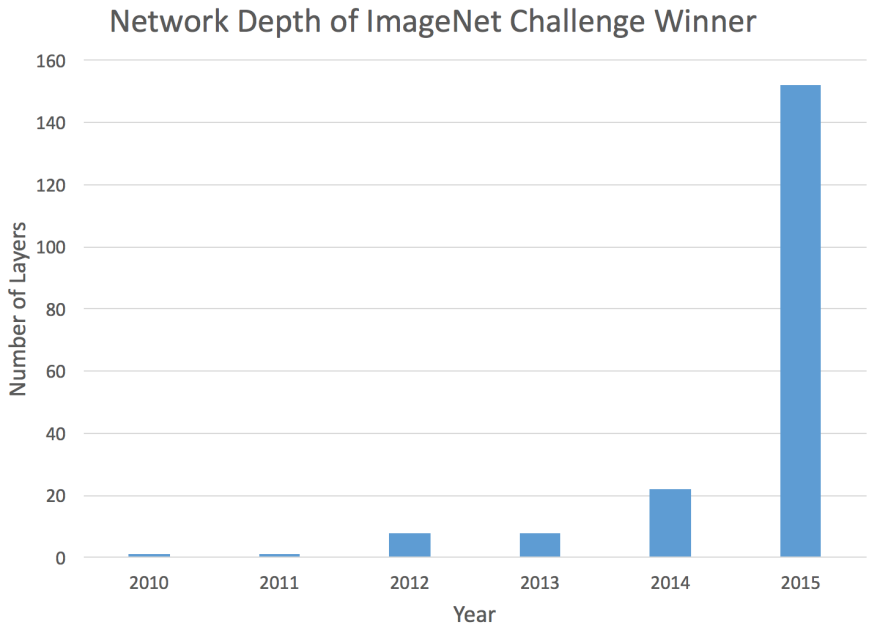
3.2 Individual Facial Feature Extraction

3.2.1 Residual Network (ResNet)

Deep convolutional neural networks [12] have resulted in a series of breakthroughs in Computer Vision recently. There has been much evidence [53] [18] revealing that network depth plays an important role in the effectiveness of the model, and the leading results on the recent challenging ImageNet dataset all make use of very deep network architectures (shown in Figure 3.1). Considering more and more effort focused on the depth of the model, a question arises: Does stacking more layers lead to a better networks? One of the obstacles to answering this question was the notorious vanishing/exploding gradients

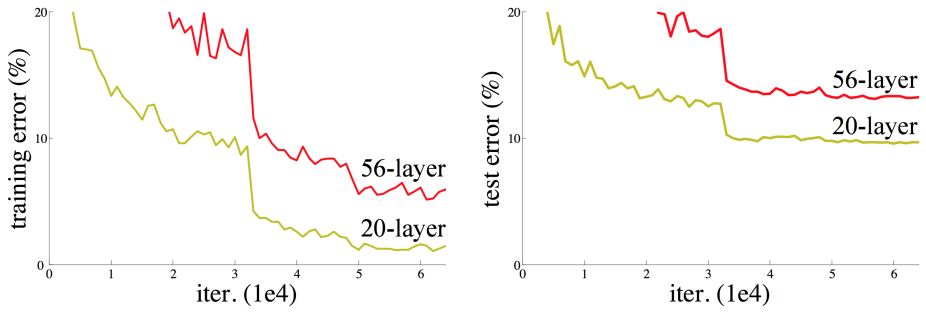
problem [54], which hinderes convergence of the model from the beginning. Fortunately, thanks to normalized initialization [54] [55] [56] and intermediate normalization layers [57], the problem has been largely addressed. They allow networks with tens of layers to start converging for stochastic gradient descent (SGD) with back-propagation [37].

FIGURE 3.1: **The trend of deeper networks**



While deeper networks have the ability to converge, a degradation problem has occurred: accuracy gets saturated and then degrades rapidly as the network depth increases. It is unexpected that overfitting will not lead to such degradation, and adding more layers to a suitably deep model results in higher training error, as reported in [58] [59], and thoroughly verified by the experiments conducted by [2]. Figure 3.2 [2] clearly shows the degradation problem.

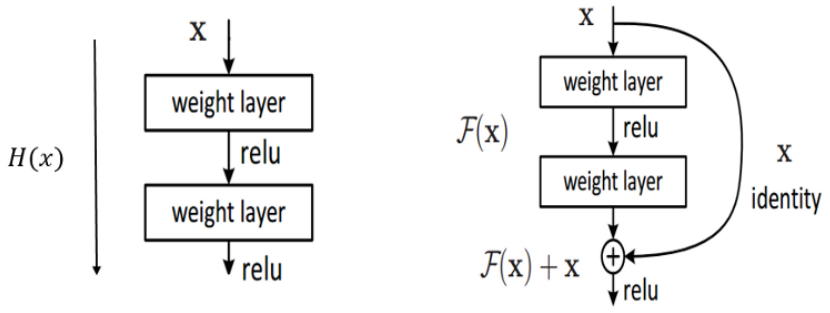
FIGURE 3.2: **Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks** [2]



The deeper network has higher training error, and thus test error.

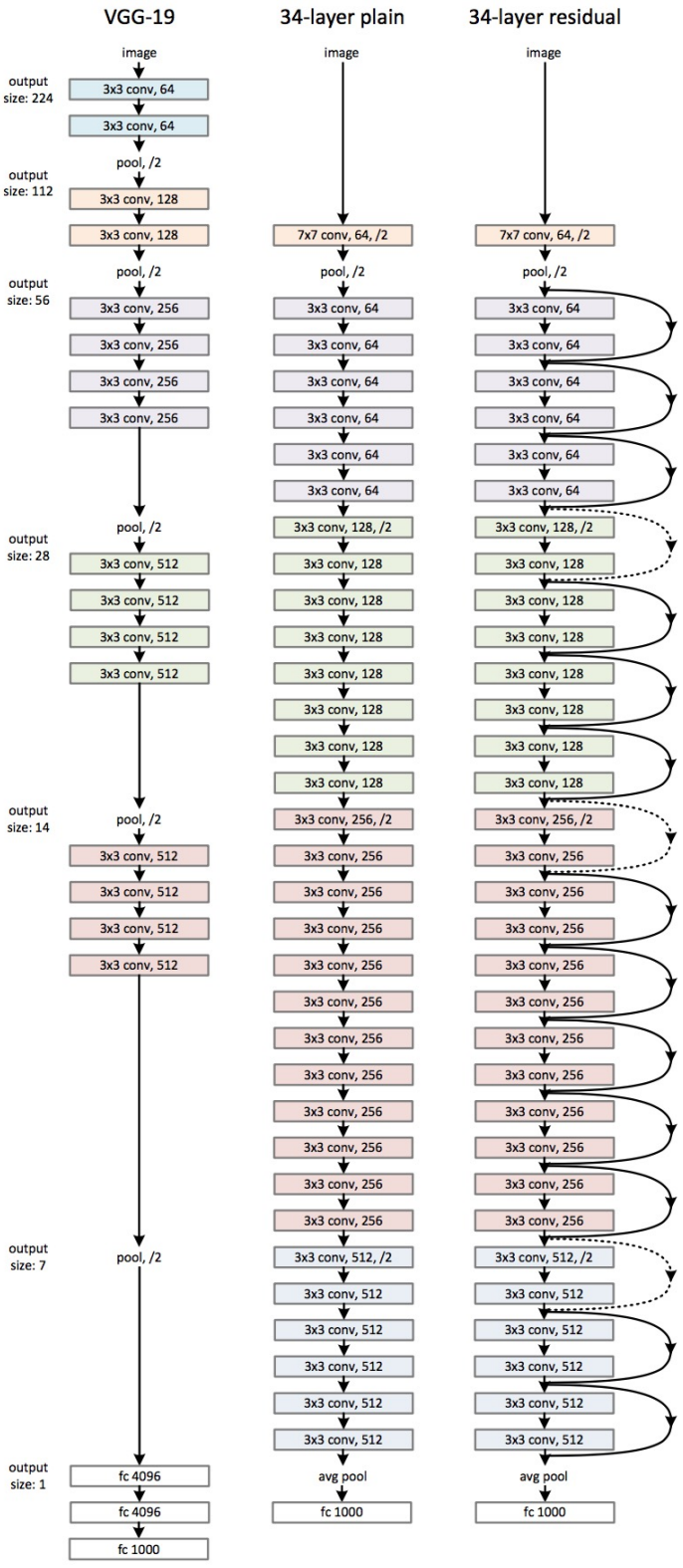
In order to solve this problem, [2] proposes a *deep residual learning* framework. Rather than attempting to fit each stacked layers directly with a desired underlying mapping, they manage to make these layers adhere to a residual mapping. Formally, denoting the desired underlying mapping from input to output as $H(x)$, they use nonlinear function $F(x) := H(x) - x$. As shown in the Figure 3.3, at the output of the second weight layer, x is added to the $F(x)$, and then $F(x)+x$ is passed through Rectified Linear Unit (ReLU). The "$+x$" at the end is realized by shortcut connection [60] [60] [61]. It allows the gradient to pass backwards directly. By stacking these layers, the gradient could theoretically skip over all the intermediate layers and reach the bottom without being diminished. It is actually similar principle introduced with LSTM cells. By doing this, it makes it much easier to optimize the residual mapping compared with the original mapping.

FIGURE 3.3: **Normal CNN vs residual CNN**



The plain baselines (Figure 3.4, middle) are mainly based on the VGG nets [53] (Figure 3.4, left). The convolutional layers mostly have 3×3 filters and adhere to the two design rules: (i) layers have the same number of filters for the same output feature map size; and (ii) if the feature map size is halved, the number of filters is doubled so as to preserve the time complexity per layer. Downsampling is performed directly by convolutional layers that have a stride of 2. The network ends with a global average pooling layer and a 1000-way fully-connected softmax layer. The total number of weighted layers is 34 in Fig. 3.4 (middle).

FIGURE 3.4: **Example network architectures of normal CNN and ResNet-34 for ImageNet** [2]



Left: the VGG-19 model [53] (19.6 billion FLOPs) as a reference. Middle: a plain network with 34 parameter layers (3.6 billion FLOPs). Right: a residual network with 34 parameter layers (3.6 billion FLOPs). The dotted shortcuts increase dimensions.

The ResNets won the 1st places in several tracks in ILSVRC & COCO 2015 competitions: COCO detection, COCO segmentation, ImageNet detection, and ImageNet localization. Due to the outstanding performance of ResNet and its attractive features, such as easy to optimize and fewer parameters, we choose to use ResNet as our CNN architecture.

3.2.2 Random Recurrent Deep Ensembles (RRDE)

[47] uses stochastic multiple choice learning to train n networks and improve the accuracy of cifar-10 by 2-3%. This method proves that CNN can be used as base classifier in ensemble learning. Inspired by this framework, an original feature extraction framework RRDE is proposed. It first trains n CNNs for each image, then extracts one vector representation from each network, and finally aggregates n feature vectors for each image into one compact feature vector using LSTM.

3.2.2.1 Bootstrapping

Although random weight initialization and Stochastic Gradient Descent(SGD) can also make diverse CNNs, these CNNs turn out to be very similar if they are trained with the same dataset. In this case, the performance is not gained very much when aggregating them together. This is one main cause of the relatively worse performance of CNN in ensemble learning. [47] uses stochastic multiple choice learning to address the problem. They select the network with the minimum loss and backward it, which makes the difference between CNNs more distinct. However, it is not effective when used in a relatively small dataset like HAPPEI dataset.

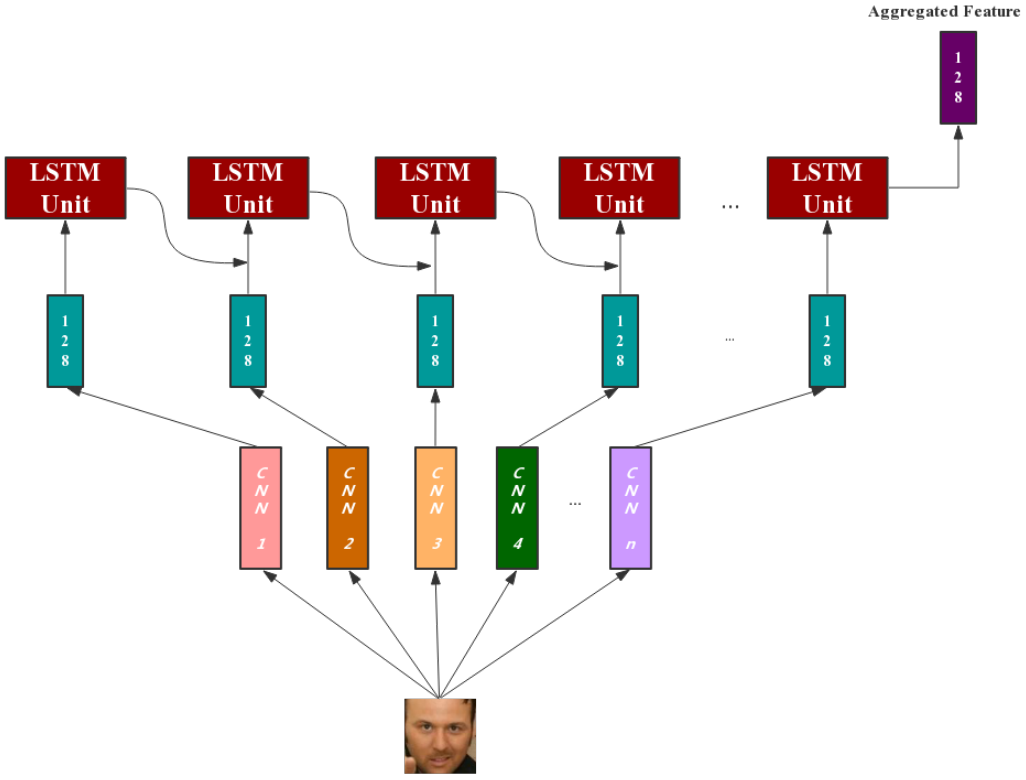
In order to make each CNN diverse, we adopt the first step of random forest, bootstrapping [62]. For every class, a random sample is selected repeatedly B times with replacement from the training set. After balancing the dataset, 380 images (the minimum number of images in a class) for each class are selected and for each network, bootstrapping is performed once so that each network is trained with different set of data (6*380=2280 images).

3.2.2.2 LSTM based Feature Aggregation

One straightforward way to aggregate the features is mean encoding. However, due to the diversity of these CNNs, for the same image, the difference between features extracted by different network can be huge. Extreme values have large impact on the final result. Therefore, mean encoding is not an ideal method for aggregation in this

case. Inspired by the success of LSTM in sequencing learning [63], it is believed that LSTM can be used for aggregating these features. Illustration of one LSTM cell has been shown in Figure 2.1 and 2.2. A cell memory C is used to memorize good feature representation of face feature vectors when scanning the sequence of feature vectors. The forget gate will decide how much of the content in the cell memory should be forgotten and the input gate in LSTM is employed to determine how much the new feature should be fed into the cell memory. The output gate decides how the features in the cell memory are converted to output features. The pipeline for LSTM based feature aggregation is shown in Figure 3.5.

FIGURE 3.5: **LSTM based feature aggregation pipeline**



3.3 Group Emotion Model (GEM)

After obtaining aggregated feature representation of each individual face in an image, we now combine them in order to get an estimation of the overall happiness of the group in image. In this subsection we discuss our GEMs for group-level estimation before feeding into the final regressor.

3.3.1 Mean of Face-level Estimations

One naive way for overall group-level happiness estimation is to use the mean of all face-level estimations in the group as introduced by [1].

$$g = \frac{\sum_{i=1}^{s} f_i}{s} \tag{3.1}$$

where g is the happiness intensity estimation of the group, s is the total number of detected faces in this group and f_i is the happiness intensity estimation for face i.

3.3.2 Mean Encoding

Another straight forward feature aggregation is mean encoding. We use the average of the face features as the feature representation of the whole image.

3.3.2.1 Context-based Weighted GEM

In above two GEMs, both local information, such as the level of occlusion of a face, and global information, such as the relative position of people in the image, are ignored. In this case, the effectiveness is compromised because all detected faces are assumed to contribute equally to the group-level estimation, which has been shown not generally the case [1]. In order to take into consideration the bottom-up and top-down components, the significance s_i of a face i is estimated using the following equation:

$$s_i = \frac{\Theta_i}{\delta_i} \tag{3.2}$$

where Θ_i represents the level of occlusion of face i, and δ_i represents the relative position of the face i among the group.

Based on the size of the face, Θ_i is further equal to the size of the bounding box of face i (in pixels). Based on the positions of all faces in the group, δ_i is further equal to:

$$\delta_i = \sum_{j=1}^{n} ||c_i - c_j|| \tag{3.3}$$

where c_j and c_i represent the coordinate of the centroid of corresponding face.

Equation (3.2) essentially incorporates social context features when estimating the group happiness intensity by normalizing the occlusion level of a face by the sum of its Euclidean distance from all other faces. In this case, small faces which are located away

from the group are penalized, while larger faces which are closer to all others are assigned a higher significance. If n = 1, then significance is set to 1. In the experiment, this context-based weight scheme is further cooperated together with mean of face-level estimation and mean encoding GEM.

3.4 Regression

In terms of statistical modeling area, regression (analysis) is aimed at estimating the relationships among variables. It is directly related to machine learning since it has wide usage in prediction.

Normally, a regression model has three variables:

1. One regression parameter vector, denoted as β. It is assumed to be a (p+1)-dimenssional vector where β_0 represents the constant (offset) term.
2. One *regressand* or *dependent variable*, denoted as Y. It is assumed to a n-dimensional vector.
3. One *regressors* or *independent variables* variable, denoted as X. It is assumed to a $n \times p$ matrix.

A regression model can be expressed as below, where the form of function f is based on knowledge about the relationship between Y and X.

$$Y \approx f(X, \beta) \tag{3.4}$$

In our work, several regression techniques are compared including Linear Regression, Logistic Regression, Support Vector Regression (SVR), and Partial Least Squares Regression (PLS regression).

3.4.1 Linear Regression

In linear regression, the relationship between variables are modeled using linear predictor functions with data-based estimated parameters. Such models are called linear models [64]. In most common cases, given the value of X, the conditional mean of Y is estimated to be an affine function of X. Similar to most forms of regression analysis, linear regression is focused on the conditional probability distribution of Y given X.

Given a data set $\{y_i, x_{i1}, \ldots, x_{ip}\}_{i=1}^{n}$ of n samples, a linear regression model assumes that the relationship between the dependent variable y_i and the p-dimensional vector of regressors x_i is linear. To model this relationship, an error variable ε_i is used to add noise to the linear relationship. The model in vector form is expressed as:

$$Y = X\beta + \varepsilon \tag{3.5}$$

The main task in linear regression is parameter estimation. There are several estimation techniques varied in computational cost:

Least-squares estimation and related techniques Among least-squares related estimation techniques, ordinary least squares (OLS) is the simplest and thus most common estimator. The OLS method minimizes the sum of squared residuals, and results in a closed-form expression for the estimated value of the unknown parameter β:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y} = \left(\sum \mathbf{x}_i \mathbf{x}_i^\top \right)^{-1} \left(\sum \mathbf{x}_i y_i \right). \tag{3.6}$$

Maximum-likelihood estimation and related techniques Maximum likelihood estimation can be performed when the distribution of the error terms is known to belong to a certain parametric family $_\theta$ of probability distributions [65]. When $_\theta$ is a normal distribution with zero mean and variance , the resulting estimate is identical to the OLS estimate.

Bayesian linear regression Bayesian linear regression is based on the framework of Bayesian statistics. Particularly, the regression coefficients β are assumed to be random variables with a specified prior distribution. Additionally, the Bayesian estimation process produces an entire posterior distribution rather than only a single point estimate for the "best" values of the regression coefficients. In this case, it could completely describe the uncertainty surrounding the quantity.

3.4.2 Logistic Regression

Logistic regression [65] is a regression model where the dependent variable is categorical. It used a logistic function to estimate probabilities and measure the relationship between the categorical dependent variable and one or more independent variables. From a easy point of view, the logistic regression can be seen as finding the β parameters that best

fit:

$$y = \begin{cases} 1 & \beta_0 + \beta_1 x + \varepsilon > 0 \\ 0 & \text{else} \end{cases} \tag{3.7}$$

The regression coefficients are usually estimated using maximum likelihood estimation. [66]

3.4.3 Support Vector Regression

Support Vector Machine (SVM) can also be used as a regression method with all distinct and characterized features of the algorithm. The Support Vector Regression (SVR) [67] uses the same principles, such as maximal margin, as the SVM for classification while only a few minor parts are different. First of all, due to the fact that output becomes a real number, it is very difficult to predict the output directly. In terms of regression, a margin of tolerance ϵ is set to specify the epsilon-tube within which no penalty is associated in the training loss function with points predicted within a distance epsilon from the actual value. The constant c is the box constraint which controls the penalty on observations that lie outside the epsilon margin and prevents overfitting (regularization).

Similar to SVM, SVR can be classified into two categories: linear SVR and non-linear SVR. Particularly, non-linear SVR is useful for some regression problems that cannot adequately be described using a linear model.

In terms of solver algorithms, sequential minimal optimization (SMO) is the most popular approach for solving SVM problems.[68] In SMO, there is a series of two-point optimizations performed. In each iteration, a working set of two points are chosen based on a selection rule that uses second-order information. Then the Lagrange multipliers for this working set are solved analytically using the approach described in [69] and [70].

3.4.4 Partial Least Squares Regression

Partial least squares regression (PLS regression) is a regression method that bears some relation to principal components regression. It is focused on finding a linear regression model by projecting the predicted variables and the observable variables to a new space, instead of finding a hyperplane that maximize the variance between the dependent and independent variables. Since both the independent variables X and Y data are projected to new spaces, the PLS model is known as bilinear factor model.

The set of methods based on PLS has recently become very popular in computer vision [71], [72], [46]. PLS is used for dimensionality reduction before classification. [72] [46]

Effective usage of KPLS based regression in simultaneous dimensionality reduction and age estimation indicate that PLS based methods work well for face analysis when the feature vector is high dimension. [71]

Chapter 4

Experiments and Evaluation

In this chapter, results of the proposed RRDE based group-level happiness estimation are presented and our system is evaluated.

The whole pipeline framework of our RRDE based system is shown in Figure 4.1. There are four main stages including feature extraction, feature aggregation, using GEMs to leverage individual-level features, and finally performing regression to make estimations.

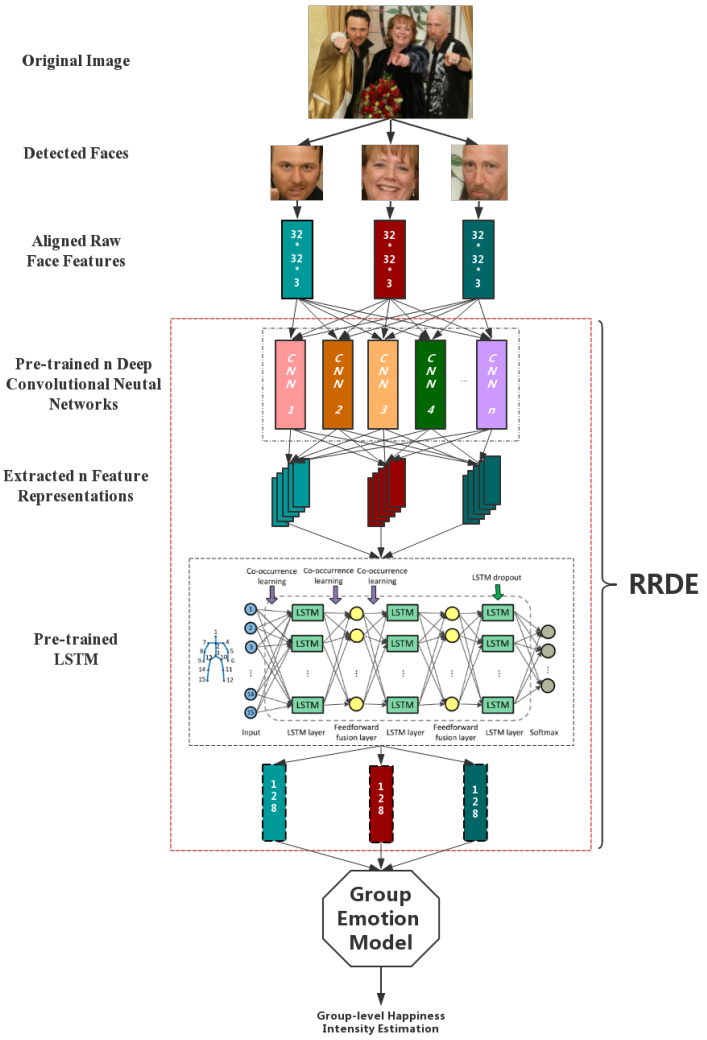
For all the experiments, the models are trained on the balanced training set of HAPPEI and tested on the validation set of HAPPEI. The summary of HAPPEI database is in Table 4.1database. The training and validation sets contain 2,638 images and 10,400 faces in total. However, as shown in Figure 2.2, the HAPPEI database is a highly unbalanced database, with most of the faces/groups labelled as 3, but very few of them labelled as 0 or 5. Training directly with such skewed data may compromise the generalization ability of the whole system.

In this case, we resample the existing database in order to create a more balanced subset, which is subsequently used as training set for face-level happiness estimation. More specifically, we aim at creating a uniform training distribution, which includes all the training examples of the most under-represented intensity quantization bin: those annotated as 0 (neutral) and those annotated as 5 (thrilled). Finally, we select 380 training examples per intensity bin, which is the number of faces annotated as 5. This results in a training subset with a total of 380 * 6 = 2280 training faces.

TABLE 4.1: **Summary of the HAPPEI database**

Dataset	Train	Validation
Number of Faces	1500	5549
Number of Groups	1138	4851

FIGURE 4.1: **The whole framework of proposed RRDE system**



The experiment environment is set in Python 2.7.10 with a number of toolkits including Theano(0.8.2), Scikit-learn(0.18.1), Numpy(1.12.0), Tensorflow(1.0.1) and etc. The training process is accelerated using GPUs.

4.1 Facial Feature Extraction

To estimate group happiness intensity, the most important step is to find good face feature representation. [30] uses the geometric feature of one face, which proves such feature is useful to measure the happiness intensity. However, this kind of feature is too hand-crafted to generalize on various qualities of face images. Convolution neural network (CNN) has showed its powerful capability to extract features [12], and deep residual network is currently one of the state-of-art CNN structures for object recognition

TABLE 4.2: **Simple ResNet architecture summary**

Output Map Size	32×32	16×16	8×8
Number of Layers	1+2n	2n	2n
Number of Filters	16	32	64

This architecture can be generalized to ResNet with $6n+2$ layers. In our ResNet-20, n is equal to 3.

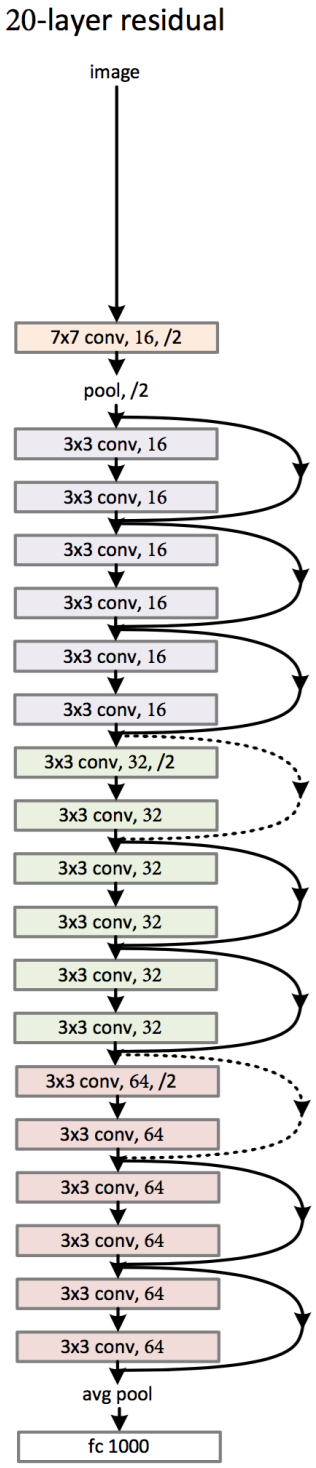
[2]. Therefore, ResNet is adopted as the base feature extraction method. Since HAPPEI database is of relative small size, using deeper network like ResNet-50 or ResNet-101 will result in overfitting easily. After experiments, a 20-layer ResNet is finally trained, which has the same structure as the one trained by [2] on Cifar-10 database. Cifar-10 database consists of 60000 images and 10 classes [73].

Particularly, the basic residual architecture follows the form in Figure 3.4 (right), but we adopt a simpler version called ResNet-20. The network inputs to our 20-layer ResNet are $32 \times 32 \times 32$ images, with every pixel value scaled to $[-1, 1]$. Then we use a stack of $6n$ (n equal to 3 in our RresNet-20) layers with 3×3 convolutions on the feature maps of sizes $32, 16, 8$ respectively. As a result, there are $2n$ layers for each feature map size. The numbers of filters are $16, 32, 64$ respectively. Convolutions with a stride of 2 are used to conduct subsampling. Finally, the network ends with a global average pooling, a 10-way fully-connected layer, and softmax. There are in total $6n + 2$ (equal to 20) stacked weighted layers. Table 4.2 summarizes the architecture, and Figure 4.2 shows the architecture of ResNet-20.

Shortcut connections are connected to the pairs of 3×3 layers (totally $3n$ shortcuts). For each face image, it is forwarded through the whole network. The activations from the penultimate layer is extracted as face image representation. The dimension of the extracted face feature is 64.

When training the network, a weight decay of 0.00001 is used. The initial learning rate is 0.01 and the batch size is 32. The network is trained for 20000 iterations in total and for every 5000 iterations, learning rate is multiplied by 0.1. Measuring happiness is a regression problem, so it is supposed to use Squared Hinge Loss (L2-loss) in this problem, but since the interests are only in the feature extracted by CNN, cross entropy loss can also be used. In the experiment, the feature extracted by the network with cross entropy loss are found to be better than L2-loss. Therefore cross entropy loss is used with 6 classes as the final loss layer. Bilinear interpolation is used to resize these images and randomly adjust the brightness, contrast, saturation when training so the data set is augmented to train a more robust network. In total, there are n networks and n feature representations for each image.

FIGURE 4.2: **ResNet-20 architecture**

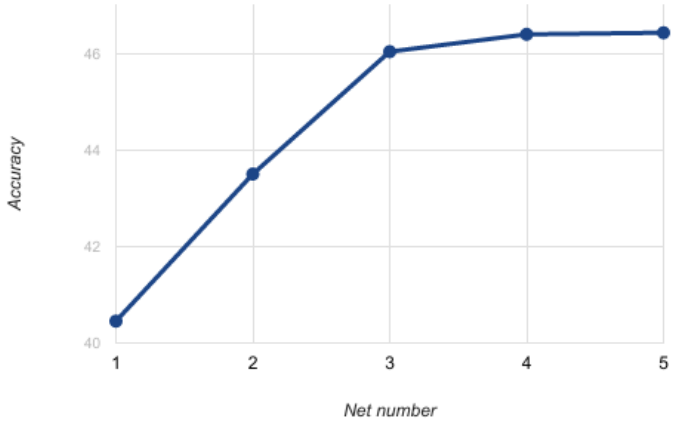


ResNet-20 consists of 7 layers with 16 filters, 6 layers with 32 filters and 6 layers with 64 filters.

When evaluating the effectiveness of network ensembles, the majority voting strategy is used so that each image is predicted as the maximum number of labels it gets. The model under different number of networks is evaluated and it is found that there is no

obvious improvement when using 6 or more networks. This may result from the fact that the ensembles tend to saturate quickly as the ensemble size grows. The result is shown in in Figure 4.3.

FIGURE 4.3: **Evaluation of performance of the ensembles when using majority voting strategy**

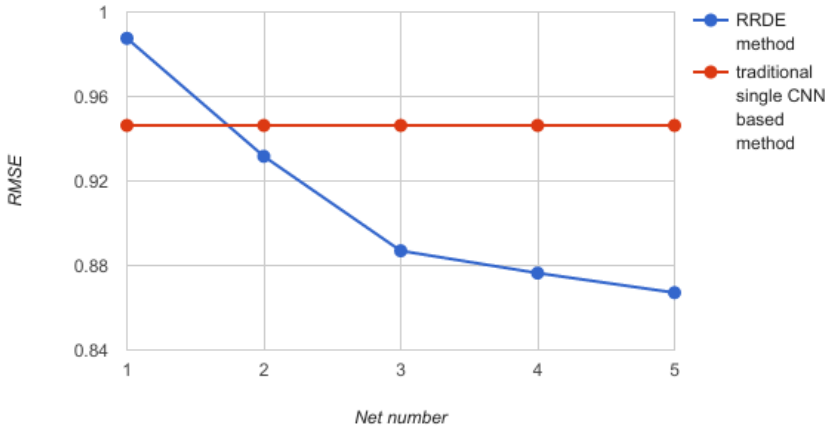


4.2 Facial Feature Aggregation

LSTM is used to scan features extracted by ResNet-20. Two cells are stacked into the LSTM unit, and the memory size for each cell is 128. The order of scanning these features is the same so that LSTM can remember which network extracts better feature for a specific image. For every face image in the whole database, it is fed to n networks and get n 64-dimensional feature vectors. Then LSTM scans these vectors and L2-loss is used as loss function when training. Finally, a 128-dimensional vector is extracted from the final LSTM output for the final group happiness intensity analysis.

In order to prove the effectiveness of the proposed RRDE method, a single ResNet-20 is trained using all training data of HAPPEI with the L2-loss as loss function, to compare with a RRDE. All the parameters of this single ResNet are the same as ResNets in the ensemble except for the iteration changed from 20000 to 30000. The result is shown in Figure 4.4. This single ResNet is named as ResNet-20-All because it will be used again in the group-level happiness estimation comparison.

FIGURE 4.4: **Comparison between the proposed RRDE method with the traditional single CNN method on individual happiness estimation**



When the number of networks in the ensembles is 1, the performance is relatively worse than the single ResNet trained with all data. This may result from the fact that all data provides more information for the network to generalize. However, with the number of networks in the ensembles increases, the performance improves significantly.

4.3 Group Emotion Modeling

After getting one efficient feature representation for each face image, the effectiveness of four GEMs mentioned in Chapter 3 is tested. As shown in Table 4.3, for each of four GEMs, including normal mean-encoding, normal mean of face-level estimations, context-based weighted mean-encoding and mean of face-level estimations, the model with different features extracted from ensembles of different size is tested.

One immediate observation from Table 4.3 is that the performance of GEM improves consistently as the ensemble size increases. With the increase in the size of ensembles, more reliable and effective feature representation could come from the strong consensus of multiple sub-networks in the ensemble. Another insight from Table 4.3 is that the feature extracted from the ensembles trained after bootstrapping is much more effective than that trained by all training data.

Mean-encoding model consistently exhibits more competitive results than the simple averaging of face-level estimations, which indicates that the extracted feature contains very useful information, that may otherwise be ignored by simple averaging face-level estimations.

TABLE 4.3: **Results of GEMs on the validation set**

Features	Group-level	RMSE (validation)
1network-All-RRDE	mean-encoding	0.6003
	weighted mean-encoding	0.5947
	mean of face-level estimations	0.6208
	weighted mean of face-level estimations	0.6117
1network-RRDE	mean-encoding	0.5885
	weighted mean-encoding	0.5875
	mean of face-level estimations	0.5956
	weighted mean of face-level estimations	0.5918
2networks-RRDE	mean-encoding	0.5794
	weighted mean-encoding	0.5767
	mean of face-level estimations	0.5921
	weighted mean of face-level estimations	0.5848
3networks-RRDE	mean-encoding	0.5659
	weighted mean-encoding	0.5646
	mean of face-level estimations	0.5877
	weighted mean of face-level estimations	0.5836
4networks-RRDE	mean-encoding	0.5715
	weighted mean-encoding	0.5647
	mean of face-level estimations	0.5923
	weighted mean of face-level estimations	0.586
5networks-RRDE	mean-encoding	0.5536
	weighted mean-encoding	**0.5479**
	mean of face-level estimations	0.563
	weighted mean of face-level estimations	0.5556

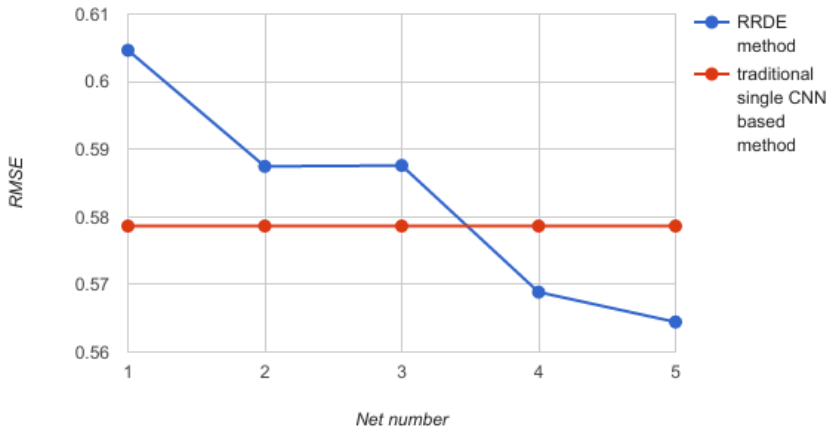
1network-All-RRDE refers to the features extracted from the RRDE with only one CNN trained on whole training set. The final regression is done by a SVR with penalty parameter C equal to 0.5, epsilon E equal to 0.13 and linear kernel type. The best result is **0.5479** achieved by a 5networks-RRDE with weighted mean-encoding.

The significance factor of each face, which takes into consideration the relative position of people in the image, and the level of occlusion of a face, exhibits a modest performance improvement. When compared to the simple GEM without weighting scheme, the addition of the face significance factor improves the RMSE. Since no training method is involved in the use of a simple averaging, this improvement may not be the result of overfitting, but may have to do with the efficiency of the feature.

Moreover, the performance of the proposed RRDE method on group-level happiness estimation with the traditional single CNN based method is also compared. The same CNN called ResNet-20-All mentioned in Figure 4.4 is used here. Since the output vector

from ResNet-20-All is of 64 dimension, the LSTM hidden units are resized from 128 to 64 so that the aggregated feature output from LSTM is of 64 dimension. The group-level estimation is based on mean-encoding GEM. The result is shown in Figure 4.5.

FIGURE 4.5: **Comparison between the proposed RRDE method with the traditional single CNN method on group-level happiness estimation**



Similar to the case of individual-level happiness estimation, the proposed RRDE method beats the traditional method quickly with the number of networks in the ensemble increases.

4.4 Regression

Several regression models mentioned in Chapter 3 have been compared. The grid search combined with cross-validation is used to find the optimal parameters of each regressor. As shown in the Figure 4.6 and Table 4.4 below, the optimal regressor is a SVR with penalty parameter C equal to 0.5, epsilon E equal to 0.13 and kernel type chosen as linear. For SVR, the penalty parameter C is ranged from 0.5 to 1.5, epsilon E is ranged from 0.05 to 0.15 and kernel function is chosen from linear, poly, rbf and sigmoid. In terms of Logistic Regression, the penalty parameter C is ranged from 0.5 to 1.5, the solver function is chosen from newton-cg, lbfgs, liblinear and sag.

FIGURE 4.6: **Confusion matrices of regressors**

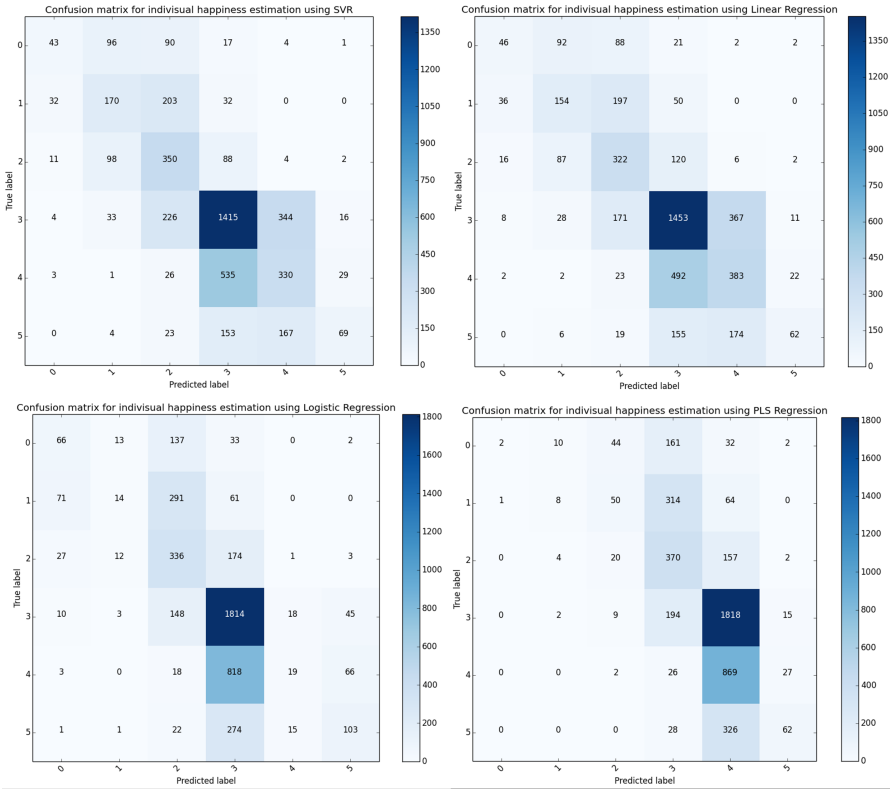


TABLE 4.4: **Comparison of regressors**

	SVR					Linear Regression				
	Precision	Recall	F1-score	Support	RMSE	Precision	Recall	F1-score	Support	RMSE
0	**0.46**	0.17	0.25	251		0.43	0.18	0.26	251	
1	**0.42**	**0.39**	**0.41**	437		0.42	0.35	0.38	437	
2	0.38	**0.63**	0.48	553		**0.39**	0.58	**0.67**	553	
3	**0.63**	0.69	0.66	2038	**0.867**	0.63	0.71	0.47	2038	0.8697
4	0.39	0.36	0.37	924		**0.41**	0.41	**0.41**	924	
5	0.59	0.17	0.26	416		**0.63**	0.15	0.24	416	
Avg/Total	0.52	0.51	0.5	4619		**0.53**	**0.52**	**0.51**	4619	

	Logistic Regression					PLS Regression				
0	0.37	**0.26**	**0.31**	251		0.67	0.01	0.02	251	
1	0.33	0.03	0.06	437		0.33	0.02	0.03	437	
2	0.35	0.61	0.45	553		0.16	0.04	0.06	553	
3	0.57	**0.89**	**0.7**	2038	1.0024	0.18	0.1	0.12	2038	1.2313
4	0.36	0.02	0.04	924		0.27	**0.94**	**0.41**	924	
5	0.47	**0.65**	**0.32**	416		0.57	0.15	0.24	416	
Avg/Total	0.46	0.51	0.42	4619		0.27	0.25	0.17	4619	

All four regressors are trained on the same training set, *train_5nets_LSTM_features*, and evaluated on the same validation set, *val_5nets_LSTM_features*. Each result in red corresponds to the best case. The best regressor is SVR with least RMSE equal to **0.867**

The final SVRs for group-level estimation experiments mentioned above have the same hyper-parameters, but are trained on corresponding training sets. The details of the best results, together with the baseline provided by the challenge [9], are depicted in Table 4.5.

To conclude this chapter, I present some example outputs of our RRDE based GEA

TABLE 4.5: **Final result**

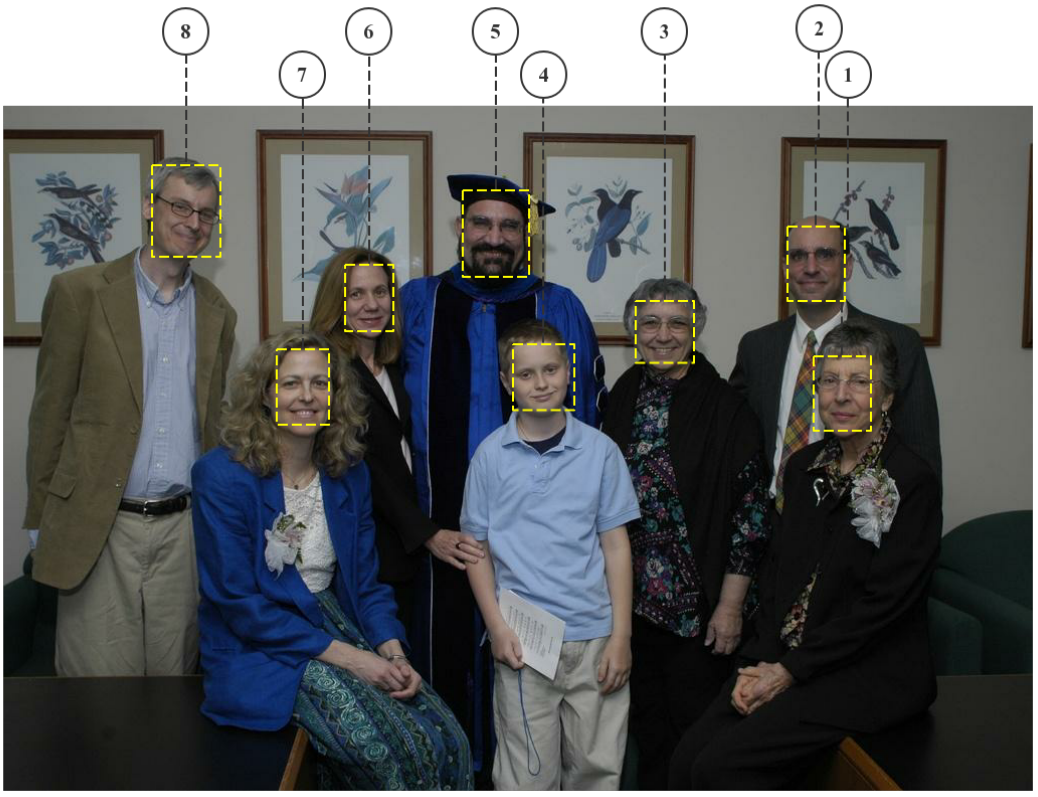
Model	RMSE
Baseline	0.78
5-networks RRDE	0.55

TABLE 4.6: **Result for the example**

Face ID	1	2	3	4	5	6	7	8	Group
Ground Truth Label	1	2	3	2	3	2	3	2	2
Rounded Predicted	1.0	2.0	3.0	1.0	2.0	3.0	3.0	2.0	2.0
Predicted Label	1.44	2.11	3.03	1.36	2.33	2.94	3.07	2.18	2.25

system. The group image in the example is taken from the validation set of HAPPEI database. The group in the image consists both females and males, the young and the old, the long hair and short hair, the one with glasses and the one without glasses, as shown in Figure 4.7. The result from Table 4.6 shows that our system generalizes well on diverse images.

FIGURE 4.7: **An example of emotion analysis on HAPPEI database**

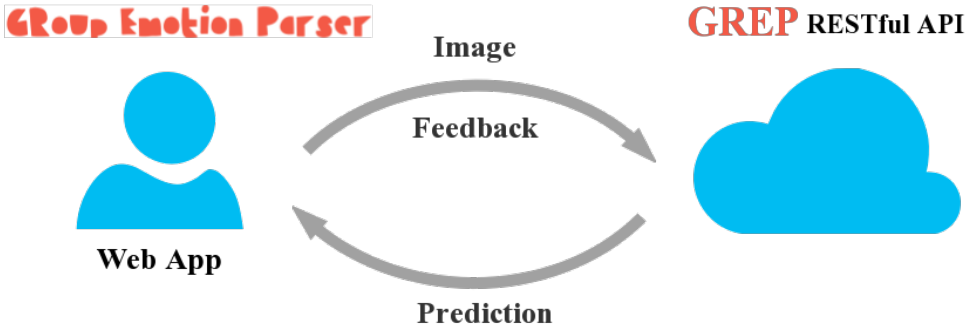


Chapter 5

Application

For demonstration and future research purpose, a set of RESTful APIs and an interactive web application are set, as shown in Figure 5.1 below. The backend is built using Flask in Python.

FIGURE 5.1: **RESTful APIs and web app**



5.1 RESTful API: GREP(v1.0.0)

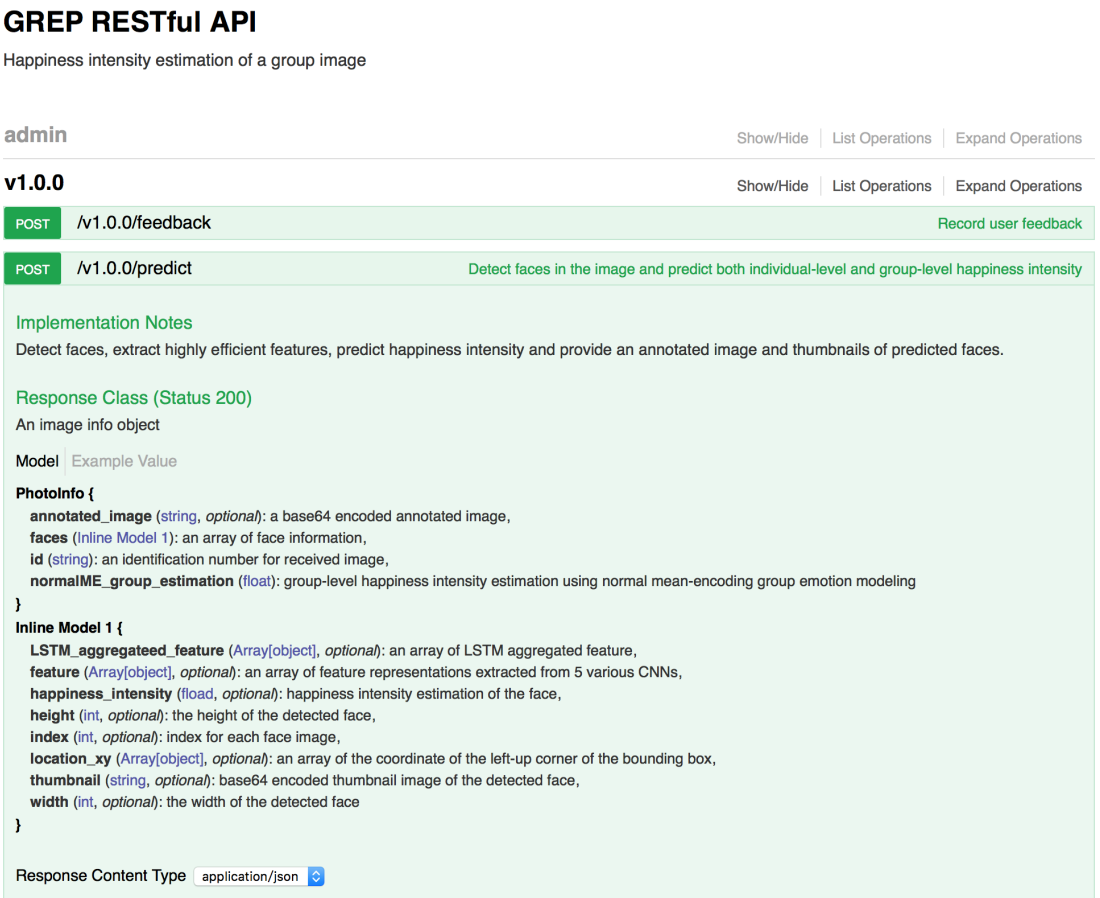
RESTful refers to representational state transfer (REST), which is a way of offering interoperability between computer systems on the Internet. Web service APIs that adhere to the REST architectural constraints are called RESTful APIs.

For this project, a set of RESTful APIs called GREP (GRoup Emotion Parser) that detects human faces within the image and estimates both individual-level and group-level happiness intensity is set and configured on a virtual server. Currently, GREP only offers two APIs:

1. POST /v1.0.0/predict
2. POST /v1.0.0/feedback

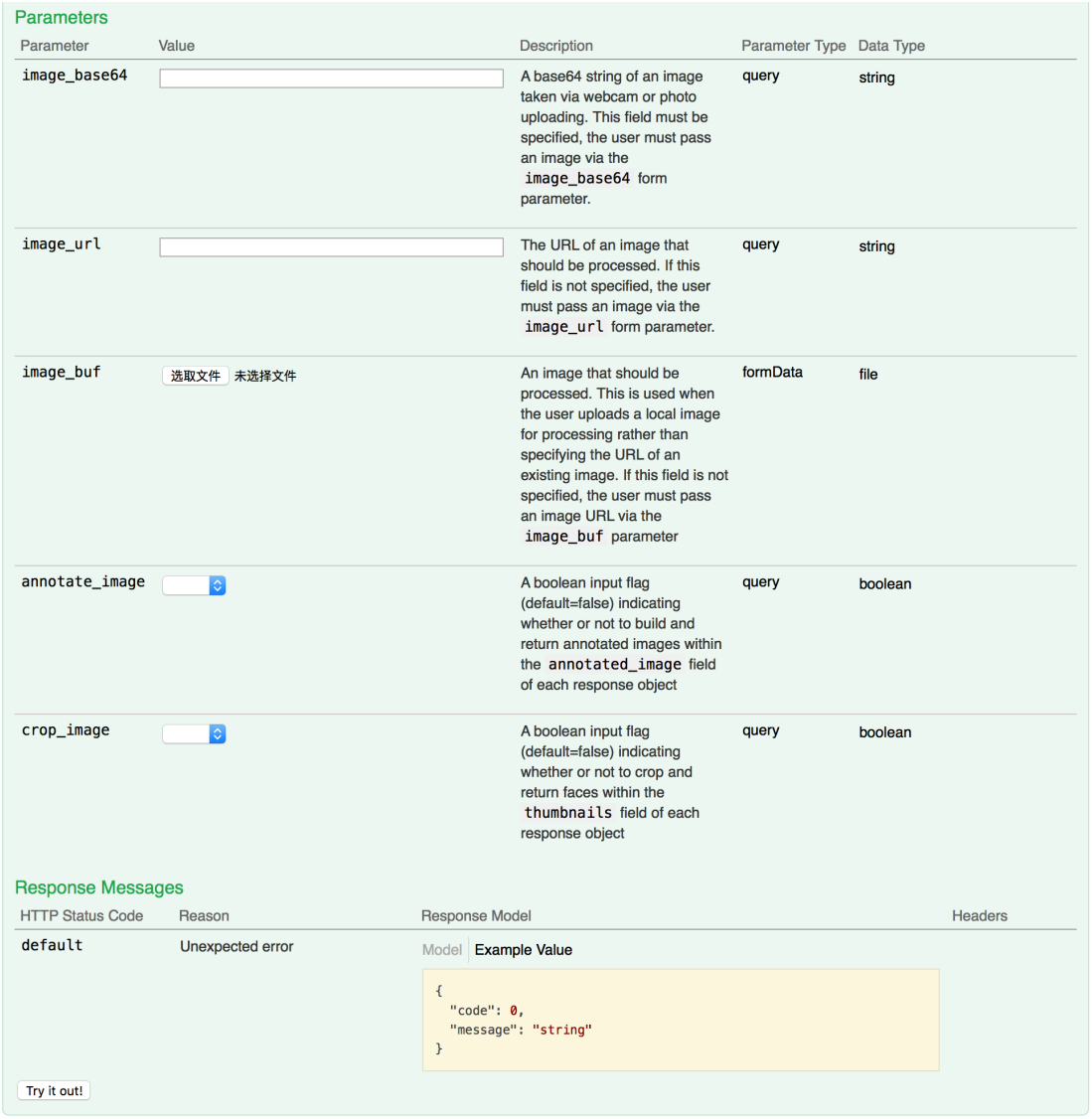
In order to get happiness estimation, the user needs to send a request with the url of the image or a base-64 format of the image. The user has the option to specify whether to return annotated faces and cropped thumbnail of each face in base64. Figure 5.2 and 5.3 shows the detailed request parameters and response. The base url for GREP API is: *https://grep.net.cn/v1.0.0/*.

FIGURE 5.2: **GREP(v1.0.0) POST /v1.0.0/predict response detail**



POST /v1.0.0/feedback allows for collecting user feedback from the web app for incorrect predictions.

FIGURE 5.3: **GREP(v1.0.0) POST /v1.0.0/predict request detail**



Parameters

Parameter	Value	Description	Parameter Type	Data Type
image_base64		A base64 string of an image taken via webcam or photo uploading. This field must be specified, the user must pass an image via the image_base64 form parameter.	query	string
image_url		The URL of an image that should be processed. If this field is not specified, the user must pass an image via the image_url form parameter.	query	string
image_buf	选取文件 未选择文件	An image that should be processed. This is used when the user uploads a local image for processing rather than specifying the URL of an existing image. If this field is not specified, the user must pass an image URL via the image_buf parameter	formData	file
annotate_image		A boolean input flag (default=false) indicating whether or not to build and return annotated images within the annotated_image field of each response object	query	boolean
crop_image		A boolean input flag (default=false) indicating whether or not to crop and return faces within the thumbnails field of each response object	query	boolean

Response Messages

HTTP Status Code	Reason	Response Model	Headers
default	Unexpected error	Model Example Value	

```
{
  "code": 0,
  "message": "string"
}
```

Try it out!

5.1.0.1 Documentation

A detailed and interactive documentation website is set using flask-swagger and Sawgger-UI. It can be accessed through the url: https://grep.net.cn/doc.

5.2 Interactive Web Application: GREP.net.cn

For demonstration purpose, an interactive web application GREP is built and running at https://grep.net.cn. It is an end-to-end system that detects faces in the group image and estimates happiness intensity on both individual level and group level. This app uses the RESTful API mentioned above to perform the estimation task. It allows the user to have multiple options to upload an image, including directly using url, selecting

a local image, or using the webcam to take the picture. After uploading the image, each detected face together with individual happiness estimation and group happiness estimation will show on the webpage. The app also provides the user with a way to give feedback, which will serve as a training sample and help improve the system in the future.

The web application is the so called single-page application (SPA). It is such a web application that fits on a single web page with the goal of providing a more fluid user experience similar to a desktop application.

The front-end of the web application is built upon Angular.js, which is an open-source web application framework, powerful for building SPA.

Chapter 6

Conclusion

To conclude, I first give an overall summary of the work presented in this thesis. I then point out some of the limitations of the current system and possible directions for future work.

6.1 Summary

This thesis proposes a new ensemble learning based method (RRDE) to estimate group-level happiness intensity. Particularly, in total there are n (5 is found to be the best) convolutional neural networks with different data selected using bootstrapping and these networks are used to extract n 64-dimensional vectors for each image. LSTM is used to scan these vectors to aggregate them. The output from LSTM is treated as the final individual face feature representation (128-dimensional vector). For both face-level and group-level estimation, two parameter-optimized SVRs are used for regression. To aggregate group-level feature, four different GEMs including mean-encoding, weighted mean-encoding, mean of face-level estimations and weighted mean of face-level estimations are tried and the best performance with RMSE equal to 0.55 on validation set of HAPPEI dataset is achieved when using weighted mean-encoding model with 5-network RRDE. This result is significantly better than the baseline of 0.78, which indicates the effectiveness of our proposed feature extraction framework, RRDE. Finally, for demonstration and future research purpose, a set of RESTful APIs with detailed documentation and a web application is built based on web techniques.

6.2 Limitations of RRDE and Future Work

First of all, as shown in Figure 4.3, CNNs tend to saturate very quickly as the ensemble size grows. With the number of CNNs increased to 5, there is already no more performance improvement in deep ensembles, which leaves RRDE with very few improvement space. This may result from the fact that although bootstrapping has already brought more diversity in final decisions, there lacks enough diversity in CNN structures. In the future works, one possible direction is to design various and effective objective functions in training individual deep CNNs in order to get more diverse in network structures. Additionally, CNNs in the ensemble could have more diverse architectures. For instance, different CNNs in the ensemble could be designed for particular purposes.

Secondly, in terms of group-level happiness intensity estimation task, the existing database (HAPPEI database) is an extremely unbalanced dataset, which may have compromised the generalization ability of the system when training directly on such skewed data. One necessary work for future research in this area is to extend the existing dataset to incorporate negative emotion group images to the database.

Thirdly, in our GEMs, although context information such as significance of faces is considered, it is not enough. A multi-modal framework combining face, upper-body and context information needs to be built in order to better predict on the group entity. Moreover, human body pose can be merged with the face analysis of a group of people. Based on recent work by [21], there is much useful emotion information embedded in body pose. Further, attributes such as background scene and clothes colors are also useful as they could give important information about the social event and, therefore, help estimate the group-level emotion. In the future, social context factors, such as age and gender, together with more complex top-down scene-related features, such as the CENTRIST descriptor, should also be explored.

Finally, the main contribution of this thesis is the proposition of a novel feature extraction framework RRDE. Is order to demonstrate the broad applicability and efficacy of our proposed RRDE framework, it is necessary to apply it to more tasks, rather than only experimenting on GEA task in our case. It may be more persuasive to test the method on tasks such as semantic segmentation, image captioning and image classification.

Bibliography

[1] Abhinav Dhall, Roland Goecke, and Tom Gedeon. Automatic group happiness intensity analysis. *IEEE Transactions on Affective Computing*, 6(1):13–26, 2015.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[3] Abhinav Dhall, Akshay Asthana, and Roland Goecke. Facial expression based automatic album creation. In *International Conference on Neural Information Processing*, pages 485–492. Springer, 2010.

[4] Thomas Vandal, Daniel McDuff, and Rana El Kaliouby. Event detection: Ultra large-scale clustering of facial expressions. In *Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on*, volume 1, pages 1–8. IEEE, 2015.

[5] Charles Darwin, Paul Ekman, and Phillip Prodger. *The expression of the emotions in man and animals*. Oxford University Press, USA, 1998.

[6] Ciprian A Corneanu, Marc Oliu, Jeffrey F Cohn, and Sergio Escalera. Survey on rgb, 3d, thermal, and multimodal approaches for facial expression recognition: history, trends, and affect-related applications. 2016.

[7] Evangelos Sariyanidi, Hatice Gunes, and Andrea Cavallaro. Automatic analysis of facial affect: A survey of registration, representation, and recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(6):1113–1133, 2015.

[8] Abhinav Dhall, OV Ramana Murthy, Roland Goecke, Jyoti Joshi, and Tom Gedeon. Video and image based emotion recognition challenges in the wild: Emotiw 2015. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 423–426. ACM, 2015.

[9] Abhinav Dhall, Roland Goecke, Jyoti Joshi, Jesse Hoey, and Tom Gedeon. Emotiw 2016: Video and group-level emotion recognition challenges. In *Proceedings of*

the 18th ACM International Conference on Multimodal Interaction, pages 427–432. ACM, 2016.

[10] Ping Liu, Shizhong Han, Zibo Meng, and Yan Tong. Facial expression recognition via a boosted deep belief network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1805–1812, 2014.

[11] Hong-Wei Ng, Viet Dung Nguyen, Vassilios Vonikakis, and Stefan Winkler. Deep learning for emotion recognition on small datasets using transfer learning. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 443–449. ACM, 2015.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[13] Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3642–3649. IEEE, 2012.

[14] Bo-Kyeong Kim, Jihyeon Roh, Suh-Yeon Dong, and Soo-Young Lee. Hierarchical committee of deep convolutional neural networks for robust facial expression recognition. *Journal on Multimodal User Interfaces*, pages 1–17, 2016.

[15] Samira Ebrahimi Kahou, Christopher Pal, Xavier Bouthillier, Pierre Froumenty, Çaglar Gülçehre, Roland Memisevic, Pascal Vincent, Aaron Courville, Yoshua Bengio, Raul Chandias Ferrari, et al. Combining modality specific deep neural networks for emotion recognition in video. In *Proceedings of the 15th ACM on International conference on multimodal interaction*, pages 543–550. ACM, 2013.

[16] Zhiding Yu and Cha Zhang. Image based static facial expression recognition with multiple deep network learning. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 435–442. ACM, 2015.

[17] Ali Mollahosseini, David Chan, and Mohammad H Mahoor. Going deeper in facial expression recognition using deep neural networks. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–10. IEEE, 2016.

[18] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[19] Xiangyun Zhao, Xiaodan Liang, Luoqi Liu, Teng Li, Yugang Han, Nuno Vasconcelos, and Shuicheng Yan. Peak-piloted deep network for facial expression recognition. In *European Conference on Computer Vision*, pages 425–442. Springer, 2016.

[20] Hatice Gunes and Massimo Piccardi. Bi-modal emotion recognition from expressive face and body gestures. *Journal of Network and Computer Applications*, 30(4): 1334–1345, 2007.

[21] Andrea Kleinsmith and Nadia Bianchi-Berthouze. Affective body expression perception and recognition: A survey. *IEEE Transactions on Affective Computing*, 4 (1):15–33, 2013.

[22] Aggeliki Vlachostergiou, George Caridakis, and Stefanos Kollias. Context in affective multiparty and multimodal interaction: Why, which, how and where? In *Proceedings of the 2014 workshop on Understanding and Modeling Multiparty, Multimodal Interactions*, pages 3–8. ACM, 2014.

[23] Louis-Philippe Morency. The role of context in affective behavior understanding. *Social Emotions in Nature and Artifact*, page 128, 2013.

[24] Sigal G Barsade and Donald E Gibson. Group emotion: A view from top and bottom. *Research on managing groups and teams*, 1(4):81–102, 1998.

[25] Janice R Kelly and Sigal G Barsade. Mood and emotions in small groups and work teams. *Organizational behavior and human decision processes*, 86(1):99–130, 2001.

[26] Abhinav Dhall, Jyoti Joshi, Karan Sikka, Roland Goecke, and Nicu Sebe. The more the merrier: Analysing the affect of a group of people in images. In *Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on*, volume 1, pages 1–8. IEEE, 2015.

[27] Xiaohua Huang, Abhinav Dhall, Xin Liu, Guoying Zhao, Jingang Shi, Roland Goecke, and Matti Pietikainen. Analyzing the affect of a group of people using multi-modal framework. *arXiv preprint arXiv:1610.03640*, 2016.

[28] Jianxin Wu and Jim M Rehg. Centrist: A visual descriptor for scene categorization. *IEEE transactions on pattern analysis and machine intelligence*, 33(8):1489–1501, 2011.

[29] Jianshu Li, Sujoy Roy, Jiashi Feng, and Terence Sim. Happiness level prediction with sequential inputs via multiple regressions. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pages 487–493. ACM, 2016.

[30] Vassilios Vonikakis, Yasin Yazici, Viet Dung Nguyen, and Stefan Winkler. Group happiness assessment using geometric features and dataset balancing. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pages 479–486. ACM, 2016.

[31] Bo Sun, Qinglan Wei, Liandong Li, Qihua Xu, Jun He, and Lejun Yu. Lstm for dynamic emotion and group emotion recognition in the wild. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pages 451–457. ACM, 2016.

[32] Abhinav Dhall, Roland Goecke, Simon Lucey, and Tom Gedeon. Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 2106–2112. IEEE, 2011.

[33] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1):157–173, 2008.

[34] Christopher M Bishop. Pattern recognition. *Machine Learning*, 128:1–58, 2006.

[35] Bekir Karlik and A Vehbi Olgac. Performance analysis of various activation functions in generalized mlp architectures of neural networks. *International Journal of Artificial Intelligence and Expert Systems*, 1(4):111–122, 2011.

[36] Tao Wang, David J Wu, Adam Coates, and Andrew Y Ng. End-to-end text recognition with convolutional neural networks. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 3304–3308. IEEE, 2012.

[37] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[38] Dan C Cireşan, Ueli Meier, Jonathan Masci, Luca M Gambardella, and Jürgen Schmidhuber. High-performance neural networks for visual object classification. *arXiv preprint arXiv:1102.0183*, 2011.

[39] Christopher Olah. Understanding lstm networks, aug 2015. URL http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[40] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[41] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[42] Michael Kearns and Leslie Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *Journal of the ACM (JACM)*, 41(1):67–95, 1994.

[43] David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

[44] ZongYuan Ge, Alex Bewley, Christopher McCool, Peter Corke, Ben Upcroft, and Conrad Sanderson. Fine-grained classification via mixture of deep convolutional neural networks. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pages 1–6. IEEE, 2016.

[45] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[46] Karim Ahmed, Mohammad Haris Baig, and Lorenzo Torresani. Network of experts for large-scale image categorization. In *European Conference on Computer Vision*, pages 516–532. Springer, 2016.

[47] Stefan Lee, Senthil Purushwalkam Shiva Prakash, Michael Cogswell, Viresh Ranjan, David Crandall, and Dhruv Batra. Stochastic multiple choice learning for training diverse deep ensembles. In *Advances in Neural Information Processing Systems*, pages 2119–2127, 2016.

[48] Abner Guzman-Rivera, Dhruv Batra, and Pushmeet Kohli. Multiple choice learning: Learning to produce multiple structured outputs. In *Advances in Neural Information Processing Systems*, pages 1799–1807, 2012.

[49] Abner Guzman-Rivera, Pushmeet Kohli, Dhruv Batra, and Rob A Rutenbar. Efficiently enforcing diversity in multi-output structured prediction. In *AISTATS*, volume 2, page 3, 2014.

[50] Xiangxin Zhu and Deva Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2879–2886. IEEE, 2012.

[51] Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[52] Xuehan Xiong and Fernando De la Torre. Supervised descent method and its applications to face alignment. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 532–539, 2013.

[53] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[54] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pages 249–256, 2010.

[55] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

[56] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[57] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[58] Kaiming He and Jian Sun. Convolutional neural networks at constrained time cost. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5353–5360, 2015.

[59] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

[60] Christopher M Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995.

[61] William N Venables and Brian D Ripley. *Modern applied statistics with S-PLUS*. Springer Science & Business Media, 2013.

[62] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[63] Hasim Sak, Andrew W Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Interspeech*, pages 338–342, 2014.

[64] Hilary L Seal. *The historical development of the Gauss linear model*. Yale University, 1968.

[65] Kenneth L Lange, Roderick JA Little, and Jeremy MG Taylor. Robust statistical modeling using the t distribution. *Journal of the American Statistical Association*, 84(408):881–896, 1989.

[66] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

[67] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

[68] John Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. 1998.

[69] Rong-En Fan, Pai-Hsuen Chen, and Chih-Jen Lin. Working set selection using second order information for training support vector machines. *Journal of machine learning research*, 6(Dec):1889–1918, 2005.

[70] Pai-Hsuen Chen, Rong-En Fan, and Chih-Jen Lin. A study on smo-type decomposition methods for support vector machines. *IEEE Trans. Neural Networks*, 17(4): 893–908, 2006.

[71] Guodong Guo and Guowang Mu. Simultaneous dimensionality reduction and human age estimation via kernel partial least squares regression. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 657–664. IEEE, 2011.

[72] William Robson Schwartz, Aniruddha Kembhavi, David Harwood, and Larry S Davis. Human detection using partial least squares analysis. In *Computer vision, 2009 IEEE 12th international conference on*, pages 24–31. IEEE, 2009.

[73] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.