Abstract

Facial micro-expressions are quick, involuntary and low intensity facial movements. An interest in detecting and recognizing micro-expressions arises from the fact that they are able to show person’s genuine hidden emotions. The small and rapid facial muscle movements are often too difficult for a human to not only spot the occurring micro-expression but also be able to recognize the emotion correctly. Recently, a focus on developing better micro-expression recognition methods has been on models and architectures. However, we take a step back and go to the root of task, the data.

We thoroughly analyze the input data and notice that some of the data is noisy and possibly mislabelled. The authors of the micro-expression datasets have also acknowledged the possible problems in data labelling. Despite this, no attempts have been made to design models that take into account the potential mislabelled data in micro-expression recognition, to our best knowledge. In this paper, we explore new methods taking noisy labels into special account in an attempt to solve the problem. We propose a simple, yet efficient label refurbishing method and a data cleaning method for handling noisy labels. The data cleaning method achieves state-of-the-art results in the MEGC2019 composite dataset.

Introduction

As opposed to the typical macro-expressions each of us is confronted every day, micro-expressions (MEs) have a significantly lower intensity and a quicker duration [17]. The motivation for studying micro-expression arises from the fact that they are involuntary, meaning that micro-expressions show a person’s genuine feelings. The ability to see person’s true feelings has tremendous applications in psychotherapy, medical applications, business negotiations, lie detecting, security and marketing research [17].

However, due to the characteristics of MEs (low intensity, rapidness and involuntary) the task is not easy. In fact, the task is even difficult for humans. A micro-expression training tool [17] was developed to teach people how to spot and recognize emotions, but even after the training the accuracy was a mere 50%. Due to the potential applications and the difficulty of the task for humans, there has been an increasing amount of works that try to solve the problem of recognizing micro-expressions automatically.

Previous works on ME recognition have mainly focused on modelling and architectures. A problem that has been completely neglected to the best of our knowledge is the reliability of the datasets—more precisely the reliability of the labels for the samples. The authors of the ME datasets [23][2] have acknowledged the difficulty and ambiguity of the labelling process. In fact, the inter-coder variability of labelling the AUs (action units) is only around 80% [23][2]. The labelling is ambiguous as the same set of AUs may correspond to different emotions [30]. Another reason is the subjectivity of emotions, the personal feelings of the subject may not correspond to the labels of the annotator.

To see the extent of ambiguities, we manually go through the datasets to observe if the datasets contain ambiguous data. We find that there are indeed many ambiguous samples, including possibly mislabelled samples within the datasets. These mislabelled samples are often referred to as noisy labels in the literature [6][21].

Motivated by these findings, we propose to use noisy label methods from the literature in an attempt to solve the problem. However, we found that the methods presented in the literature of noisy label methods did not work as effectively as expected. We hypothesize that this is due to the methods being designed for vastly different datasets with injected label noise, whereas we have a real world dataset. A similar finding was done in [5], where the authors notice the differences between synthetic datasets and real-world datasets. Therefore, we propose our own methods that are more suitable for ME data, but also potentially for similar datasets with the characteristics of a small number of samples and ambiguous data (as opposed to easily identifiable mislabelled...
samples).

In this paper, we perform a qualitative analysis of ME data and develop noisy label methods for the potentially mislabelled samples. We conduct extensive experiments using multiple different noisy label methods and compare our developed methods to the state-of-the-art results in ME recognition.

Related Work

A Micro-expression analysis system can be divided into two parts. Firstly, the objective is to spot the occurring micro-expression. The output of spotting should be the onset and offset frame positions in which the ME is occurring. Secondly, after obtaining the sequence in which the ME is occurring, the task is to recognize the emotion with its corresponding category. In this paper, we only consider the latter task of recognizing the emotions, due to the complexity of the task. We will also look at the noisy label problem and the methods developed in an attempt to solve the problem from literature.

Micro-expression recognition methods

Previous work on ME recognition has been mainly in two categories: traditional methods and deep learning methods. The traditional methods refer to feature extraction by hand and they can be further split into two categories, appearance features and movement features. Many of the appearance-based methods are based on LBP (Local Binary Pattern). One of the most common benchmarks is the LBP-TOP (LBP on Three Orthogonal Planes) [31]. LBP based methods have further been explored by LBP-SIP (LBP with Six Intersection Points) [23], LBP-MOP (LBP with Mean Orthogonal Planes) [24] and ELBP-TOP (Extended LBP-TOP) [4]. These methods focus on the pixel intensities, while movement based methods rely on the optical flow (OF). OF calculates the difference between two frames to find the movement between them. Methods using this strategy are HOOF (Histogram of Oriented Optical Flows) [1], MDMO (Main Directional Mean Optical Flow) [15], MDMOSparse [14] and Bi-WOOF [11, 8]. The deep learning methods instead use an automatic technique to extract the features. Methods from this section include Off-ApexNet [3], STSTNet [9] and NMER [13]. The deep learning methods have only recently become the state-of-the-art methods, due to the lack of data in ME datasets.

Noisy labels

Noisy labels refer to samples with incorrect labels. There are two main categories and common reasons for the mislabelled samples. 1) Massive datasets that have been generated without the supervision of humans. An example of this is WebData that is created by crawling images from the internet and labelling them based on the text around the image [21]. 2) Datasets with ambiguous samples. These are often smaller datasets and have been annotated by multiple humans, but due to the innate difficulty and ambiguity of the samples, there may be mistakes. Examples include medical datasets and ME datasets.

We formally define the noisy label problem here. Let \( \mathcal{X} \subseteq \mathbb{R}^d \) be the space of inputs and the space \( \mathcal{Y} = \{1, \ldots, c\} \) be the label space, where \( d \) is the dimension of the inputs and \( c \) is the number of classes. In a typical case, we hope for the training set to consist of clean tuples of \((x, \hat{y}) \in (\mathcal{X}, \mathcal{Y})\), but in a noisy label case we have \((x, \tilde{y}) \in (\mathcal{X}, \tilde{\mathcal{Y}})\), where \( \tilde{\mathcal{Y}} \) is the set of noisy labels.

Table 1. Distributions of emotion classes for MEGC2019

<table>
<thead>
<tr>
<th>Emotion class</th>
<th>Positive</th>
<th>Negative</th>
<th>Surprise</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMIC</td>
<td>51</td>
<td>70</td>
<td>43</td>
<td>164</td>
</tr>
<tr>
<td>CASME II</td>
<td>32</td>
<td>88</td>
<td>25</td>
<td>145</td>
</tr>
<tr>
<td>SAMM</td>
<td>26</td>
<td>92</td>
<td>15</td>
<td>133</td>
</tr>
<tr>
<td>Combined</td>
<td>109</td>
<td>250</td>
<td>83</td>
<td>442</td>
</tr>
</tbody>
</table>

The task is to use the noisy set \((\mathcal{X}, \tilde{\mathcal{Y}})\) to learn the dataset as well as possible.

Noisy labels in micro-expressions

The noisy labels in micro-expressions may be caused by several possibilities. The labels are obtained from a convex combination of the following: 1) the subjects’ own feelings, 2) the emotion from the eliciting video and 3) the annotator’s subjective opinion. Any of these options are subject to mistakes. For example, different subjects may respond differently to a scary video—some find it scary but others just disgusting. In addition, the annotator may think of the emotion as something different, giving us a total of three different labels from each source. Further details on the ambiguity of the samples and noisy labels in micro-expressions can be found from [22].

Small-loss trick

A crucial finding that is commonly used throughout the different noisy label methods, named as the small-loss trick by [21].

The loss of a single sample is determined by the DNNs prediction. For example, the frequently used categorical cross-entropy \(-\sum_{\mathcal{Y}} y_n \log(\hat{y}_n)\) measures the difference between two distributions, where \(\mathcal{Y}\) is the set of possible classes. If the distribution of the predicted label \(\hat{y}\) is similar to that of the real label \(y\), the loss will be small. If the distributions are different, then the loss will be high. For representative and unambiguous clean samples we would expect the loss to be small at the end of the training. For difficult clean samples, the loss may be high as the network is not able to fully match the distributions. For samples with noisy labels, we expect the loss to be high, as the network is predicting a distribution that does match the real label’s distribution.

Noisy label methods

The noisy label methods can be roughly split into six different categories: loss functions, label cleaning, label refurbishment, transition matrices, loss reweighting and training procedures. The small-loss trick is utilized by q-percentile [6], which sorts the losses and discards q% of the samples with the highest losses. Generalized cross-entropy [29] uses a convex combination of \(\ell_1\) loss and the cross-entropy loss, as the \(\ell_1\) loss is able to find samples with high loss values, while cross-entropy keeps the training stable. T-revision [26] initially estimates the transition matrix \(T\) between the real labels \(y\) and the noisy labels \(\hat{y}\), using the predictions from the used network and the matrix is later revised by adding a slack variable to the initial estimation. Meta-Weight-Net [19] uses meta-learning to learn the weighting function of loss values, as opposed to manually selecting it like focal loss or self-paced learning. Manifold mixup [23] is a general regularizer that has also been used for noisy label learning. It creates new samples by a convex combination of samples’ hidden layers and their labels.
A look at the data

This section provides a qualitative analysis of the ME data, where we will look at the the OF (optical flow) between the onset and the apex frame from the samples in the MEGC2019 [18] dataset. MEGC2019 is a composite dataset consisting of SMIC [7], CASME II [28] and SAMM [2]. The reason for combining the datasets is the sheer lack of samples, which can be seen from Table [1]. Each sample consists of three channels: the horizontal component of OF, the vertical components of OF and the optical strain denoted by the triple \((V_x, V_y, \varepsilon)\). We observe the OF domain as spotting and recognizing emotions from the original RGB videos is too difficult. Most recent works also use the OF as their input, as methods with RGB video input have not been nearly as effective as OF.

Figure 1 shows the mean samples of each emotion class. The surprise emotion has mainly movements on the forehead and eyebrows. The positive emotion mainly has movement on the left (from the participant) cheek. By looking at the optical strain, the movement seems to be on both cheeks and the mouth area. The negative emotion does not seem to have any distinct movements, there are some movements around the eyebrows, but also near the mouth. We believe this is due to the aggregated classes, as the MEGC2019 negative contains multiple other subclasses. The eyes are highlighted on all classes due to blinking. We will use these distinct movements of each class to search for any inconsistencies in the dataset.

Figure 2 shows the first 12 samples of MEGC2019. We can immediately observe that many of the samples contain noise at some level. An interesting discovery can be made from samples eight, nine, ten and eleven. All of these samples look similar, but one of them is labelled as positive while the others are labelled as surprise. Sample number eight also contains some movements on the cheeks which may have been the reason for the label positive, but so does sample number ten and it has not been labelled as positive. This is a typical example of an ambiguous sample, where there could even be two correct labels, as the sample contains two distinct movements for two different emotions. However, confirming that this sample has a noisy label is difficult as we do not have an expert to confirm the finding and since this is only the apex of the OF, many more things could be happening in the other frames in the original domain. Nevertheless for the model performing the classification this sample will most likely be classified as surprise due to the distinct movements in the forehead. We find more similar samples by going through the dataset and hypothesize that these are samples that have been labelled incorrectly.

Methodology

In this section we will introduce our developed methods initial data cleaning (IDC) and iterative label correction (ILC) to deal with the noisy labels. In addition we provide a neural network architecture based on the related work.

Initial data cleaning (IDC)

It was found in [6] that only using clean data (10% of the full data) can lead to a better performance than using the whole dataset, including the noisy samples. Inspired by this, we manually go through the dataset to find clean samples similarly to the previous section. From the 442 samples of MEGC2019 we find a total of 240 clean samples and 202 samples that are noisy, have a potentially incorrect label, ambiguous or do not fit the typical characteristics of MEs found in the previous section.

However, after experimentation, we found that the performance had decreased when only training with the clean samples. A finding in [21] points out that data cleaning methods may be cleaning too many samples, discarding samples that are crucial for the training. In the work [12] the authors find that regularizing the network early significantly prevents the network from memorizing the noisy labels in the later stages of the training. Inspired by this we propose a warm-up period during which the network is only trained with clean labels for some duration of epochs, hence the data cleaning is only done initially. After the warm-up period, the network has access to all of the data.

Loss thresholding with moments (LTM)

We introduce loss thresholding with moments here as it is used as a sub method in ILC. We use an alternative way of performing the \(q\)-percentile also based on the small-loss trick. One of the downsides of \(q\)-percentile is the need for a warm-up period as the small-loss trick does not apply at the beginning of the training when the losses are essentially random. In [6] the authors propose to store the mean \(\mu\) and standard deviation \(\sigma\) of the losses of the
most recent 100 training samples. Then they threshold the values
if the loss value is higher than \( \mu + 1.5 \sigma \). We use the same idea
but set the coefficient to be a hyperparameter \( \alpha \), thus the threshold
value is then given by

\[
t = \mu + \alpha \sigma.
\]

The hyperparameter \( \alpha \) behaves similarly to \( q \) and is required as
different datasets have different levels of noise. The number of
training samples from which the mean and standard deviation are
calculated is set to be equal to the number of samples in a batch.
This allows us to perform an update based on the most recent loss
values and have enough samples to calculate the moments. In
addition, by calculating the moments from the batch samples, we
can avoid more complex implementations. The value may have to
be adjusted depending on the size of the dataset and if the ratio
of noisy labels is small. We refer to the method as loss thresholding
with moments (LTM) as we use the first and second moments of
the distribution to formulate the threshold value.

The set of samples with clean labels

\[
C = \{ (x, y) \in (X, Y) \mid L(f(x; \theta), y) < t \}
\]

is then obtained by thresholding the loss values obtained from an
arbitrary loss function \( L \) of each sample, where the network \( f \)
gives the prediction. Then, the samples with low loss values are
ignored when updating the parameters of the network \( \theta \) and only
the set of clean samples is used in the update. The benefit of
using LTM over \( q \)-percentile is that there is no need for a warm-
up period at the start of the training. As the threshold value \( t \)
given as a function of the moments of the loss distribution, the
number of discarded samples is adaptive and there is no need for
a warm-up period.

Figure 3. The figure depicts iterative label correction. (Left) We start with
a set of noisy labels. (Middle) We detect the noisy labels similarly to LTM.
(Right) The samples that are detected as noisy labels are refurbished by the
network’s most recent prediction and we are left with the clean set \( C \) and the
corrected set \( \hat{C} \) given by Equation (3). By correcting the labels we are left
with the same amount of samples that we started with.

**Iterative label correction (ILC)**

An issue with LTM, and overall with methods discarding samples, is that the discarded samples may be important for the
training in small datasets. We can include the detected samples
with noisy labels by refurbishing the noisy label with a corrected
one as shown in Figure 3. Our method is similar to SELFIE [20],
but instead of using \( q \)-percentile we apply the LTM for finding the
noisy labels. The refurbished label is given by the current prediction
of the network instead of a uncertainty condition employed by
SELFIE. ILC only corrects the labels of samples that have been
detected as high loss samples, while SELFIE also includes some
of the clean samples.

The set of samples with corrected labels

\[
\hat{C} = \{ (x, \hat{y}) \mid (x, y) \in C, \hat{y} = f(x; \theta) \}
\]

is given by the predictions of the network for samples that were
not found to be clean, i.e., the complement of \( C \). The parameters
\( \theta \) are then updated using the set \( C \cup \hat{C} \).

The correction of labels can be risky however. If the network
has not learned the dataset correctly, it is not able to distinguish
between the samples and the corrected labels may be incorrect.
Giving the high loss samples incorrect labels could lead to an in-
crease in the number of noisy labels.

**Shallow Single Stream Network**

We introduce SSSNet (Shallow Single Stream Network) that
is motivated by Off-ApexNet [10], STSTNet [9] and from the
findings in [27]. One of the problems with Off-ApexNet and
STSTNet is that they have a separate stream for each of the input
channels (\( V_x, V_y, e \)). This does not allow the network to combine
the learned information from each of the channels until the fusion
of the features. We simplify the network by only using a single
stream that contains all input channels. The structure of the net-
work can be seen in Figure 4. We refer to SSSNet24 as the model
in which the number of filters in the second convolutional layer is
changed to 24.

**Experiments**

This section presents the experimental settings and the re-
sults of our experiments. We first start by comparing different
noisy label methods, next we analyze the results and the effec-
tiveness of the methods and lastly compare our methods to the
state of the arts.

**Evaluation methods**

The evaluation is performed using the LOSO (leave-one-
subject-out) protocol. In LOSO the data is split into training and
testing data such that the testing set only contains samples from a
single subject and the training set contains the rest. This is done
due to the results being highly subject dependent. The performance is
measured by the unweighted

\[
F1 = \frac{1}{C} \sum_{c=1}^{C} \frac{2TP_c}{2TP_c + FP_c + FN_c}
\]

and the

\[
UAR = \frac{1}{C} \sum_{c=1}^{C} \frac{TP_c}{N_c},
\]

where \( C \) is the number of classes and \( N \) is the number of samples,
as the classes are imbalanced.

**Comparison with noisy label methods**

Table 2 showcases the results from different noisy label
methods. SSSNet24 is used for the baseline, as well as the back-
bone for the other methods. In the middle part of the table we
showcase the results using methods from the literature. All of
the methods are able to outperform the baseline for MEGC2019 except for T-revision. We hypothesize that since the matrix transition methods model the transition of classes, it is better suited for methods with a high number of noisy labels and larger datasets. In addition, the transition matrix only mitigates the effects of noisy labels, but we found that the data also contains samples with noise in the data itself, which the other methods are also able to detect. The other methods are able to increase the performance of the baseline, but quite moderately. In the bottom section of the table, we have the proposed methods. These methods were found to better perform for ME recognition, as they are specifically designed for the task. The initial data cleaning method is able to increase the baseline by 4%, while the loss thresholding with moments and iterative label cleaning fall a bit short of that.

The success of multiple different noisy label methods indicates the presence of noisy labels in the dataset. The relative incremental increases in performance can be explained as some of the methods are designed for larger datasets with higher amounts of noisy labels. In addition, a recent work from [5] discusses the differences between real-world label noise and synthetically generated label noise and finds the distributions to be relatively different. And since most of the noisy label methods in the literature have been developed using synthetically generated noise, they may fall short with real world data. Probably the biggest reason as to why the improvements are only incremental compared to what the noisy label methods are able to achieve in their own benchmarks, is that we do not have access to clean validation data. Due to the constraints of the LOSO protocol, all samples have to be tested. This means that even if we were able to perfectly detect all noisy samples, the improvement would be marginal as the testing data still contains noisy labels. However, the improvements achieved here are still a sign that the methods are able to better generalize and do not overfit to noisy labels as heavily.

Figure 5 shows a projection from the last hidden layer of SSSNet24 to two dimensions using UMAP (uniform manifold approximation and projection) [16]. The classes cluster relatively well and the class borders can be roughly drawn. Some samples seem to be in the wrong clusters, which we have been able to detect using loss thresholding with moments. Unfortunately, not all samples that seem to be in the wrong clusters have been detected. In addition, samples with a correct label may also have
been marked as high loss samples as can be seen with samples marked at the classification border of negative and positive.

Figure 5 shows the same projection, but the labels from the samples that were detected as high loss samples have now been corrected with ILC. Here we can see that essentially all of the samples were corrected to a label that best represents their neighboring points. Once more, some arguments can be made about the samples in the classification border of negative and positive.

Comparison to state of the arts
We compare our methods to the state-of-the-art methods from the literature in Table 3. At the top we have the two traditional methods LBP-TOP and MDMO. Both of the methods achieve similar results at around 0.58 F1. In the middle we have the current state-of-the-art methods for ME recognition that have been evaluated using the MEGC2019 dataset. SSSNet24 with the proposed methods are presented in the bottom of the table. The IDC method is able to achieve the state of the art for ME recognition of 0.77 in the F1-score. Both the ILC and IDC are very close to breaking the 0.90 mark for F1 in the CASME II dataset. Significant increases can be seen in both CASME II and SAMM compared to the previous state-of-the-art methods.

Conclusions
We have showcased the effectiveness of using noisy label methods for micro-expression recognition, although previous ME recognition methods have ignored the ambiguity of the data. We started off by observing the labelling process of the datasets and by looking at individual samples, in order to better understand the problem of noisy labels. We proposed two new methods: initial data cleaning and iterative label correction, to address the issue of noisy labels. In initial data cleaning, we manually go through the dataset to find clean samples and only use the clean samples in the initial stages of the training. Iterative label correction finds noisy samples using loss thresholding with moments and then corrects the noisy labels using the network’s predictions. We perform experiments using noisy label methods for micro-expression recognition and find that nearly all of the methods are able to increase the performance, indicating the existence of noisy samples in micro-expression datasets. Besides, we achieve the state of the art for MEGC2019 with an F1-score of 0.77 by using the initial data cleaning method showing its effectiveness. As a future work, we hope to study the relationship between subjective facial emotion perception and the proposed methods for finding samples with noisy labels.

Acknowledgements
This work was supported by the Academy of Finland for ICT 2023 project (grant 328115), project MiGA (grant 316765), and Infotech Oulu. As well, the authors wish to acknowledge CSC – IT Center for Science, Finland, for computational resources.

Table 3. Comparison to the state-of-the-art methods

<table>
<thead>
<tr>
<th>Model</th>
<th>MEGC2019 UAR</th>
<th>MEGC2019 F1</th>
<th>SMIC UAR</th>
<th>SMIC F1</th>
<th>CASME II UAR</th>
<th>CASME II F1</th>
<th>SAMM UAR</th>
<th>SAMM F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>LBP-TOP [31]</td>
<td>0.5785</td>
<td>0.5882</td>
<td>0.5280</td>
<td>0.2000</td>
<td>0.7429</td>
<td>0.7026</td>
<td>0.4102</td>
<td>0.3954</td>
</tr>
<tr>
<td>MDMO [15]</td>
<td>0.5782</td>
<td>0.5881</td>
<td>0.5511</td>
<td>0.5587</td>
<td>0.7925</td>
<td>0.8014</td>
<td>0.3073</td>
<td>0.3065</td>
</tr>
<tr>
<td>Off-ApexNet [13]</td>
<td>0.7164</td>
<td>0.7176</td>
<td>0.6972</td>
<td>0.7039</td>
<td>0.8094</td>
<td>0.8159</td>
<td>0.6172</td>
<td>0.6126</td>
</tr>
<tr>
<td>STSTNet [9]</td>
<td>0.7125</td>
<td>0.7095</td>
<td>0.6726</td>
<td>0.6618</td>
<td>0.8132</td>
<td>0.8325</td>
<td>0.6257</td>
<td>0.6176</td>
</tr>
<tr>
<td>NMER [13]</td>
<td>0.5936</td>
<td>0.5936</td>
<td>0.5555</td>
<td>0.5607</td>
<td>0.6929</td>
<td>0.7624</td>
<td>0.4894</td>
<td>0.6389</td>
</tr>
<tr>
<td>RCN-A [27]</td>
<td>0.7138</td>
<td>0.7168</td>
<td>0.6619</td>
<td>0.6590</td>
<td>0.7894</td>
<td>0.8109</td>
<td>0.6531</td>
<td>0.6547</td>
</tr>
<tr>
<td>SSSNet24</td>
<td>0.7505</td>
<td>0.7415</td>
<td>0.7071</td>
<td>0.7005</td>
<td>0.8739</td>
<td>0.8716</td>
<td>0.6417</td>
<td>0.6302</td>
</tr>
<tr>
<td>SSSNet24 + LTM</td>
<td>0.7647</td>
<td>0.7553</td>
<td>0.7009</td>
<td>0.6895</td>
<td>0.8814</td>
<td>0.8838</td>
<td>0.6848</td>
<td>0.6738</td>
</tr>
<tr>
<td>SSSNet24 + ILC</td>
<td>0.7729</td>
<td>0.7639</td>
<td>0.6979</td>
<td>0.6919</td>
<td>0.9051</td>
<td>0.8995</td>
<td>0.6919</td>
<td>0.6829</td>
</tr>
<tr>
<td>SSSNet24 + IDC</td>
<td>0.7794</td>
<td>0.7707</td>
<td>0.7090</td>
<td>0.6979</td>
<td>0.9060</td>
<td>0.8996</td>
<td>0.6990</td>
<td>0.6919</td>
</tr>
</tbody>
</table>

References
[10] LIANG, S.-T., GAN, Y. S., YAU, W.-C., HUANG, Y.-C., and


Author Biography

Tuomas Varanka received his B.S. and M.S. degree in computer science and engineering from the University of Oulu, Finland, in 2019 and 2020 respectively. He is currently pursuing his Ph.D. degree in University of Oulu. His work has focused on micro-expression recognition.

Wei Peng is currently a Ph.D. candidate with the Center for Machine Vision and Signal Analysis, University of Oulu, Oulu, Finland. He received the M.S. degree in computer science from the Xiamen University, Xiamen, China, in 2016. His articles have published in mainstream conferences and journals, such as AAAI, ICCV, ACM Multimedia, Transactions on Image Processing. His current research interests include machine learning, affective computing, medical imaging, and human action analysis.

Guoying Zhao received the Ph.D. degree (2005) in computer science from the Chinese Academy of Sciences, Beijing, China. She is currently a full professor with the Center for Machine Vision and Signal Analysis, University of Oulu, Finland. Her work has focused on affective computing and machine learning. She is IEEE senior member and ELLIS member, and associate editor for Pattern Recognition, IEEE Transactions on Circuits and Systems for Video Technology, and Image and Vision Computing Journals.
JOIN US AT THE NEXT EI!

IS&T International Symposium on

Electronic Imaging

SCIENCE AND TECHNOLOGY

Imaging across applications . . . Where industry and academia meet!

• SHORT COURSES • EXHIBITS • DEMONSTRATION SESSION • PLENARY TALKS •
• INTERACTIVE PAPER SESSION • SPECIAL EVENTS • TECHNICAL SESSIONS •

www.electronicimaging.org