Analysis of Students' Concentration Levels for Online Learning Using Webcam Feeds

Linh Le*, Ying Xie*, Sumit Chakravarty†, Michael Hales‡, John Johnson‡, Tu N. Nguyen*
*College of Computing and Software Engineering
†Southern Polytechnic College of Engineering and Engineering Technology
‡WellStar College of Health and Human Services
Kennesaw State University, Marietta, GA 30060, USA
Email: {linh.le, yxie2, schakra2, mhales, jjohn235, tu.nguyen}@kennesaw.edu

2021 IEEE International Conference on Big Data (Big Data) | 978-1-6654-3902-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/BigData52589.2021.9671466

***Abstract*—Tracking the concentration of students during online learning offers great benefits. For examples, distracted students can be suggested to do a brief exercise to refresh their brains; or the teacher can be notified when too many students have difficulties on concentration so the class could take a short break. Traditionally, mental states like concentration levels can be analyzed using Electroencephalogram (EEG) or Functional Near-Infrared Spectroscopy (fNIRS). However, methods that utilize these data require specialized equipment which is not feasible to deploy on a large scale. On the other hand, recent breakthroughs in deep learning provide possibilities of scalable solutions to detect concentration levels using only webcam. Leveraging this advancement, we investigate the task of tracking students' concentration levels during online learning using facial data coupled with deep learning based computer vision technologies. More specifically, we examine the performances of different representations of facial data integrated with various deep architectures to empirically determine a solution balanced between prediction accuracy and time efficiency that is suitable for real-time application. Our experimental study shows that the proposed solution achieves over 91% accuracy while keeping execution time low enough for real-time deployment.**

***Index Terms*—concentration level tracking, facial video classification, deep learning, machine learning, computer vision**

I. Introduction

Concentration is among the key factors associated with successes in learning [1]. To achieve good academic performances, students should maintain a good level of concentration during their study. However, concentration levels quickly drop as the brain endures intensive activities such as studying [2]. Due to the COVID-19 pandemic, more classes are moving online where the isolated learning environment and other distracting factors found outside a classroom may cause more issues on students' concentration capabilities. Tracking students' concentration states during online classes, therefore, may offer great benefits for both learning and teaching. For examples, distracted students can be suggested to do a brief exercise to refresh their concentration capability; or the teacher can be notified when too many students have difficulties concentrating on the lecture so the class could take a short break. With such motivation, in this paper, we investigate the task of tracking students' concentration levels during online learning. Our research goal is to find a solution that could be applied to real-time online learning at a large scale.

Traditionally, human mental states, including concentration levels, are analyzed through Electroencephalogram (EEG) [3] or Functional Near-Infrared Spectroscopy (fNIRS) [4]. However, collecting these types of data requires the use of relatively expensive equipment, which hinders its wide application at a large scale. Recently, the area of deep learning has seen breakthroughs that make it possible to produce scalable solutions using computer vision through the video data collected by webcam. Facial data captured by webcam, coupled with deep neural networks, have been utilized on modeling different human mental states [5], [6], [7], [8], [9] with reasonable successes. These works typically use Convolutional Neural Network (CNN) [10] and its variants on facial images or video frames for emotion or expression recognition [11]. In the narrower area of analyzing concentration levels during learning, however, there are currently a limited number of works that uses deep-learning based computer vision. The research described in [12] is one of those. In this research, the authors propose a temporal CNN to track engagement levels of children and achieves 80% accuracy rate on several datasets. Overall, we feel that it is worth further investigating on how to leverage deep-learning based computer vision in analyzing students' engagement.

A challenge with analyzing video is the high-dimensional and density of the data. Additionally, machine learning models that analyze video data can be computationally expensive for real-time applications. To address this issue, we first examine methods that transform facial video into different representations including facial landmarks [13], eye movements [14]; embedding vectors generated by Google FaceNet [15]; and embedding vectors generated by VGG-Face descriptor [16]. Then, we couple these data with various deep learning architectures such as CNN and its 1D and 3D variants, Gated Recurrent Unit (GRU) [17], bi-directional GRU, and Attention network [18]. Our goal is to empirically design an end-to-end concentration level tracking framework that is balanced between accuracy and efficiency as such applicable in a real time applications on large scales. Our study presents two models which demonstrates over 90% accuracy rates while keeping execution time low enough for real-time application.

978-1-6654-3902-2/21/$31.00 ©2021 IEEE

Authorized licensed use limited to: JIANGSU NORMAL UNIVERSITY. Downloaded on July 20,2023 at 16:08:41 UTC from IEEE Xplore. Restrictions apply.

In short, our **contributions** and **intellectual merits** in this paper can be summarized as follows:

1) We present an in-depth study on the task of analyzing students' concentration levels in online learning using facial videos, where we examine the accuracy and execution time of different representations of facial data on various deep learning architectures. To the best of our knowledge, similar studies have not yet been seen in literature.
2) We present two deep-learning-based concentration analyzing architectures that are able to achieve over 90% in terms of accuracy rate while maintaining reasonable time efficiency for real time applications.

The rest of the paper is organized as follows. In Section II, we formally define the problem that this paper is focusing. In Section III, we review the deep learning technologies that are relevant to our proposed methods. Section IV discusses feature extraction methods for video data. Section V and VI describe our deep learning models and experiment results. Finally, we conclude the paper in Section VII.

II. Problem Definition

In this section, we formally describe the task of analyzing concentration levels from webcam feeds. Our first assumption is that concentration levels, while varying during a studying session, are relatively stable in small time windows (e.g., a few seconds). Accordingly, the input to our task is a short video segment fed by the webcam. The output of our task given the input of a short video segment is one of the two binary labels, *"concentrate"* or *"distract"*. Therefore, the overall task is a binary classification on a short video segment.

Video data is usually 4D tensors of which elements are the pixel values across dimensions include frames, frame size - width and height, and color channels. A video of f frames at resolution $w \times h$ with c color channels is therefore represented by a tensor V of size $f \times w \times h \times c$. Furthermore, techniques that handle video data should be able to model the spatial correlations of pixels in the same frame, and the temporal correlations among frames in the same video segment.

Alternatively, video data can be transformed into other representations before being input in a classifier. The correct choices of representation methods may significantly affect both the classification accuracy and the execution time. Accordingly, we will explore different representations besides the original videos in this work. More specifically, we consider three types of representations that are 1) facial landmarks [13] coupled with eye movements [14]; 2) embedding vectors generated by Google FaceNet [15]; and 3) embedding vectors generated by VGG-Face descriptor [16].

The choice of data representations also significantly decides the deep learning architectures to be used. For landmark representation and embedding vectors, we develop models that are based on GRU, bi-GRU, and 1DCNN. For video data, we utilize deep architectures that are integrated among 2DCNN, 1DCNN, GRU, bi-GRU, and 3DCNN. Overall, our research goal is to determine an end-to-end analysis framework, including the data representation and the deep neural network architecture, that yields the best balance in terms of accuracy rates and time efficiencies.

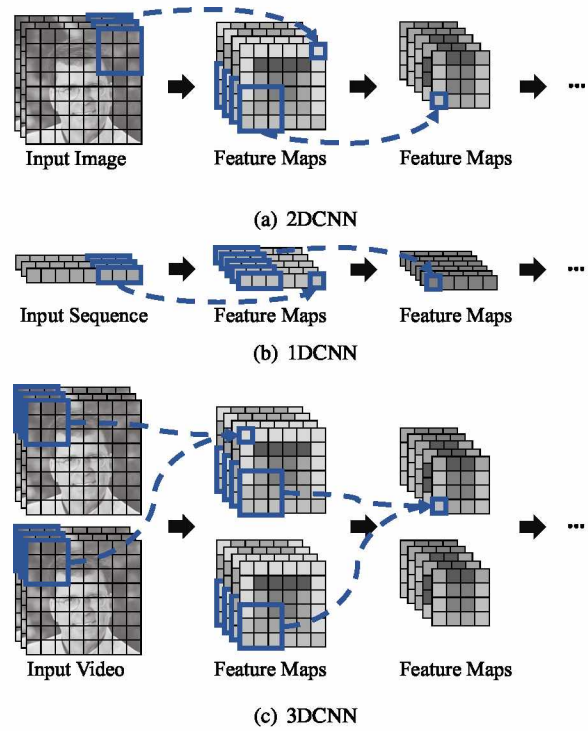


Fig. 1: Computational processes of convolutional architectures (facial image obtained from public domain)

III. Deep Learning Technologies for Video Data

In this section, we briefly describe the deep learning technologies that can be utilized for the task of analyzing concentration levels from facial videos. The most important architecture for our work is Convolutional Neural Network (CNN) [10]. CNN is a type of locally connected neural network that is designed for image data. A layer of CNN maps an image to sets of feature maps using a set of filters that move horizontally and vertically across the input matrix. Each element at position (i, j) in the output is a nonlinear transformation (by applying a filter) of a patch of elements $((i-k \rightarrow i+k), (j-k \rightarrow j+k))$ from the input matrix (where 2k is the number of filters). The feature maps output by a CNN layer can then be used as input to another CNN layer. In general, CNN continuously maps its input to higher level representations feature maps with lower dimensionality. An output architecture can be laid upon the highest level feature map for decision making. The architecture of CNN is illustrated in Figure 1(a) in which the input is an input image of three color channels. A set of filters move across the image to generate the feature maps, with one set of filter yield one feature map. Henceforth, we refer to this type of CNN as 2DCNN as the filers move in two directions.

2DCNN is designed for images, and therefore cannot effectively handle videos which are a sequence of images. To capture the temporal correlation among frames in a video,

Authorized licensed use limited to: JIANGSU NORMAL UNIVERSITY. Downloaded on July 20,2023 at 16:08:41 UTC from IEEE Xplore. Restrictions apply.

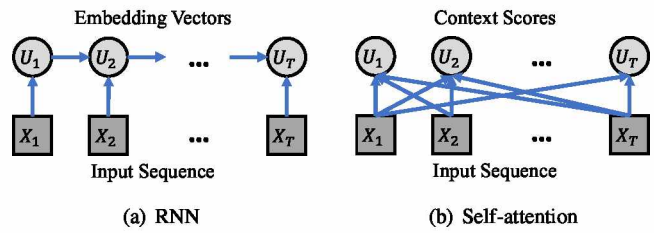


Fig. 2: Computational processes of recurrent neural network and self-attention network

2DCNN can be combined with one of its variants that are designed for sequential data, the 1DCNN. Similar to 2DCNN, 1DCNN utilizes sets of 1D filter to continuously maps input sequence to a shorter output sequence but at a higher level representation. Specifically, element at step t of the output is computed from elements from $t-k$ to $t+k$ in the input. We illustrate the computational logic of 1DCNN in Figure 1(b). In this case, the set of filters only move along the sequential axis (that usually represents the time axis), hence the name 1DCNN. Another CNN variant which is 3DCNN can be considered a combination of 2DCNN and 1DCNN in which the input and output are sequences off 2D images. In 3DCNN, each value at position (i, j) at step t of an output sequence is computed from patches at $((i-k \rightarrow i+k), (j-k \rightarrow j+k))$ in the input sequence across step $t-k$ to $t+k$. We show the computational process of 3DCNN in Figure 1(c). A filter is applied on patches at the same locations in consecutive steps in the input sequence to generate one feature map at one step in the output sequence.

GRU [17] is a different type of neural network for sequential data that is among the family of recurrent neural network (RNN) [19]. In brief, RNN transforms an input sequence into an output sequence in which the *state at each step* t is computed from the input state at t and output state at $t-1$. This design allows each state in the output sequence to capture historical patterns from the previous steps. At the moment, GRU is among the more advanced variants of RNN that is balanced between accuracy and complexity. GRU can be modified to capture information from steps *before and after* the current one, in which case the model is called bidirectional-GRU (biGRU) [20]. GRU and biGRU can be applied on top of 2DCNN which helps the complete architecture handle temporal correlations in data. We will also use 1DCNN, GRU, and biGRU, to model *embedding vector sequences* that are 1D representations of video data generated by different descriptors. Finally, we want to mention the Self-Attention mechanism [18] which is more recently developed compared to the other mentioned architectures. Given an input sequence, an attention layer computes a "context" score of each state with respect to the rest of the time steps. This gives attention mechanism great representation capabilities over sequential data. We show the computation flows of RNN and self-attention in Figure 2.

IV. Facial Data Representations

In this section, we describe different representations of video data. Our target is a method that can preserve important patterns towards classifying concentration levels while being light-weighted enough for real-time processing. Prior to the emergence of deep learning, analysis on facial data typically uses handcrafted features or uses shallow models such as Local Binary Patterns [21] to process inputs. Handcrafted features are heuristic of which quality largely depend on the domain knowledge of users. On the other hand, compared to deep learning, shallow models limited in terms of representation capability. Accordingly, we do not consider these methods in this work. Instead, we select the three representations for facial data: reshaped videos, landmarks, and embedding vectors.

A. Reshaped Video

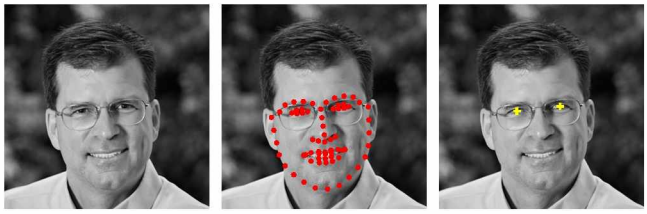
The recent advancement in CNN-type networks makes it possible to use videos as direct inputs to a model. Regardless, the high dimensionality and large size of video data remains a challenge especially in real-time applications. Furthermore, most deep learning models require inputs to have a uniform size, in this case, frame rates, frame width and height, and number of color channels. In reality however, students may be using different types of webcams that output videos at different resolutions and frame rates that may not satisfy the input size constraint. The students' relative distances and positions to the webcam is also an important factor impacting the data quality. To ensure quality, input video should be centered around the students' face and with as little irrelevant information as possible. Therefore, input videos need to undergo certain preprocessing steps such as re-sampling, cropping, resizing, etc. We call a video after these micro-processing steps a reshaped video.

B. Landmarks

Facial landmarks [13] refer to the main points on a human face such as eyebrows, eyes, nose, mouth, etc. Formally, in a facial image, each landmark is represented by two values that refer to its vertical and horizontal coordinates. Besides the main points, we also consider the locations of the left and right pupils as important features, since eye movement patterns may have a correlation to concentration levels. Overall, this representation consists of the coordinates for important points of a face, and is referred to as landmarks. Figure 3 shows an example of a facial image (original image obtained from public domain), as well as the facial landmarks (extracted using the DLib library [22]) and pupil coordinates (extracted using the DLib and OpenCV [23] libraries) of the image.

We expect the sequence of landmarks extracted from a video to be able to capture most essential features that are relevant to the task of analyzing concentration levels. By the described landmark extraction, a video of size $f \times w \times h \times c$ is reduced to a sequence of $f \times 2d_l$ elements where d_l is the number of landmarks. As a result, the dimensionality and size of input data are greatly reduced, so as the complexity of the machine learning model.

Authorized licensed use limited to: JIANGSU NORMAL UNIVERSITY. Downloaded on July 20,2023 at 16:08:41 UTC from IEEE Xplore. Restrictions apply.



(a) (b) (c)

Fig. 3: A facial image (a) and the extracted landmarks (b) and pupils(c) (facial image obtained from public domain)

C. Deep Embedding Vectors

An advantage of deep learning is the availability of *pre-trained* models which are deep neural networks that have been trained on certain datasets. Pre-trained networks can be applied to similar tasks to reduce the needs of computational power while utilizing more external knowledge (i.e., data that is external to the analysis). For this work, we consider two pre-trained deep face descriptors, Google FaceNet [15] and VGG-Face [16], both of which are very deep and complex CNN architectures trained on a very large amount of data on facial recognition. FaceNet and VGG-Face transform an input image into a 1D vector with dimensionality to be 128 and 2048, respectively. While the dimensionality of output vector from VGG-Face may seem high, it is much lower than that of an input image. For example, an RGB image at resolution 160x160 yields a vector of 76800 elements when flattened. Both descriptors can be adapted to videos by applying them to each individual frames with which the output for the input video is sequence of 1D embedding vectors.

V. Analysis Framework

We design our analysis framework to include three phases: 1) data preprocessing, 2) data transformation, and 3) decision making. The preprocessing phase performs initial cleaning and reshaping on the input videos. The processed data is then fed to the one of the descriptor discussed in Section IV to generate embedding vectors. Finally, a deep learning model is applied at the decision making phase to label the input videos.

A. Data Preprocessing

Raw videos that are output directly from the students' webcams may vary in frame rates and resolutions. Furthermore, the student distance to the webcam and their position in front of the lens also affect the input video quality. For example, if the student is too far from the webcam, their face will not fill the whole frame which may cause problems for the descriptor to extract features. Overall, we empirically define the preprocessing steps as follows.

1) Resampling the video. We first resample the input videos with the following two purposes; first, to unify the frame rates of all videos; and second, to reduce the size of the input data. As concentration states are relatively stable at small time windows, we consider a sampling rate of 10 - 15 frames per second (fps) to be the best fit.

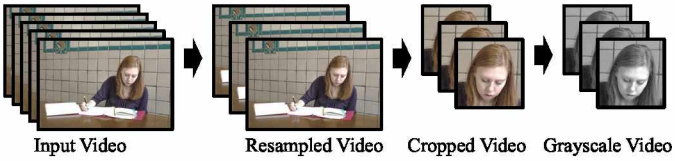


Fig. 4: facial video preprocessing steps (facial image obtained from public domain)

2) Locating the facial area. Depending on a student's position in front of the lens, their face locations may vary from frame to frame in the video. To obtain good quality of input data for the descriptors, we aim for the faces to be around the center of the videos. Therefore, in this step, we locate the facial area in each sampled frame. This can be done using a facial landmark model, or deep learning models such as MTCCN [24].
3) Cropping the videos. With the facial area determined from the previous step, we crop all frames in the resampled video so that the face is centered and takes the majority of each frame. This step also makes sure all input videos have the same width and height.
4) Converting to grayscale. Video data with three color channels is highly expensive, even after cropping, with respect to memory consumption and processing time. Given that color information is not essential for our analysis, the last step of preprocessing is to convert the cropped videos to grayscale to further reduce the input.

The overall preprocessing flow is illustrated in Figure 4.

B. Data Transformation

In this phase, we use one of the mentioned representation methods to transform processed video data into formats that are ready for modeling. As discussed, we consider three representations in this work, which are reshaped video, landmarks, and deep embedding vectors. Each type, except for reshaped video data, is generated by a pre-trained model, i.e., landmark vectors being output from a landmark model, and deep embedding vectors being generated by FaceNet or VGG-Face. It should be noted that, due to the computational resource limitation, we do not further finetune any descriptors with the new objective of classifying concentration levels.

C. Decision Making

The output from the descriptor is fed to a deep neural network in this phase to make decision. Depending on the input type, the modeling architecture may vary. For example, if the input is video data from the reshaping descriptor, the model architecture needs at least a block of 2DCNN or 3DCNN. On the other hand, 1D inputs like landmarks and deep embedding vectors can be fed directly to a sequential model like GRU or 1DCNN. We refer to the sub-component of the model that handles data from the descriptor as the input block. In all cases, the output from the input block is a sequence of 1D vectors, which is then fed to an attention layer and finally a fully connected layer to make decision. We design the

Authorized licensed use limited to: JIANGSU NORMAL UNIVERSITY. Downloaded on July 20,2023 at 16:08:41 UTC from IEEE Xplore. Restrictions apply.

TABLE I: Hyper-parameters of all tested architectures

Model	Descriptor	Input Dimension	Input Block	Output Dimension
FN+1DCNN	FaceNet	128	1DCNN (128,3) + 1DCNN (128,3)	512
FN+GRU	FaceNet	128	GRU (512)	512
FN+biGRU	FaceNet	128	biGRU (512)	512
LM+1DCNN	Landmark	140	1DCNN (128,3) + 1DCNN (128,3)	512
LM+GRU	Landmark	140	GRU(140)	140
LM+biGRU	Landmark	140	biGRU(140)	140
VD+2DCNN+1DCNN	Reshaping	160x160	4x2DCNN(8,4,4) + 1DCNN(128,3) + 1DCNN(64,3)	64
VD+2DCNN+GRU	Reshaping	160x160	4x2DCNN(8,4,4) + GRU(64)	64
VD+2DCNN+biGRU	Reshaping	160x160	4x2DCNN(8,4,4) + biGRU(64)	64
VD+3DCNN+GRU	Reshaping	160x160	3x3DCNN(8,2,4,4) + GRU(64)	64
VGG+1DCNN	VGG-Face	2048	1DCNN (512,3) + 1DCNN (256,3)	512
VGG+GRU	VGG-Face	2048	GRU(512)	512
VGG+biGRU	VGG-Face	2048	biGRU(512)	512

task of detecting concentration levels from video as a binary classification task. Accordingly, the output layer of the model is a Sigmoid layer.

Except for the preprocessing phase, the other two have different model options that form neural network architectures. In the next section, we provide an in-depth evaluation on the performances of these architectures.

VI. Experiment Study

All experiments are done in Python 3.6. We implement the landmark descriptor using the DLib library [22]) and OpenCV [23]; the pre-trained version of FaceNet is available at [25]; the pre-trained VGG-Face is available at [26]. We also want to acknowledge the use of the Python packages Numpy [27], Matplotlib [28], and Tensorflow [29]. All experiments are done on a workstation with Intel(R) Core(TM) i9-9940X CPU, 128GB of RAM, and four Nvidia Quadro P6000 graphic cards.

A. Dataset

We perform experiments on a dataset that consists of 25 students. Each student has ten one-minute videos in which they were concentrating while, and ten one-minute videos in which they were distracted. In both cases, the students are following an online lecture. We do not restrict the format of the recorded videos to simulate real world situations. Overall, the dataset has 250 videos labeled "concentrate" and 250 videos labeled "distract".

B. Analysis

We analyze the videos following the proposed framework. The videos are preprocessed to be 15fps in grayscale. The frame resolution is either 160x160 for reshaping, landmark, or FaceNet descriptors, or 224x224 for VGG-Face descriptor. The videos are further split into 30-frame segments; segments from each video have the same labels as the original video (i.e., "concentrate" or "distract"). At the end, we obtain a processed dataset of 12,868 samples which is then split into 70% training, 15% validation, and 15% testing.

In terms of modeling architectures, we test a total of 13 designs with the finetunable components being descriptor, input block, and output dimension. The hyper-parameters (i.e. number of layers in a block, number of neurons each layer, etc.) are finetuned with the training and validation data. The final design of each architecture are showed in Table I. The notations we use for input blocks are as follows:

- $1DCNN(k, l)$ represents one 1DCNN layer of k filters each of which has size l
- $2DCNN(k, w, h)$ represents one 1DCNN layer of k filters each of which has size $w \times h$
- $3DCNN(k, l, w, h)$ represents one 3DCNN layer of k filters each of which has size $l \times w \times h$
- $GRU(n)$ represents one GRU layer that has output of size n
- $biGRU(n)$ represents one bi-directional GRU layer that has output of size n. In terms of implementation, we use two GRU layers of size $n/2$ that go forward and backward on the input sequence. Their outputs are then concatenated by time steps.

TABLE II: Model performances on predicting concentration levels of students during online learning

Model	Training	Validation	Testing
FN+1DCNN	99.24 ± 0.62	73.53 ± 2.72	70.16 ± 4.04
FN+GRU	99.14 ± 0.4	74.02 ± 3.49	72.66 ± 4.75
FN+biGRU	98.97 ± 0.53	74.34 ± 3.16	72.51 ± 4.77
LM+1DCNN	98.96 ± 0.27	91.04 ± 3.12	**91.14 ± 2.39**
LM+GRU	97.13 ± 0.37	89.6 ± 3.48	88.92 ± 3.55
LM+biGRU	97.86 ± 0.77	89.09 ± 3.76	88.82 ± 3.08
VD+2DCNN+1DCNN	99.65 ± 0.19	75.34 ± 4.02	73.56 ± 2.47
VD+2DCNN+GRU	99.5 ± 0.15	75.93 ± 5.06	76.11 ± 2.1
VD+2DCNN+biGRU	99.43 ± 0.14	76.38 ± 4.12	75.6 ± 4.07
VD+3DCNN+GRU	99.03 ± 0.26	75.3 ± 4.5	74.12 ± 4.97
VGG+1DCNN	99.51 ± 0.2	**91.97 ± 4.13**	*90.24 ± 2.17*
VGG+GRU	99.34 ± 0.33	*91.69 ± 4.01*	90.12 ± 1.92
VGG+biGRU	99.38 ± 0.14	91.45 ± 3.4	89.94 ± 2.33

Each architecture is tested on ten random splits of the data. Since the labels are balanced in this data, we use accuracy rate to evaluate the models. The models are trained using Stochastic Gradient Descent [30] with learning rates starting at 10^{-3} and decaying by a factor of 10 whenever training loss fluctuates heavily. Training stops when validation accuracy does not improve after 20 epochs. The models' performances are reported in Table II.

C. Discussion

As can be seen from Table II, architectures using Landmark and VGG-Face descriptors outperform the reshaping and

Authorized licensed use limited to: JIANGSU NORMAL UNIVERSITY. Downloaded on July 20,2023 at 16:08:41 UTC from IEEE Xplore. Restrictions apply.

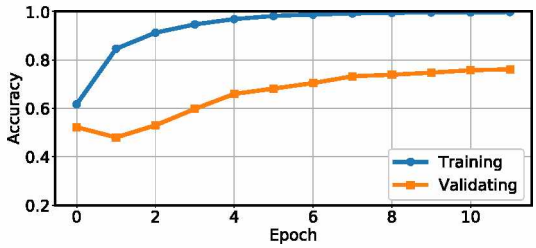


Fig. 5: An example of training and validation accuracy during training

FaceNet descriptor by a large margin. Models from the former two achieve around 90% accuracy in validation and testing whereas that of models using FaceNet are at approximately 73% and reshaping are at 75%. It is rather surprising that FaceNet descriptor gets the lowest performances. We suspect that it could be due to the low output dimensionality coupled with the training objective of facial identification. Specifically, the output dimensionality of FaceNet is too low for the model to generalize to new tasks.

Another observation that can be made from Table II is that *all* models are overfitting to some extent. Except for those using landmark descriptor, the remaining models' training accuracy rates are all around 99%. Their validation and testing accuracy rates then drop quite significantly. The early stopping strategy does not prevent this situation due to the model overfitting the training data. However, continuing to train the data set has the potential to improve performance in the validation process. This situation is illustrated in Figure 5 which shows the training accuracy and validation accuracy of the first 12 epochs in training FN+GRU. We can see that the accuracy in both sets start around 60% in the first epoch, however, training accuracy quickly raises to over 90% on the fifth epoch while the validation accuracy is still at 65%. From the seventh epoch, training accuracy is very close to 100%, however, we can still see improvement in validation accuracy. This pattern persists in all models even with fine-tuning. Therefore, we speculate that this could be an issue with this specific dataset and the given task.

We further examine the stability of the models, i.e. how their performances vary in different training and testing data. Figure 6 shows the box plots of all models' accuracy in the validation and testing data. We can observe that models using landmark and VGG-Face descriptors achieve the most stability, with VGG models being slightly better. Models using reshaping and FaceNet descriptors have their accuracy vary significantly with ranges up to over 15%.

On individual models' performances, Table II and Figure 6 both suggest the best models are either LM+1DCNN or VGG+1DCNN, with LM+1DCNN achieving the highest testing performance at 91.14% on average, and VGG+1DCNN the second highest testing accuracy at 90.24%. The other two VGG models get relatively close performance to the best two, while the rest of the landmark models' performances drop slightly more.

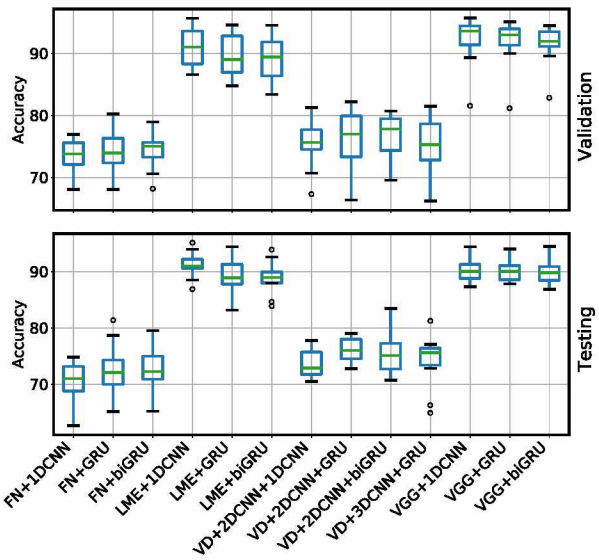


Fig. 6: Box-plot of model accuracy rates in validation and testing

As the two champion models are those using landmark and VGG-Face descriptor with 1DCNN input block, we analyze their time performance to determine the better fit for real-time detection applications. Specifically, we investigate their execution time in three phases: processing, training, and predicting. Table III shows the two models' average executing time in each cases in milliseconds. Processing time and predicting time are averaged for one single 30-frame segment from ten runs, whereas training time is for 20 epochs on a training set of approximately 10,000 samples (each of which is also a 30-frame segment). Despite being a much larger model, VGG-Face still achieves significantly shorter processing time. This is due to the fact that it is a deep learning model and can utilize the computational power of a graphic card. Although the landmark models are smaller, since being executed using CPU, they are about four times slower. In terms of predicting time, models using landmark embedding is almost ten times faster since it has a much simpler architecture (input dimension of 140 versus that of 2048 in the VGG model). However, comparing to processing time, predicting times in both models are negligible. The complexity of VGG models is also reflected in terms of in training time, given that it takes almost three times longer than the landmark model to train 20 epochs of the same data size. This fact may be note-worthy when training these models on a much larger dataset since training time does not increase linearly with data size. Nevertheless, during production phases in which the models only need to process and predict the data, both models are able to process over 30 frames per second, which is fast enough for real-time applications.

The accuracy and time analysis allow us to conclude the best-performed architectures are landmark and VGG descriptors with 1DCNN input blocks. Both achieve over 90% testing accuracy while maintaining practically acceptable execution

Authorized licensed use limited to: JIANGSU NORMAL UNIVERSITY. Downloaded on July 20,2023 at 16:08:41 UTC from IEEE Xplore. Restrictions apply.

TABLE III: Average execution times of the two best performed models (ms)

Model	Processing	Training	Predicting
VGG+1DCNN	73	3045	0.253
LM+1DCNN	305	1080	0.027

efficiencies. We also find it interesting that FaceNet turns out to be the descriptor that yields the lowest accuracy by a large margin. We suspect that FaceNet's architecture and training objective do not generalize well to this new task pf analyzing concentration levels based on facial videos.

VII. Conclusion

In this paper, we investigate the task of detecting students' concentration levels during online study using video data that is output by webcams. We formulate the problem as binary classification in which the input is a short facial video segment and the output is whether the student is "concentrate" or "distract". Our goal is to empirically design an architecture that is balanced between prediction accuracy and execution time so that it is applicable for real-time settings.

We present an in-depth study in which we consider four types of representations for input video coupled with various deep learning architectures. The data representations include 4D video, landmarks, embedding vectors from FaceNet, and embedding vectors from VGG-Face. The neural network architectures include a mixture of 1DCNN, 2DCNN, 3DCNN, GRU, bi-directional GRU, and Attention. Our experimental study showed two architectures that are able achieve over 91% in terms of accuracy while maintaining reasonable computational efficiency for real-time application. We believe these frameworks may bridge the areas of deep learning and online education through which students' learning is better supported.

For future work, we aim to expand the scale of the experiments. Other tasks we are planning include exploring and improving descriptors, extending the concentration levels, and enhancing the decision making models.

References

[1] A. Moran, "Concentration: Attention and performance," *The Oxford handbook of sport and performance psychology*, pp. 117–130, 2012.

[2] J. Stuart and R. D. Rutherford, "Medical student concentration during lectures," *The lancet*, vol. 312, no. 8088, pp. 514–516, 1978.

[3] D. B. Lindsley, "Psychological phenomena and the electroencephalogram." *Electroencephalography & Clinical Neurophysiology*, 1952.

[4] S. C. Bunce, M. Izzetoglu, K. Izzetoglu, B. Onaral, and K. Pourrezaei, "Functional near-infrared spectroscopy," *IEEE engineering in medicine and biology magazine*, vol. 25, no. 4, pp. 54–62, 2006.

[5] Y. Fan, V. Li, and J. C. Lam, "Facial expression recognition with deeply-supervised attention network," *IEEE transactions on affective computing*, 2020.

[6] M. Z. Uddin, M. M. Hassan, A. Almogren, M. Zuair, G. Fortino, and J. Torresen, "A facial expression recognition system using robust face features from depth videos and deep learning," *Computers & Electrical Engineering*, vol. 63, pp. 114–125, 2017.

[7] S. Zhang, X. Pan, Y. Cui, X. Zhao, and L. Liu, "Learning affective video features for facial expression recognition via hybrid deep learning," *IEEE Access*, vol. 7, pp. 32 297–32 304, 2019.

[8] T. Zhang, "Facial expression recognition based on deep learning: a survey," in *International conference on intelligent and interactive systems and applications*. Springer, 2017, pp. 345–352.

[9] C.-M. Kuo, S.-H. Lai, and M. Sarkis, "A compact deep learning model for robust facial expression recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 2121–2129.

[10] Y. LeCun, Y. Bengio *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[11] S. Li and W. Deng, "Deep facial expression recognition: A survey," *IEEE transactions on affective computing*, 2020.

[12] W.-H. Yun, D. Lee, C. Park, J. Kim, and J. Kim, "Automatic recognition of children engagement from facial video using convolutional neural networks," *IEEE Transactions on Affective Computing*, vol. 11, no. 4, pp. 696–707, 2018.

[13] M. Koestinger, P. Wohlhart, P. M. Roth, and H. Bischof, "Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization," in *2011 IEEE international conference on computer vision workshops (ICCV workshops)*. IEEE, 2011, pp. 2144–2151.

[14] L. R. Young and D. Sheena, "Survey of eye movement recording methods," *Behavior research methods & instrumentation*, vol. 7, no. 5, pp. 397–429, 1975.

[15] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[16] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," 2015.

[17] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[19] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[20] Q. Tao, F. Liu, Y. Li, and D. Sidorov, "Air pollution forecasting using a deep learning model based on 1d convnets and bidirectional gru," *IEEE access*, vol. 7, pp. 76 690–76 698, 2019.

[21] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face recognition with local binary patterns," in *European conference on computer vision*. Springer, 2004, pp. 469–481.

[22] D. E. King, "Dlib-ml: A machine learning toolkit," *The Journal of Machine Learning Research*, vol. 10, pp. 1755–1758, 2009.

[23] G. Bradski and A. Kaehler, "Opencv," *Dr. Dobb's journal of software tools*, vol. 3, 2000.

[24] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016.

[25] H. Taniai, "Keras-facenet," https://github.com/nyoki-mtl/keras-facenet, 2017.

[26] R. Malli, "Keras-vggface," https://github.com/rcmalli/keras-vggface, 2017.

[27] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2

[28] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[29] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[30] L. Bottou, "Stochastic gradient descent tricks," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 421–436.

Authorized licensed use limited to: JIANGSU NORMAL UNIVERSITY. Downloaded on July 20,2023 at 16:08:41 UTC from IEEE Xplore. Restrictions apply.