A Deep Learning Perspective on the Origin of Facial Expressions

Ran Breuer
A Deep Learning Perspective on the Origin of Facial Expressions

Research Thesis

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

Ran Breuer

Submitted to the Senate of the Technion — Israel Institute of Technology Iyar 5777 Haifa May 2017
This research was carried out under the supervision of Prof. Ron Kimmel, in the Computer Science Department.

Some results in this thesis have been published as articles by the author and research collaborators in conferences and journals during the course of the author’s doctoral research period, the most up-to-date versions of which being:


**Acknowledgements**

I would like to thank, first of all, to my advisor, Prof Ronny Kimmel, for his patience and guidance throughout the entire research and my studies. I couldn’t have asked for a better guide.

I would also like to thank those who helped me along the way, fellow students and friends. Lastly, I’d like to thank my family for their support, and most of all to my wife, Nitzan, who stood up with the long nights and stressful times that came every once in a while.

Thank you.

The Technion’s funding of this research is hereby acknowledged.
# Contents

## List of Figures

Abstract

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
</tr>
<tr>
<td>1.1</td>
<td>Thesis Organization</td>
</tr>
<tr>
<td>1.2</td>
<td>Facial Expression Analysis</td>
</tr>
<tr>
<td>1.2.1</td>
<td>FACS</td>
</tr>
<tr>
<td>1.2.2</td>
<td>Facial Expression Recognition and Analysis</td>
</tr>
<tr>
<td>2</td>
<td>Preliminaries</td>
</tr>
<tr>
<td>2.1</td>
<td>Deep Learning</td>
</tr>
<tr>
<td>2.2</td>
<td>Convolutional Neural Networks</td>
</tr>
<tr>
<td>2.3</td>
<td>Training CNN Models</td>
</tr>
<tr>
<td>2.4</td>
<td>Visualizing and Understanding CNNs</td>
</tr>
<tr>
<td>3</td>
<td>Experiments</td>
</tr>
<tr>
<td>3.1</td>
<td>Introduction</td>
</tr>
<tr>
<td>3.2</td>
<td>Datasets</td>
</tr>
<tr>
<td>3.3</td>
<td>Network Architecture</td>
</tr>
<tr>
<td>4</td>
<td>Results</td>
</tr>
<tr>
<td>4.1</td>
<td>Implementation</td>
</tr>
<tr>
<td>4.2</td>
<td>Visualizing the CNN Filters</td>
</tr>
<tr>
<td>4.3</td>
<td>Correlating CNN features to FACS</td>
</tr>
<tr>
<td>5</td>
<td>Applications</td>
</tr>
<tr>
<td>5.1</td>
<td>Transfer Learning</td>
</tr>
<tr>
<td>5.1.1</td>
<td>Cross-Dataset Learning</td>
</tr>
<tr>
<td>5.1.2</td>
<td>Cross-Task Performance</td>
</tr>
<tr>
<td>5.2</td>
<td>Micro-Expression Detection</td>
</tr>
<tr>
<td>5.2.1</td>
<td>CASME II Dataset</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Micro-Expression Detection</td>
</tr>
</tbody>
</table>
6 Conclusions

A Appendix

A.1 CNN Feature Visualizations ................................. 41
A.2 Feature Correlation Tests ................................. 46

Hebrew Abstract
## List of Figures

1.1 Expressive images and their active AU coding. This demonstrates the composition of describing one’s facial expression using a collection of FACS based descriptors. ........................................ 3

1.2 Images from Duchenne de Boulogne’s experiments. Facial expressions are obtained by stimulating facial muscles with electricity [19] ........ 4

1.3 Example of primary universal emotions. From left to right: disgust, fear, happiness, surprise, sadness, and anger.1 ................................. 5

1.4 FACS Action Units (AUs). Each AU corresponds to a group of facial muscles that contract together to form this movement. [21] ........... 6

1.5 FACS Action Unit intensity scale. [21] ................................. 6

1.6 Examples of identical AU notations performed by different subjects [51] 7

2.1 An example of a 2D convolution over a 2D tensor. In this case output is restricted to only positions where the kernel lies completely within the image, often referred to as “valid” convolution. [30] ......................... 11

2.2 The rectified linear unit (ReLU) activation function .................... 12

2.3 LeNet-5 architecture, as designed by LeCun in 1998 [40]. ........... 12

2.4 Numerically generated images, illustrating the class model appearance as learnt by ImageNet [54]. Different aspects of the class in question are captured in a single image. ........................................ 15

2.5 Class saliency maps for selected classes in ILSVRC-2013 challenge. Maps are extracted using a single backpropagation pass through AlexNet classifier. [39, 54] ........................................ 16

2.6 Top: A deconvolutional layer (left) is attached to a convolutional layer (right). The deconvolutional layer approximates a reconstruction of the input to the convolutional layer based on the output on the convolution. Bottom: A demonstration of the unpooling operation by using switches to record the location of the local maxima. Switches are illustrated in the middle (black and white). [66] ................................. 17
2.7 Visualization of features on ImageNet [39]. Each feature is represented by the top 9 activations from the validation data. Each region is reconstructed back to pixel space using the deconvolutional network approach. These reconstructed patterns are those responsible for high activations in the given feature map. Each reconstruction is displayed alongside the original image region. One can observe a few principles: (i) there is a strong grouping within each feature map, (ii) the deeper the layer is - the higher the invariance and (iii) the deeper the layer is - the more complex and abstract the concepts it learns. [66]...

2.8 Different ways of propagating back through a ReLU nonlinearity. For each approach there is a formal definition (left) and schematic (right) for how a neuron output activation propagates back through a ReLU. [56]...

2.9 Visualization difference of the same image region with different reconstruction methods. [56]...

2.10 Visualization of different feature maps when given the same input (middle). This shows that while one neuron is searching for “cat” (left), the other is looking for “dog” (right). [56]...

3.1 Demonstration of the filter visualization process.

3.2 Images from CK+ (top), NovaEmotions (middle) and FER2013 (bottom) datasets.

4.1 Feature visualization for the trained network model. For each feature we overlay the deconvolution output on top of its original input image. One can easily see the regions to which each feature refers. Visualizations are done on CK+ dataset.

4.2 Feature visualization for the trained network model. Visualizations are done on the FER2013 dataset.

4.3 Correlation results on CK+. For each neuron (filter), the best scoring action units are described. Correlation score is computed as described in 4.1.

4.4 Several feature maps and their corresponding FACS Action Unit.

5.1 Demonstration of the evolution of a micro-expression. The onset, apex and offset frames are at 40, 11 and 160 ms respectively. The emotion displayed in this sequence portrays disgust, which can be seen by the movement of AU4+9. The three highlighted rectangles above the frames focuses on the inner brow lowerer (AU4).

[63]...
A.1 Feature visualization for the trained network model. For each feature we overlay the deconvolution output on top of its original input image. One can easily see the regions to which each feature refers. Visualizations are from experiments on the CK+ dataset.

A.2 More Feature visualizations on CK+.

A.3 Feature visualization for the trained network model. Visualizations are from experiments on the FER2013 dataset.

A.4 More feature visualizations on FER2013.

A.5 More feature visualizations on FER2013.

A.6 Correlation results on CK+. For each neuron (filter), the best scoring action units are described. Correlation score is computed as described in 4.1.

A.7 Correlation results on CK+.

A.8 Correlation results on CK+.

A.9 Correlation results on CK+.
Abstract

Facial expressions play a significant role in human communication and behavior. Psychologists have long studied the relationship between facial expressions and emotions. Paul Ekman et al. [24, 21], devised the Facial Action Coding System (FACS) to taxonomize human facial expressions and model their behavior. FACS is an anatomically based system for describing all observable facial movements for each emotion. The ability to recognize facial expressions automatically, enables novel applications in fields like human-computer interaction, social gaming, and psychological research. There has been a tremendously active research in this field, with several recent papers utilizing convolutional neural networks (CNN) for feature extraction and inference.

We employ CNN visualization and understanding methods to study the relation between the features these computational networks are using, Ekman’s FACS and the Action Units (AU) that comprise it. We verify our findings on several datasets, among which the Extended Cohn-Kanade (CK+), NovaEmotions and FER2013 datasets. We apply these models to various tasks and tests using transfer learning (or knowledge transfer), including cross-dataset validation and cross-task performance. Finally, we exploit the nature of the FER based CNN models for the detection of micro-expressions and achieve state-of-the-art accuracy using a simple long-short-term-memory (LSTM) recurrent neural network (RNN).
Chapter 1

Introduction

Figure 1.1: Expressive images and their active AU coding. This demonstrates the composition of describing one’s facial expression using a collection of FACS based descriptors..

Human communication consists of much more than verbal elements, words and sentences. Facial expressions (FE) play a significant role in inter-person interaction. They convey emotional state, truthfulness and add context to the verbal channel. Automatic FE recognition (AFER) is an interdisciplinary domain standing at the crossing of behavioral science, psychology, neurology, and artificial intelligence.

1.1 Thesis Organization

The organization of this thesis is as follows. We start by reviewing the field of facial expression analysis in Section 1.2. We provide overview on Ekman’s Facial Action Coding System (FACS) in 1.2.1 and describe methods of automatic facial expression analysis in 1.2.2. We lay the basis for understanding convolutional neural networks (CNN) in Chapter 2. We next introduce this work’s main contribution. We start by describing the datasets and methods used in our experiments in Chapter 3. A review
of our results and analysis is presented in Chapter 4. Various applications of our experimental findings are explored in Chapter 5. We conclude our findings and their applications in Chapter 6 and discuss possible future directions.

1.2 Facial Expression Analysis

The analysis of human emotions through facial expressions is a major part in psychological research. Physiognomy assumed that one’s personality could be inferred from a their outer appearance, especially the face and eyes. This was refuted for lack of scientific support by many, among which Leonardo Da Vinci. In the 17th century, John Bulwer published the first consistent work in the English language on the muscular mechanism of facial expressions. In his book, *Pathomyotomia, or, Dissection of the Significant Muscles of the Affections of the Mind* [10], Bulwer focused on the sign language of the hearing impaired. A couple of centuries later, influenced by the creationist approach, Sir Charles Bell studied expressions for his work on sensory and motor control. Bell believed that facial expressions were endowed by the Creator solely for the purpose of communication. At that same time in France, Duchenne de Boulogne conducted studies on the production of facial expressions in humans [19]. He published images of facial expressions obtained by electrical stimulation of facial muscles, see Figure 1.2. Darwin’s work in the late 1800’s [16] placed human facial expressions within an evolutionary context. Darwin suggested that facial expressions are the residual actions of more complete behavioral responses to environmental challenges. When in disgust, constricting the nostrils served to reduce inhalation of noxious or harmful substances. Widening of the eyes in surprise increased the visual field to better see an unexpected stimulus.

![Figure 1.2: Images from Duchenne de Boulogne’s experiments. Facial expressions are obtained by stimulating facial muscles with electricity [19]](image-url)
Inspired by Darwin’s evolutionary basis for expressions, Ekman et al. [24] introduced their seminal study about facial expressions. They identified seven primary, universal expressions where universality related to the fact that these expressions remain the same across different cultures [22]. Ekman labeled them by their corresponding emotional states, that is, happiness, sadness, surprise, fear, disgust, anger, and contempt, see Figure 1.3. Due to its simplicity and claim for universality, the primary emotions hypothesis has been extensively exploited in cognitive computing.

![Figure 1.3: Example of primary universal emotions. From left to right: disgust, fear, happiness, surprise, sadness, and anger.](image)

1.2.1 FACS

In order to further investigate emotions and their corresponding facial expressions, Ekman devised the *facial action coding system* (FACS) [21]. FACS is an anatomically based system for describing all observable facial movements for each emotion, see Figure 1.1. FACS seeks to describe nearly all possible facial expressions in terms of anatomically-based actions. Facial expressions are coded in *action units* (AU). Each action unit describes a cluster of facial muscles that act together to form a specific movement, see Figure 1.4 for details. Using FACS as a methodological measuring system, one can describe any expression by the action units (AU) one activates and its activation intensity. FACS also provides the rules for visual detection of the AUs as well as their temporal segments (*onset, apex, offset, ordinal intensity*). According to Ekman, there are 44 facial AUs, describing actions such as “open mouth”, “squint eyes” etc., and 20 other AUs were added in a 2002 revision of the FACS manual [25], to account for head and eye movement.

Action unit intensity is scored on a non-linear scale, ranging from A (*trace*) to E (*Maximum*). The range for each measure on this scale varies. See Figure 1.5 for a graphical representation of the intensity scale. The intensity scale is, however, somewhat of a conceptual model as opposed to a well-defined one. The FACS manual defines heuristics of scoring action unit intensities, most often noted in terms of whether certain features are apparent or not.

---

Footnote 1: Images taken from [52] ©Jeffrey Cohn
Figure 1.4: FACS Action Units (AUs). Each AU corresponds to a group of facial muscles that contract together to form this movement. [21]

<table>
<thead>
<tr>
<th>Upper Face Action Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>AU 1</td>
</tr>
<tr>
<td>Inner Brow Raiser</td>
</tr>
<tr>
<td>*AU 41</td>
</tr>
<tr>
<td>Lid Droop</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Lower Face Action Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>AU 9</td>
</tr>
<tr>
<td>Nose Wrinkler</td>
</tr>
<tr>
<td>AU 15</td>
</tr>
<tr>
<td>Lip Corner Depressor</td>
</tr>
<tr>
<td>AU 23</td>
</tr>
<tr>
<td>Lip Tightener</td>
</tr>
</tbody>
</table>

Figure 1.5: FACS Action Unit intensity scale. [21]

Action units are defined in terms of the amount of muscular effort is invested with respect to one’s neutral pose. FACS is thus defined independently of a person’s facial structure and characteristics. As such, the same FACS coding can be made on individuals with completely different appearance. See Figure 1.6 for an illustration of
several examples for similar FACS coding performed by multiple subjects. Note how differences in physiognomy affects what a FACS coder would render identical.

From a mathematical standpoint, FACS’ action units can be addressed as a sparse over-complete basis that spans the space of the human facial expressions. By using FACS, one can describe any facial expressions as a weighted combination of AUs. There is however, no guarantee of linearity in defining action units with respect to the neutral pose.

Figure 1.6: Examples of identical AU notations performed by different subjects [51]

1.2.2 Facial Expression Recognition and Analysis

The ability to automatically recognize facial expressions and infer the emotional state has a wide range of applications. These included emotionally and socially aware systems [20, 61, 17], improved gaming experience [9], driver drowsiness detection [62], and detecting pain in patients [45] as well as distress [36]. Recent advances have even integrated automatic analysis of viewers’ reaction for the effectiveness of advertisements [1, 2, 3].

Various methods have been used for automatic facial expression recognition (FER or AFER) tasks. Methods are usually comprised of a feature extraction step followed by a
classification of regression one. Extracted features can be divided into *predesigned* and *learned*. Learned features are learned automatically from the training data, whether supervised or unsupervised, whereas Predesigned features are hand-crafted to extract relevant information.

*Predesigned features* can be further divided into appearance and geometrical. Early papers used geometric representations, that is, describing faces through distances and shapes. These methods usually describe facial attributes as shapes or region, or facial deformation based on the location of specific fiducial points. Among such methods are vectors descriptors for the motion of the face [13], active contours for mouth and eye shape retrieval [8], and using 2D deformable mesh models [38]. The latter uses deformation parameters of a template mesh model to describe facial deformation. Others used appearance representation based methods. Appearance features use the intensity information of an image or video frame. Some examples of appearance representation based methods are Gabor filters [41], or local binary patterns (LBP) [53].

These feature extraction methods usually were combined with one of several regressors to translate these feature vectors to emotion classification or action unit detection. The most popular regressors used in this context were support vector machines (SVM) and random forests. For further reading on the methods used in FER, we refer the reader to [46, 14, 68, 65].

Over the last part of this past decade, *convolutional neural networks* (CNN) [40] and *deep belief networks* (DBN) have been used for feature extraction, classification and recognition tasks. Convolutional neural networks are used to jointly perform feature extraction and recognition. These CNNs have achieved state-of-the-art results in various fields, including object recognition [39], face recognition [57], and scene understanding [70]. Leading challenges in FER [18, 60, 59] have also been led by methods using CNNs [32, 12, 28].
Chapter 2

Preliminaries

2.1 Deep Learning

The early days of artificial intelligence (AI) aimed to solve and tackle problems that are difficult intellectually to humans but can be described in a straightforward fashion for computers. These problems can be explained through formal, mathematically described sets of rules. The true challenge of artificial intelligence proved to be solving tasks that seem easy and intuitive for people to perform but are hard to describe formally. Problems that we solve in an intuitive manner, somewhat automatically, such as reading handwritten text, recognizing spoken words or inferring objects in images.

A solution for the above problems is to allow computers to learn from experience and understand the world in terms of hierarchies of concepts, where each concept is defined by its derivation from simpler, more primitive concepts. By letting a computer learn from experience, we avoid the need to formulate all the prior knowledge one would require to complete the task at hand. The hierarchy of concepts constructed by the computer, allows it to learn more complicated concepts by inferring them from simpler ones. If this structure were to be visualized as a graph, each layer of concepts would be stacked on top of a simpler concept layer, the graph can be very deep. For this reason, we refer to this approach as Deep Learning [30].

The earliest predecessors of modern deep learning were simple linear models motivated from a neuroscientific perspective. The McCulloch-Pitts Neuron [5] was an early model of brain function. Rosenblatt’s Perceptron [49] became the first model able to learn the weights defining the classification when given examples from each category. The neurons used today in deep learning is a generalization of the original perception with a few variations. The inputs and output of a neuron are of a continuous nature, as opposed to the original binary form. Second, replacing the step function, other non-linear functions are applied over the output. The modern neuron usually takes the following formulation

\[ y = \varphi (Wx + b) \]  

(2.1)
where \( x \) and \( y \) are the input and output vectors respectively. \( \varphi \) is the non-linearity or activation function, \( W \) is the weights matrix and \( b \) is the neuron’s bias.

Neuroscience has inspired the field of artificial neural networks greatly. The idea of having many computational units that become intelligent through connections and interactions with each other is based on the neural structure in the brain. Most neural networks today are based on a model neuron called the rectified linear unit [15]. The Neocognitron [27] introduced a powerful model architecture for processing images greatly inspired by the mammal visual system and its structure. This model later became the basis for the modern convolutional network [40].

2.2 Convolutional Neural Networks

The modern structure of convolutional neural networks is a biologically inspired variant of the multilayer perceptron. From Huben and Wiesel’s early work on the cat’s visual cortex [35], we know that the visual cortex contains a complex arrangement of cells. Each cell is sensitive to small sub-regions of the visual field, called a receptive field. The sub-regions are tiled to cover the entire visual field. These cells act as local filters over the input space and are well-suited to exploit the strong spatially local correlation present in natural images. Additionally, two basic cell types have been identified. Simple cells respond to specific edge-like patterns within their respective receptive field. Complex cells have larger receptive fields and are locally invariant to the exact location of the pattern. Since the animal visual cortex is considered the most powerful visual processing system, it seems only natural to emulate its structure and behavior.

Convolutional neural networks, as first proposed by LeCun in 1998 [40], are a specialized kind of neural network for processing data that has a known, grid-like topology. Examples include time-series or audio, which can be modeled on a 1D grid taking samples at regular time intervals, and image data, which can be thought of as a 2D grid of pixels. The name implies that the networks employs a mathematical operation called convolution. Convolutional neural networks are simply neural networks that use convolutions in at least one of their layers, replacing the general matrix multiplication in 2.1. A convolution is a specialized kind of linear operation.

\[
Y = \sum_{n=0}^{N-1} W_n \ast X_n + b
\]  

(2.2)

Where \( X \in R^{N\times W \times H} \) is the input, with \( N \) channels and grids of width of \( W \) and height of \( H \) pixels. \( W \in R^{N \times K \times K} \) is the convolution filter collection with filters of size \( K \times K \). In convolutional network terminology, \( W_n \) is often referred to as the convolution kernel. \( Y \), the output is sometimes referred to as the feature map. Each feature map detects the presence of a single feature at all possible input locations. A convolution
over a 2D image $I$ with kernel $K$ is represented by

$$Y(i,j) = (I \ast K)(i,j) = \sum_{m} \sum_{n} I(i - m, j - n)K(m,n)$$  \hspace{1cm} (2.3)

See Figure 2.1 for an example of convolution applied to a 2D tensor or image.

A convolutional neural network (CNN) is usually comprised of the following steps, or layers. A convolutional layer which performs several convolutions in parallel to produce a set of linear activations. These layers are defined by the number of kernels applied on the input and the kernel sizes. Next, each linear activation is run through a nonlinear function, usually referred to as activation. Common activation functions are sigmoid, tanh and the rectified linear unit (ReLU). A ReLU function applies the following function to the output

$$\varphi(x) = \max(0, x)$$  \hspace{1cm} (2.4)

Figure 2.2 shows a graph for the ReLU activation function. A third stage of a convolutional network is optional but used very often, that is a down-sampling or pooling. A pooling function replaces the output of a net (or layer) at a certain location with a
Figure 2.2: The rectified linear unit (ReLU) activation function

summary statistic of nearby outputs. For example, max pooling reports the maximum output with a rectangular neighborhood. Other popular pooling functions include the average pooling, the $L_2$ norm of a neighborhood or a weighted average based on the distance from the central pixel. In all cases, pooling helps the representation become approximately invariant to small translation of the input.

The design of a suitable, high performant architecture or structure is one of the most important and challenging aspects of convolutional networks. Among the earlier notable designs for CNNs was LeNet-5 [40], was developed for handwritten digit recognition. The network architecture is shown in Figure 2.2. A gray-level image of $32 \times 32$ pixels is given as input to the network. 2 Convolutional layers are applied to the input, with kernel pools of sizes 6 and 16 respectively. Kernel sizes in each convolutional layer are $28 \times 28$ and $10 \times 10$. After each convolution layer, a tanh activation function is applied as well as a $2 \times 2$ max-pooling operation. The last convolution output is then flattened and connected to a fully connected “classical” layer comprised of 120 neurons. Output is passed through a hidden layer of 84 neurons and finally an output vector of size 10 is received. While considered shallow in present day measures, the LeNet structure is the inspiration for many of today’s leading models. Among the common models in the most recent surge of neural networks are AlexNet [39], VGG [55] and ResNet [33].

Figure 2.3: LeNet-5 architecture, as designed by LeCun in 1998 [40].
2.3 Training CNN Models

While neural networks are undoubtedly powerful, training such a model has proved to be a challenging task. The cornerstone of neural network optimization is the backpropagation algorithm [50] devised by Hinton et al. in 1988. Backpropagation is a method of training an artificial neural network in conjunction with an optimization method such as gradient descent. The algorithm repeats a two phase cycle comprised of forward propagation and weights update. An input is presented to the network and is propagated forward through the network layers, until it reaches the output layer. The output is then compared to the desired output, using a cost, objective or loss function, and an error value is calculated for each of the neurons in the output layer. The error values are then propagated backwards, from the output all the way to the input layer. Each neuron receives an associated error value which roughly represents its contribution to the original output.

Backpropagation uses the error values to calculate the gradient of the loss function with respect to the weights in the network. During the second phase, the gradient is fed to the optimization method, which in turn uses it to update the weights, thus attempting to minimize the loss function. The importance of this process is that while the network is trained, the neurons in the intermediate layers organize themselves in such a way that different neurons learn to recognize different characteristics or patterns of the total input space.

A loss function is a function that maps values of multiple variables to a real number. In backpropagation, the loss function calculates the difference (or distance) between the expected output and the resulting output, once the input example has propagated through the network. Among common loss function is the euclidean distance

\[ E(x) = \frac{1}{2} \| y(x) - \hat{y}(x) \|^2 \]  

(2.5)

where \( y, \hat{y} \) are the actual and expected outputs for input \( x \) respectively.

Another common loss function is the cross-entropy function. The cross-entropy loss function is commonly used alongside the softmax classifier. A softmax classifier receives a scoring function over multiple classes, meaning that each class receives a probability of the input belonging it. It then interprets these scores as the unnormalized log probabilities for each class and replaces the normal hinge loss with the cross-entropy function.

\[ L_i = -\log \left( \frac{e^{f_{y_i}}}{\sum_j e^{f_j}} \right) \]  

(2.6)

where \( f_j \) is the score for class \( j \) in the scores vector \( f \). The function \( f_i(x) = \frac{e^{x_i}}{\sum_j e^{f_j}} \) is called the softmax function. It takes a vector of arbitrary real-value scores \( x \) and transforms it to a vector of values in the range \([0, 1]\) whose sum is 1. The cross-entropy,
in information theory, is defined as

\[ H(p, q) = -\sum_x p(x) \log q(x) \] (2.7)

where \( p \) is the “true” distribution and \( q \) is an estimated distribution. The softmax is hence minimizing the cross-entropy between the estimated class \( c \) probabilities, \( q_c = \frac{E_{I \sim c} f_c}{\sum_j E_{I \sim j} f_j} \) and the “true” distribution \( p_c \). \( p_c \) is a zero vector where only the \( c \)-th element is 1. It can be thought of as if all the probability mass is on the correct class \( c \).

### 2.4 Visualizing and Understanding CNNs

In the effort of improving CNN performance, researchers have developed methods of exploring and understanding the models learned by these methods. Erhan et al. [26] visualized deep models by finding an input image which maximized the neuron activity of interest by carrying out an optimization using gradient ascent in the image space. The problem of convolutional neural networks visualization was introduced by Zeiler et al. [67]. For convolutional layer visualization, they proposed a deconvolutional network. The DeconvNet architecture aims to approximately reconstruct the input of each layer from its output.

Simonyan et al. [54] addressed the visualization of deep image classification CNNs, such as ImageNet [39]. Simonyan harnessed the technique of generating an image that maximally activates a neuron for class model visualization. This visualization method consists of numerically generating an image which maximizes

\[ \arg \max_I S_c(I) - \lambda \| I \|_2^2 \] (2.8)

In 2.8, \( S_c(I) \) is the score of class \( c \), as computed by the classification (final) layer of a convolutional network for image \( I \). \( \lambda \) is a regularization parameter for an \( L_2 \) regularization term on \( I \). It should be noted that in their method, Simonyan et al. are using unnormalized class scores \( S_c \) rather than the class posteriors as returned by the softmax layer. The reason is that the maximization of the class posterior can be achieved by minimizing scores of other class rather than maximizing the score for the class in question, \( c \). Figure 2.4 shows the results of different class model visualizations as applied on ImageNet.

Simonyan further explores class models through class saliency extraction [54]. Class saliency extraction aims to compute the spatial support of a particular class. This means, finding the pixels that affect the class score the most. One can expect that such pixels correspond to the object (or feature) location in the image.

We start with a motivational example. Consider the linear score model for class \( c \)

\[ S_c(I) = w_c^T I + b_c \] (2.9)
where $I$ is the vectorized input image, and $w_c, b_c$ are the weight vector and bias of the model, respectively. In the linear case, it is easy to see that the importance of the pixels of $I$ to class $c$ are represented by the magnitude of corresponding weights in $w_c$. In the case of CNNs, the scoring function $S_c(I)$ is a non linear function of $I$, so we cannot apply the linear example directly. However, we can approximate $S_c(I)$ as a linear function in the neighborhood of an input image $I_0$ by using the first-order Taylor expansion

$$S_c(I) \approx \frac{\partial S_c}{\partial I} \bigg|_{I_0} \cdot I + b$$

(2.10)

Given an input image $I_0^{m \times n}$ and a class $c$, the class saliency map $M \in R^{m \times n}$ is computed as follows. Once the derivative $w = \frac{\partial S_c}{\partial I} \bigg|_{I_0}$ is computed, the saliency map is obtained by rearranging the elements of $w$. The saliency map $M$ is constructed such that $M_{i,j} = \left| w_{h(i,j)} \right|$ where $h(i,j)$ is the vector element index in $w$ corresponding the image pixel $(i,j)$. Figure 2.4 shows saliency maps for selected classes from the ILSVRC-2013 challenge. One can notice that high saliency is located where the object in question is in the image. Girshick et al. continued exploring saliency [29] through visualizations that identify patches within a dataset that are responsible for strong activations at higher layers in the model.
Zeiler and Fergus [66] utilized previous work to further investigate the input stimuli that excite individual feature maps at any layer in the convolutional model. By using a multi-layered Deconvolutional Network, as proposed in [67], they project the feature activations back to the input pixel space. The main idea is to visualize the input pixels that cause a certain neuron, like a filter from a convolutional layer, to maximize its output. To do so, a deconvolutional network is attached to each of its layers, as shown in Figure 2.4(top). This provides a continuous path back to the image domain. The process starts with a forward pass, where the input image is presented to the network and features are computed throughout the layers all the way to the desired visualized layer. To examine a specific neuron activation, all other activations in the layer are set to 0. The resulting feature maps are then passes as input to the attached deconvNet layer. Operations of unpooling, rectification and deconvolution or filtering take place in that order. This is then repeated through the deconvolution layers until the input pixel space is reached.

Unpooling is the operations of reversing the max-pooling operation in the forward
Figure 2.6: Top: A deconvolutional layer (left) is attached to a convolutional layer (right). The deconvolutional layer approximates a reconstruction of the input to the convolutional layer based on the output on the convolution. Bottom: A demonstration of the unpooling operation by using switches to record the location of the local maxima. Switches are illustrated in the middle (black and white). [66]

pass. Since max-pooling is non-invertible, an approximate inverse is obtained by using switches. Switches are a set of variables that are used to record the location (or index) of the local maxima that are used in the forward pass. See Figure 2.4(bottom) for an illustration of the unpooling operation.

Rectification is simply applying the ReLU (2.4) non-linearity function at each layer. Filtering, or deconvolution, is done using the learned filters of the attached convolutional layer. The approximate inverse convolution is obtained by using transposed versions of the layer's filters.

This procedure can be assimilated to the backpropagation of a single strong activation. Each reconstruction by the deconvNet of the n-th layer input $X_n$ can be shown to be either equivalent or similar to propagating the gradient of the visualized neuron activity $h$ with respect to $X_n$. In such, the deconvolution process simply corresponds to the gradient backpropagation through the convolutional network.

For the convolutional layer performing $X_{n+1} = X_n \ast K_n$, the gradient is computed
as
\[
\frac{\partial h}{\partial X_n} = \frac{\partial h}{\partial X_{n+1}} \cdot \hat{K}_n
\]  
(2.11)

where \(K_n, \hat{K}_n\) are the convolution kernel and transposed version, respectively. Thus, the deconvolution process is computed as \(R_n = R_{n+1} \cdot \hat{K}_n\). The rectification layer computes \(X_{n+1} = \max(0, X_n)\), which sub-gradient takes the form of
\[
\frac{\partial h}{\partial X_n} = \frac{\partial h}{\partial X_{n+1}} \mathbb{1}(X_n > 0)
\]  
(2.12)

where \(\mathbb{1}\) is an element-wise indicator function. In the deconvolutional network approach, a slightly different operation is performed \(R_n = R_{n+1} \mathbb{1}(R_{n+1} > 0)\) and the indicator function is computed on the output reconstruction \(R_{n+1}\) instead of the layer input \(X_n\). The max-pooling layer \(X_{n+1}(p) = \max_{q \in \Omega(p)} X_n(q)\), where \(\Omega(p)\) is the spatial neighborhood of \(p\) which is determined by the max-pooling layer. The sub-gradient is
\[
\frac{\partial h}{\partial X_n(p)} = \frac{\partial h}{\partial X_{n+1}(p)} \mathbb{1}(s = \arg \max_{q \in \Omega(p)} X_n(q))
\]  
(2.13)

Here, we use the max-pooling “switch” as the computation of \(\arg \max\).

Zeiler et al. found that while the first layers in the CNN model seemed to learn Gabor-like filters, the deeper layers were learning high level representations of the objects the network was trained to recognize. By finding the maximal activation for each neuron, and back-propagating through the deconvolution layers, one could actually view the patterns and locations that caused a specific neuron to react. Figure 2.4 shows the deconvolutions of different neurons in various layers. Each Neuron, or filter, is visualized through the 9 image regions that caused the maximal activation through the neuron during the forward pass. Each image region is then reconstructed through the deconvolution layers, resulting in a sparse image with only the pattern that initiated the activation visible. One can easily see that as the visualized neuron belongs to a deeper layer, the more complicated and abstract concept it represents. While in layers 1 and 2 the concepts represented are “line”, “corner” or “circle”, layers deeper such as layer 5 learns feature representations for “dog”, “eye” and “text”.

Further efforts to understand the features in the CNN model, were done by Springenberg et al. who devised guided back-propagation [56]. With some minor modifications to the deconvolutional network approach, they were able to produce more understandable outputs, which provided better insight into the model’s behavior. Springenberg et al. altered the previous deconvolution layer approach of [66] in that they combine two approaches when dealing with the inversion of the non-linearity step. During deconvolution, the gradient is solely computed based on the convolution output signal, ignoring the previous input. In case of the ReLU function, this amounts to setting to 0 certain elements based on the output gradient. In backpropagation, the ReLU is activated on the opposite side of the layer, the input signal. Guided backpropagation combines the
two approaches and in that it masks out values corresponding to negative values on both the input and output signals of the layer, See Figure 2.4. This prevent backward flow of negative gradients, corresponding to neurons which decrease the activation of the desired high level neuron we aim to visualize. With this simple modification, a striking difference in image quality is obtained in feature visualization, See Figure 2.4. Figure 2.4 shows how performing the visualization operation on different neurons while given the same input, can show what each filter is looking for, or what feature it represents.

With the ability to visualize single feature maps and neurons, one can see that the feature that are learned are far from random. They represent intuitive properties that describe the model and its classes and help discriminate one class from the other. [66] shows how one can use feature visualizations to identify problems with the model and obtain better results.
Figure 2.8: Different ways of propagating back through a ReLU nonlinearity. For each approach there is a formal definition (left) and schematic (right) for how a neuron output activation propagates back through a ReLU. [56]

Figure 2.9: Visualization difference of the same image region with different reconstruction methods. [56]
Figure 2.10: Visualization of different feature maps when given the same input (middle). This shows that while one neuron is searching for “cat” (left), the other is looking for “dog” (right). [56]
Chapter 3

Experiments

Figure 3.1: Demonstration of the filter visualization process.

3.1 Introduction

Our goal is to explore the knowledge (or models) as learned by state-of-the-art methods for facial expression recognition (FER). We use CNN-based methods on various datasets to get a sense of a common model structure, and study the relation of these models to Ekman’s FACS [21]. To inspect the learned models ability to generalize, we use the method of transfer learning [64] to see how these models perform on other datasets. We also measure the models’ ability to perform on other FER related tasks, ones which they were not explicitly trained for.

3.2 Datasets

In order to get a sense of the common properties of CNN-based state-of-the-art models in FER, we employ these methods on numerous datasets. Below are brief descriptions of datasets used in our experiments. See Figure 3.2 for examples.
Extended Cohn-Kanade

The *Extended Cohn-Kanade* dataset (CK+) \[44\], is comprised of video sequences describing the facial behavior of 210 adults. Participant ages range from 18 to 50. 69% are female, 91% Euro-American, 13% Afro-American, and 6% belong to other groups. The dataset is composed of 593 sequences from 123 subjects containing posed facial expressions. Another 107 sequences were added after the initial dataset was released. These sequences captured spontaneous expressions performed between formal sessions during the initial recordings, that is, non-posed facial expressions.

Data from the Cohn-Kanade dataset is labeled for emotional classes (of the 7 primary emotions by Ekman \[24\]) at peak frames. In addition, AU labeling was done by two certified FACS coders. Inter-coder agreement verification was performed for all released data.

NovaEmotions

*NovaEmotions* \[58, 47\], aim to represent facial expressions and emotional state as captured in a non-controlled environment. The data is collected in a crowd-sourcing manner, where subjects were put in front of a gaming device, which captured their response to scenes and challenges in the game itself. The game, in time, reacted to the player’s response as well. This allowed collecting spontaneous expressions from a large pool of variations.

The NovaEmotions dataset consists of over 42,000 images taken from 40 different people. Majority of the participants were college students with ages ranges between 18 and 25. Data presents a variety of poses and illumination. In this paper we use cropped images containing only the face regions. Images were aligned such that eyes are presented on the same horizontal line across all images in the dataset. Each frame was
annotated by multiple sources, both certified professionals as well as random individuals. A consensus was collected for the annotation of the frames, resulting in the final labeling.

**FER 2013**

The **FER 2013** challenge [31] was created using Google image search API with 184 emotion related keywords, like blissful, enraged. Keywords were combined with phrases for gender, age and ethnicity in order to obtain up to 600 different search queries. Image data was collected for the first 1000 images for each query. Collected images were passed through post-processing, that involved face region cropping and image alignment. Images were then grouped into the corresponding fine-grained emotion classes, rejecting wrongfully labeled frames and adjusting cropped regions. The resulting data contains nearly 36,000 images, divided into 8 classes (7 effective expressions and a neutral class), with each emotion class containing a few thousand images (disgust being the exception with only 547 frames).

### 3.3 Network Architecture

For all experiments described in this paper, we implemented a simple, classic feed-forward convolutional neural network. Each network is structured as follows. An input layer, receiving a gray-level or RGB image. The input is passed through 3 convolutional layer blocks, each block consists of a filter map layer, a non-linearity (or activation) and a max pooling layer. Our implementation is comprised of 3 convolutional blocks, each with a rectified linear unit (ReLU [15]) activation and a pooling layer with $2 \times 2$ pool size. The convolutional layers have filter maps with increasing filter (neuron) count the deeper the layer is, resulting in a 64, 128 and 256 filter map sizes, respectively. Each filter in our experiments supports $5 \times 5$ pixels.

The convolutional blocks are followed by a fully-connected layer with 512 hidden neurons. The hidden layer’s output is transferred to the output layer, which size is affected by the task in hand, 8 for emotion classification, and up to 50 for AU labeling. The output layer can vary in activation, for example, for classification tasks we prefer softmax.

To reduce over-fitting, we used dropout [15] layers. We apply the dropout after the last convolutional layer and between the fully-connected layers, with probabilities of 0.25 and 0.5 respectively. A dropout probability $p$ means that each neuron’s output is set to 0 with probability $p$. 

25
Chapter 4

Results

4.1 Implementation

We trained our network using ADAM [37] optimizer with a learning rate of $1e^{-3}$ and a decay rate of $1e^{-5}$. To maximize generalization of the model, we use methods of data augmentation. We use combinations of random flips and affine transforms, e.g. rotation, translation, scaling, sheer, on the graphics to generate synthetic data and enlarge the training set. Our implementation is based on the Keras [11] library with TensorFlow [7] back-end. We use OpenCV [6] for all image operations.

We verify the performance of our networks on the datasets mentioned in 3.2 using a 10-fold cross validation technique. For comparison, we use the frameworks of [42, 53, 31, 41]. We analyze the networks’ ability to classify facial expression graphics into the 7 primary emotions or as a neutral pose. Accuracy is measured as the average score of the 10-fold cross validation. Our model performs at state-of-the-art level when compared to the leading methods in AFER.

4.2 Visualizing the CNN Filters

After establishing a sound classification framework for emotions, we move to analyze the models that were learned by the suggested network. We employ Zeiler et al. and Springenberg’s [66, 56] methods for visualizing the filters trained by the proposed

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gabor+SVM [41]</td>
<td>89.8%</td>
</tr>
<tr>
<td>LBPSVM [53]</td>
<td>95.1%</td>
</tr>
<tr>
<td>AUDN [42]</td>
<td>93.70%</td>
</tr>
<tr>
<td>BDBN [43]</td>
<td>96.7%</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td><strong>98.62 % ± 0.11%</strong></td>
</tr>
</tbody>
</table>

Table 4.1: Accuracy evaluation of emotion classification on the CK+ dataset
### Table 4.2: Accuracy evaluation of emotion classification on the FER 2013 challenge.
Methods and scores are documented in [31, 4, 46]

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human Accuracy</td>
<td>68% ± 5%</td>
</tr>
<tr>
<td>RBM</td>
<td>71.162%</td>
</tr>
<tr>
<td>VGG CNN [55]</td>
<td>72.7%</td>
</tr>
<tr>
<td>ResNet CNN [33]</td>
<td>72.4%</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td><strong>72.1% ± 0.5%</strong></td>
</tr>
</tbody>
</table>

Figure 4.1: Feature visualization for the trained network model. For each feature we overlay the deconvolution output on top of its original input image. One can easily see the regions to which each feature refers. Visualizations are done on CK+ dataset.

Figure 4.4: Additional visualization results can be found in the appendix. Further investigation shows that these regions and motions have significant correlation to those used by Ekman to define the FACS Action Units.

### 4.3 Correlating CNN features to FACS

We matched a filter’s suspected AU representation with the actual CK+ AU labeling, using the following method.

1. Given a convolutional layer $l$ and filter $j$, the activation output is marked as $F_{l,j}$.
2. We extracted the top $N$ input graphics that maximized, $i = \arg \max F_{i,j}(i)$.

3. For each input $i$, the manually annotated AU labeling is $A_i^{44 \times 1}$. $A_{i,u}$ is 1 if AU $u$ is present in $i$.

4. The correlation of filter $j$ with AU $u$’s presence is $P_{j,u}$ and is defined by

$$P_{j,u} = \frac{\sum A_{i,u}}{N}.$$ (4.1)

Since we used a small $N$, we rejected correlations with $P_{j,u} < 1$. Out of 60 active neurons from a 256 filters map trained on CK+, only 7 were rejected. This shows an amazingly high correlation between a CNN-based model, trained with no prior knowledge, and Ekman’s facial action coding system (FACS). Figure 4.3 shows results of the above correlation test on CK+ trained filters. It can be easily noticed that there is a string correlation between at least one action unit per filter. More than one action unit can be seldom correlated to the same filter, this can be explained through the connection some action unit have among themselves, such as “Brow lowerer”(4) and “Nose wrinlker”(9). Another explanation is the appearance of these correlated action units in the training dataset, the latter being not large enough to cover all different discriminative compositions. Table 4.3 describes some of the filters computed by our experiments, with their respective AU that gave the highest correlation scores.

In addition, we found that even though some AU-inspired filters were created more than just once, a large amount of neurons in the highest layers were found “dead”, that is, they were not producing effective output for any input. The amount of active neurons in the last convolutional layer was about 30% of the feature map size (60 out of 256). The number of effective neurons is similar to the size of Ekman’s vocabulary of action units by which facial expressions can be identified.
Table 4.3: Correlation between selected filters trained on CK+ and the AU with the highest correlation scores.

<table>
<thead>
<tr>
<th>Filter</th>
<th>AU</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>Nose Wrinkler</td>
</tr>
<tr>
<td>17</td>
<td>24</td>
<td>Lip Pressor</td>
</tr>
<tr>
<td>22</td>
<td>20</td>
<td>Lip Strecher</td>
</tr>
<tr>
<td>23</td>
<td>5</td>
<td>Upper Lid Raiser</td>
</tr>
<tr>
<td>57</td>
<td>17</td>
<td>Chin Raiser</td>
</tr>
<tr>
<td>73</td>
<td>12,14</td>
<td>Lip Corner Puller, Dimpler</td>
</tr>
<tr>
<td>134</td>
<td>23</td>
<td>Lip Tightener</td>
</tr>
<tr>
<td>180</td>
<td>16</td>
<td>Lower Lip Depressor</td>
</tr>
<tr>
<td>182</td>
<td>15</td>
<td>Lip Corner Depressor</td>
</tr>
<tr>
<td>206</td>
<td>7</td>
<td>Lid Tightener</td>
</tr>
</tbody>
</table>
Figure 4.3: Correlation results on CK+. For each neuron (filter), the best scoring action units are described. Correlation score is computed as described in 4.1
Figure 4.4: Several feature maps and their corresponding FACS Action Unit.
Chapter 5

Applications

After computationally demonstrating the strong correlation between Ekman’s FACS and the model learned by the proposed computational neural network, we study the model’s ability to generalize and solve other problems related to expression recognition on various data sets. We use the transfer learning training methodology [64] and apply it to different tasks in Section 5.1. In addition, we exploit the features learned through emotion detection methods to the field of micro-expression detection in Section 5.2.

5.1 Transfer Learning

Transfer learning, or knowledge transfer, aims to use models that were pre-trained on different data for new tasks. Neural network models often require large training sets. However, in some scenarios the size of the training set is insufficient for proper training. Transfer learning allows using the convolutional layers as pre-trained feature extractors, with only the output layers being replaced or modified according to the task at hand. That is, the first layers are treated as pre-defined features, while the last layers, that define the task at hand, are adapted by learning based on the available training set.

We tested our models on both cross-dataset and cross-task capabilities.

5.1.1 Cross-Dataset Learning

When testing emotion detection capabilities across datasets, we found that the trained models had very high scores. This shows, once again, that the FACS-like features trained on one dataset can be applied almost directly to another, see Table 5.1.

5.1.2 Cross-Task Performance

In most FER related tasks, AU detection is done as a leave-one-out manner. Given an input (image or video) the system would predict the probability of a specific AU to be active. This method is proven to be more accurate than training against the detection of all AU activations at the same time, mostly due to the sizes of the training datasets.
<table>
<thead>
<tr>
<th></th>
<th>Train CK+</th>
<th>Test FER2013</th>
<th>Test NovaEmotions</th>
</tr>
</thead>
<tbody>
<tr>
<td>CK+</td>
<td>98.62%</td>
<td>69.3%</td>
<td>67.2%</td>
</tr>
<tr>
<td>FER2013</td>
<td>92.0%</td>
<td>72.1%</td>
<td>78.0%</td>
</tr>
<tr>
<td>NovaEmotions</td>
<td>93.75%</td>
<td>71.8%</td>
<td>81.3%</td>
</tr>
</tbody>
</table>

Table 5.1: Cross dataset application of emotion detection models.

<table>
<thead>
<tr>
<th>AU</th>
<th>Baseline [44]</th>
<th>CK+</th>
<th>FER 2013</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>94.1</td>
<td>97.2</td>
<td>95.4</td>
</tr>
<tr>
<td>2</td>
<td>97.1</td>
<td>98.1</td>
<td>96.9</td>
</tr>
<tr>
<td>4</td>
<td>85.9</td>
<td>92.0</td>
<td>91.7</td>
</tr>
<tr>
<td>5</td>
<td>95.1</td>
<td>96.4</td>
<td>95.0</td>
</tr>
<tr>
<td>6</td>
<td>91.7</td>
<td>94.3</td>
<td>92.1</td>
</tr>
<tr>
<td>7</td>
<td>78.4</td>
<td>83.5</td>
<td>80.0</td>
</tr>
<tr>
<td>9</td>
<td>97.7</td>
<td>99.4</td>
<td>96.8</td>
</tr>
<tr>
<td>11</td>
<td>72.5</td>
<td>69.8</td>
<td>61.7</td>
</tr>
<tr>
<td>12</td>
<td>91.0</td>
<td>96.2</td>
<td>93.1</td>
</tr>
<tr>
<td>15</td>
<td>79.6</td>
<td>88.0</td>
<td>81.8</td>
</tr>
<tr>
<td>17</td>
<td>84.4</td>
<td>89.2</td>
<td>85.9</td>
</tr>
<tr>
<td>25</td>
<td>97.1</td>
<td>98.4</td>
<td>98.1</td>
</tr>
<tr>
<td>26</td>
<td>75.0</td>
<td>87.7</td>
<td>80.3</td>
</tr>
<tr>
<td>27</td>
<td>99.7</td>
<td>99.4</td>
<td>98.9</td>
</tr>
<tr>
<td>Avg</td>
<td>90.0</td>
<td>94.1</td>
<td>92.5</td>
</tr>
</tbody>
</table>

Table 5.2: Accuracy (%) on single AU detection using transfer learning from models trained on CK+ dataset (middle column) and FER 2013 dataset (right). Scores are compared to the baseline as published in [44]. AU detection was measured against the CK+ test set labeled by professional FACS coders. Average score is a weighted average depending on the number of positive examples in the CK+ dataset.

When testing our models against detection of a single AU, we recorded high accuracy scores with most AUs, see Table 5.2. Some action units, like AU11: *nasolabial deepener*, were not predicted properly in some cases when using the suggested model. A better prediction model for these AUs would require a dedicated set of features that focus on the relevant region in the face, since they signify a minor facial movement.

The leave-one-out approach is commonly used since the training set is not large enough to train a classifier for all AUs simultaneously (all-against-all). In our case, predicting all AU activations simultaneously for a single image, requires a larger dataset than the one we used. Having trained our model to predict only eight classes, we verify our model on an all-against-all approach and obtained result that compete with the leave-one-out classifiers. In order to increase accuracy, we apply a sparsity inducing loss function on the output layer by combining both $L_2$ and $L_1$ terms. This resulted in a
sparse FACS coding of the input frame. When testing for binary representation, that is, only an active/nonactive prediction per AU, we recorded an accuracy rate of 97.54%. When predicting AU intensity, an integer of range 0 to 5, we recorded an accuracy rate of 96.1% with a mean square error (MSE) of 0.2045.

5.2 Micro-Expression Detection

Micro-expressions (ME) are a more spontaneous and subtle facial movements that happen involuntarily, thus revealing one’s genuine, underlying emotion [23]. These micro-expressions are comprised of the same facial movements that define FACS action units and differ in intensity. Micro-expressions tend to last up to 0.5 seconds, making detection a challenging task for an un-trained individual. Each micro-expression is broken down into 3 steps: onset, apex, and offset, describing the beginning, peek, and the end of the motion, respectively.

Similar to AFER, a significant effort was invested in the last years to train computers in order to automatically detect micro-expressions and emotions. Due to its low movement intensity, automatic detection of micro-expressions requires a temporal sequence, as opposed to a single frame. Moreover, since micro-expressions tend to last for just a short time and occur in a brief of a moment, a high speed camera is usually used for capturing the frames. Figure 5.2 shows a sample sequence displaying the progression of a micro-expression. Current state-of-the-art methods for automatic micro-expression detection and recognition are mostly based on a variation of local binary patterns (LBP) made for video called local binary patterns - three orthogonal planes (LBP-TOP). This method calculates the LBP feature on 3 planes, that are described as $XT, YT, XY$, where $X,Y$ are the spatial image coordinate domains and $T$ is the temporal domain. The neural network variation commonly used for sequence (or video) processing is recurrent neural network (RNN). A recurrent neural network is a network where inter-neuron connections form a directed cycle. This creates an internal state within the network which allows to process dynamic temporal behavior.

5.2.1 CASME II Dataset

We apply our FACS-like feature extractors to the task of automatically detecting micro-expressions. To that end, we use the CASME II dataset [63]. CASME II includes 256 spontaneous micro-expressions filmed at 200fps. Sequences were captured with 35 subjects. The average age of the participants is 22. Expressions were captured by showing a subject video segments (TV commercials or short clips) that triggered the desired response. All videos are tagged for onset, apex, and offset times, as well as the expression conveyed. AU coding was added for the apex frame by professional FACS coders. Cross coder correlation scores were computed and only strongly correlated AU codes were used for the dataset. After establishing a sound FACS based encoding,
Figure 5.1: Demonstration of the evolution of a micro-expression. The onset, apex and offset frames are at 40, 11, and 160 ms respectively. The emotion displayed in this sequence portrays *disgust*, which can be seen by the movement of AU4+9. The three highlighted rectangles above the frames focuses on the inner brow lowerer (AU4). [63]

the emotions labeling previously done for the apex stage were verified against the AU codes. For example, AU4 (Brow lowerer) exists in a frown which can indicate disgust, anger or just plain attention to detail. Some subjects were able to repress the spontaneous emotion fast enough so that it couldn’t have been recorded even at 200fps, these sequences are marked as *repressed*.

### 5.2.2 Micro-Expression Detection

To implement our micro-expressions detection network, we first trained the network on selected frames from the training data sequences. For each video, we took only the onset, apex, and offset frames, as well as the first and last frames of the sequence, to account for neutral poses. Similar to Section ??, we first trained our CNN to detect emotions. We then combined the convolutional layers from the trained network, with a *long-short-term-memory* [34] recurrent neural network (RNN), whose input is connected to the first fully connected layer of the feature extractor CNN. The LSTM we used is a very shallow network, with only a LSTM layer and an output layer. Recurrent dropout was used after the LSTM layer.

We tested our network with a leave-one-out strategy, meaning one subject was designated as test and was left out of training. Our method performs at state-of-the-art level, see Table 5.3.
<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>LBP-TOP [69]</td>
<td>44.12%</td>
</tr>
<tr>
<td>LBP-TOP with adaptive magnification [48]</td>
<td>51.91%</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td><strong>59.47%</strong></td>
</tr>
</tbody>
</table>

Table 5.3: Micro-expression detection and analysis accuracy. Comparison with reported state-of-the-art methods.
Chapter 6

Conclusions

We provided a computational justification of Ekman’s facial action units (FACS) which is the core of his facial expression analysis axiomaticobservational framework. We studied the models learned by state-of-the-art CNNs, and used CNN visualization techniques to understand the feature maps that are obtained by training for emotion detection of seven universal expressions. We demonstrated a strong correlation between the features generated by an unsupervised learning process and Ekman’s action units used as the atoms in his leading facial expressions analysis methods. The FACS-based features’ ability to generalize was then verified on cross-data and cross-task aspects that provided high accuracy scores. Equipped with refined computationally-learned action units that align with Ekman’s theory, we applied our models to the task of micro-expression detection and obtained recognition rates that outperformed state-of-the-art methods.

The FACS based models can be further applied to other FER related tasks. Embedding emotion or micro-expression recognition and analysis as part of real-time applications can be useful in several fields, for example, lie detection, gaming, and marketing analysis. Analyzing computer generated recognition models can help refine Ekman’s theory of reading facial expressions and emotions and provide an even better support for its validity and accuracy.
Appendix A

Appendix

A.1 CNN Feature Visualizations

Below are additional feature visualizations similar to the ones displayed in 4.2. Each neuron visualization is shown as the reconstruction of the image region that is responsible for maximizing activation in the specified filter. For each feature we show the top 3 activations, similar to [66]. One can see the highlighted regions of interest for every neuron in the deepest convolution layer.

Figure A.1: Feature visualization for the trained network model. For each feature we overlay the deconvolution output on top of its original input image. One can easily see the regions to which each feature refers. Visualizations are from experiments on the CK+ dataset.
Figure A.2: More Feature visualizations on CK+.
Figure A.3: Feature visualization for the trained network model. Visualizations are from experiments on the FER2013 dataset.
Figure A.4: More feature visualizations on FER2013.
Figure A.5: More feature visualizations on FER2013.
A.2 Feature Correlation Tests

Below are additional correlation tests charts to those described in 4.3.

Figure A.6: Correlation results on CK+. For each neuron (filter), the best scoring action units are described. Correlation score is computed as described in 4.1.
Figure A.7: Correlation results on CK+.
Figure A.8: Correlation results on CK+. 
Figure A.9: Correlation results on CK+. 

49
Bibliography


[10] J. Bulwer. Pathomyotamia, Or, A Dissection of the Significative Muscles of the Affections of the Minde: Being an Essay to a New Method of Observing the Most Important Movings of the Muscles of the Head, as They are the Neerest and Immediate Organs of the Voluntarie Or Impetuous Motions of the Mind : with the Proposall of a New Nomenclature of the Muscles. W.W., 1649.


learning based FACS action unit occurrence and intensity estimation. In
11th IEEE International Conference and Workshops on Automatic Face and
Gesture Recognition, FG 2015, Ljubljana, Slovenia, May 4-8, 2015, pages

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual
learning for image recognition. In Proceedings of the IEEE Conference on

[34] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. Neural

[35] David H Hubel and Torsten N Wiesel. Receptive fields and functional
243, 1968.

[36] Jyoti Joshi, Abhinav Dhall, Roland Goecke, Michael Breakspear, and Gor-
don Parker. Neural-net classification for spatio-temporal descriptor based
depression analysis. In Pattern Recognition (ICPR), 2012 21st International

[37] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic opti-

[38] Irene Kotsia and Ioannis Pitas. Facial expression recognition in image se-
quences using geometric deformation features and support vector machines.

sification with deep convolutional neural networks. In Peter L. Bartlett,
Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q.
Weinberger, editors, Advances in Neural Information Processing Systems 25:
Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada,

[40] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-
based learning applied to document recognition. In Proceedings of the IEEE,

[41] Gwen Littlewort, Marian Stewart Bartlett, Ian Fasel, Joshua Susskind, and
Javier Movellan. Dynamics of facial expression extracted automatically from


more than mere representations of objects that the network is searching for. It is used in the back-propagation algorithm...

To find the maximum response of a neuron to a specific object, they added the feature of an object and the neuron that triggered the input, and then found the parts in the image of the input that respond to that neuron in the best way. Additional efforts were made to understand the learned features of the convolutional neural networks, and this was done using back-propagation.

Sometimes, when the convolutional neural network is not functioning well, it is necessary to make changes to the parameters of the algorithm, which is why it is called the back-propagation algorithm.

The contributions of this work are:

- Using new methods for visualizing convolutional neural networks to understand the models learned by the algorithms that lead the field of analysis of human facial expressions. We are leading research in human facial expressions.

- The illustrations can be used to identify the features learned by the convolutional neural networks.

- We also make sense of the cognitive states of the convolutional neural networks after they are trained on different tasks in the field of analysis of human facial expressions.

- The models are trained on FACS and are used in different applications of models based on the analysis of human facial expressions.

- We also created visual models for different facial expressions and analysis.

- We also created visual models for different facial expressions and analysis.
The page contains a continuation of a text discussing various methods for analyzing facial expressions, particularly in the context of advertising effectiveness. It mentions the use of algorithms and active contours, and references several papers and techniques. The text also touches on convolutional neural networks and their role in visualizing and understanding deep belief networks.

**Convolutional Neural Networks**

Convolutional neural networks (CNNs) play a significant role in analyzing facial expressions. They are characterized by their ability to extract features from images and videos, often using local binary patterns (LBP) and Gabor features. These networks are adept at recognizing patterns in images and are widely used in various applications, including object recognition and facial expression analysis.

**Deep Belief Networks**

Deep belief networks (DBNs) are another type of neural network that can be used for facial expression analysis. They are composed of multiple layers, each of which learns to represent features at a different level of abstraction.

**Conclusion**

The document concludes by emphasizing the importance of understanding the underlying mechanisms of facial expression recognition, which can be achieved through the use of advanced computational techniques and neural networks.
התקשורת האנושית מכילה הרבה יותר מחלקית דיבור, מילים, ומשפטים. הבעות פנים היא בובי, כי משמשות את התוכן והמטרה של הדו-שיח בין אדם. הבעות פנים נמצאות במגוון רחב של שורות, כמו ב缕ה או בתוכן מת来る, וдает קשר לקשר בין הדיבור בתקשורת המילולית. זו אחת התוכן האנושיות של בעיות פנים כיותח מקובל בשנות העשרים מלך מגוון שביו.

מדע הרוחה, פסיכולוגיה, נוירולוגיה ובינה מלאכתית.

ניתוח הבעות פנים

ניתוח רגשות האנושיים דרך הבעות פנים הוא חלק מרכז במחקרים פסיכולוגיים. דרווין, בשלהי המאה ה-19, הציג את הבעות הפנים קשורות בנושאים אבולוציוניים [16]. דרווין הציג כי הבעות פנים של יצורים אחרים הם חלק מהこれまでים של תפקודים שונים של האדם. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים שהם יוצרים. הבעות פנים הן חלק מהחריטות וה莈ים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התמודדות של אדם עם הנוכחות של אחרים והקשרים הם יוצרים. הבעות פנים נמצאות בין התmonds...
מוצאת הבעצת הפנים: פרוספקטיבית יהובית בעורר למידה עמידה

חיבור על מחקר

לשם מולי חוקי של הדרישה ל鳙ולות התאור
מניסור למדעים במדעי המחשב

ור ברויאר

רותם לתניניו הסכיניים ממכון טכנולוגי ישראל
אייר והמשם של הפרסום מאי 2017
מוצאת הביצור הפינים: פרспектיבת היישובית בחירה למידה עמותה

ן ברויאר