Automated Reading Detection in an Online Exam

https://doi.org/10.3991/ijet.v17i22.33277

Bakhitzhan Kadyrov, Shirali Kadyrov(✉), Alfira Makhmutova
Suleyman Demirel University, Kaskelen, Kazakhstan
shirali.kadyrov@sdu.edu.kz

In memory of Zaitunyam Sopekova

Abstract—In this article we study a deep learning-based reading detection problem in an online exam proctoring. Pandemia-related restrictions and lockdowns lead many educational institutions to go online learning environment. It brought the exam integrity challenge to an online test-taking process. While various commercial exam proctoring solutions were developed, the online proctoring challenge is far from being fully addressed. This article is devoted to making a contribution to the exam proctoring system by proposing an automated test-taker reading detection method. To this end, we obtain our own dataset of short video clips that resemble a real online examination environment and different video augmentation methods utilized to increase the training dataset. Two different deep learning techniques are adapted for training. The experiments show quite satisfactory results with model accuracy varying from 98.46% to 100%. The findings of the article can help educational institutions to improve their online exam proctoring solutions, especially in language speaking tests.

Keywords—exam proctoring, reading detection, video recognition, computer vision, deep learning

1 Introduction

It has been over two years since the start of the COVID-19 pandemic. The pandemic affected every aspect of our society and the education system is no exception [1]–[5]. Especially in the early stages of the disease, many countries implemented social distancing measures to combat the virus. In particular, school pupils and university students were required to stay at home and the instructors were asked to conduct online classes [6]. Online education is not a new area. Indeed, online learning platforms such as Coursera, Udemy, EdX, and learning management systems such as Moodle, Blackboard, Google Classroom, and online meeting solutions such as Webex, Zoom, Google meet, and Skype all existed before the pandemia [7]. However, with the start of the pandemic, the need for online learning methods and tools has increased drastically.

Online education has many benefits due to its flexible nature. It enables both instructor and student to decide on their learning environment. No one is required to travel for learning or teaching reducing the cost of learning and minimizing the wasted time.

Nothing comes without a price. Despite the above-listed advantages of online education, it also brings various issues together. The dishonesty in the examinations is among the biggest concerns in online learning [8] and the automated proctoring systems need to be improved to prevent possible examination malpractices. There are various partial solutions provided to prevent any dishonesty in examinations [9], [10]. Moodle's Safe Exam Browser (SEB) is an open access browser that prevents the test taker from using other windows or tabs in a computer-based exam. Some paid solutions such as Examus and ProctorEdu provide more comprehensive tools to proctor the online exams. These solutions often include metrics such as identity verification, screen casting, webcam recording, speaking or noise detection, second person detection, second monitor detection, and so on.

While the above-mentioned online proctoring systems have various functionalities, they still fall short in completely preventing cheating. This necessitates a need for further research to improve the current state-of-the-art methods. In particular, the extensive search of the literature revealed that almost no research has been conducted to detect automated reading detection. The purpose of this article is to develop a new model that can determine whether or not a test taker is reading from a source. This is particularly important during an online speaking examination, where the test taker is expected to speak on a subject without reading. Furthermore, in any online oral examination, a student may be answering the questions from an unallowed source on the computer screen or from other nearby sources.

We remark here that in general reading detection is a challenging task even for humans as often eye movements are involuntary [11]. Compared to other machine learning tasks such as natural language processing and face recognition, action recognition is much more challenging as the automated system needs to take into account both spatial and temporal features. Within action recognition tasks, clearly, reading detection is a harder task requiring a very focused feature extraction from the eye region of the human. Moreover, finding a dataset suitable for training is difficult.

In the next section, we review the literature in a related field. Section 3 provides information on the dataset and methodologies to run the experiments. In section 4, we report the findings of the experiments. The paper ends with a discussion of results and conclusion.

2 Literature review

2.1 Human action recognition

In the last decade, advancements in artificial intelligence research showed state-of-the-art methods exceeding human-level accuracy in various data science tasks including natural language processing, sound recognition, and computer vision. Human action recognition is one of the computer-vision tasks that aims to detect what type of activity is done based on a video clip. There are many real-world applications of action recognition such as human-robot interaction, entertainment, video surveillance, and autonomous driving vehicles [12]. While temporal feature extraction is crucial in handling time series data and text classification, for image-related tasks, spatial features are needed. On the other hand, one needs to take into account both temporal and spatial features for action recognition. In this subsection, we review deep learning-based approaches to the action recognition problem.

Inspired by 2D convolutional neural networks (CNN), in [13], the authors developed a novel 3D CNN for the task of action recognition. The idea was to realize a video as a three-dimensional object where time is considered a third dimension. They tested the performance of their model on KTH [14] and TRECVID 2008 dataset. On KTH dataset, the 3D CNN model slightly underperformed (90.7%) compared to HMAX (91.7) [15]. On the other hand, for TRECVID dataset, on average it overperformed previous approaches proposed in [16], [17].

In 2015 authors in [18] proposed Convolutional 3D architecture with a linear SVM classifier improving 3D CNN models. The performance of the model was tested using UCF101 dataset [19] and proved to outperform previous results including iDT [20] and Imagenet [21]. As 3D CNN-based models often consider a few frames as their third dimension, their performance is more likely to be lower for actions needing longer time intervals [12].

Other kinds of models to address action recognition are called multi-stream networks. Two-stream CNN model was proposed in [22] where one stream captures spatial information and the second one recognizes temporal features. The model showed comparable performance on UCF-101 [19] and HMDB-51 [23].

Based on [22] a Temporal Sampling Network was proposed in [24] robust to action recognition with longer videos. To enable quick and effective learning using the entire action video, it combines a sparse temporal sampling method and video-level supervision. Temporal segment networks work using a sequence of tiny snippets sparsely sampled from the entire video rather than single frames or frame stacks. Each snippet in this sequence will generate its own tentative action class prediction. The video-level prediction will then be formed from a consensus among the snippets. The proposed model reached 69.4% accuracy for HMDB-51 and 94.2% accuracy for UCF101.

Another common approach to learning both spatial and temporal information is to consider hybrid models involving both CNN and recurrent layers such as Long-Short Term memory (LSTM) or Gated Recurrent Units (GRU). Donahue et al proposed Long-term Recurrent Convolutional Networks [25] with 82.66% accuracy on UCF-101 dataset. Another hybrid model was proposed in [26] capable of handling action videos with a duration as long as 2 minutes. The experimental results on the proposed CNN-LSTM model with 30-frame unroll showed state-of-the-art results on UCF-101 with 88.6% accuracy.

2.2 Reading detection

In an early work [11] the authors study the reading detection problem with their own dataset of four participants. The proposed system has three main steps, namely, quantized eye movement representations, pooled evidence-based identification, and mode switching. A video camera with an array of LEDs that reflect infrared light into the participant’s eye was used to determine gaze direction. The eye tracking software was calibrated for each participant to ensure accurate tracking over the whole screen.

While the performance of the model is close to 100% it requires infrared cameras and requires the participant's head kept fixed.

Later, Keat et al. [27] proposed an improved algorithm for determining whether or not a person is reading. The experiment was conducted on a regular video camera with 10 participants in the experiment. Their approach reached 85% accuracy.

In 2008, Bulling et al [28] proposed a different approach using wearable electrooculography. Eight participants were involved in a 6-hour data collection, where they were allowed to read freely without any restrictions. Using string matching, their method reached 80.2% recognition accuracy.

In a recent work [29], authors studied a commercial electrooculography-based reading activity detection problem in a general environment. 980 hours long data from 7 subjects was used in the experiment with three different approaches, namely, Support Vector Machine (SVM), CNN, and LSTM-based approaches. In a natural environment, LSTM-approach overperformed the others with 73.8% accuracy, while SVM-approach was best with 92.2% accuracy in a controlled environment.

The COVID-19 pandemic period not only shaped the educational sphere but also influenced the development of online education. However, the question of cheating did not disappear, but it became a hot topic because it affects the quality of education. As a result of the literature review, this work is guided by the following research question: Can deep learning models help us to detect reading during proctoring in online exams?

3 Methodology

In this section we provide detailed information about the dataset used in the experiment and the deep learning architectures that will be used.

3.1 Dataset and preprocessing

Due to a lack of dataset availability, we developed our own. The participants were asked to record a short video where they pretend to be in an online examination environment and another short video where they were reading from a screen or a nearby source. There were no restrictions on hardware and the participants used webcams of their own laptops. As for the physical setup, they were sitting close to the computer as it is done in a regular online exam, see the first row in Figure 1. Each video lasted about one minute. There were 10 individuals who participated with total 22.4 minutes of video recordings. Additionally, we exported parts of 25 videos from Youtube with creative commons license for total 14.6 minutes, see the second row in Figure 1. All videos saved with 720p at 30 fps having a frame size of 1280 × 720. Videos were labeled as 'reading' vs. 'no reading'. In total there are 18.2 minutes of videos where participants are reading from a screen or nearby source and in the remaining 18.8-minute videos the participants were not involved in reading. Since both labels are represented by roughly the same amount of video recordings, this will help us create a balanced dataset.

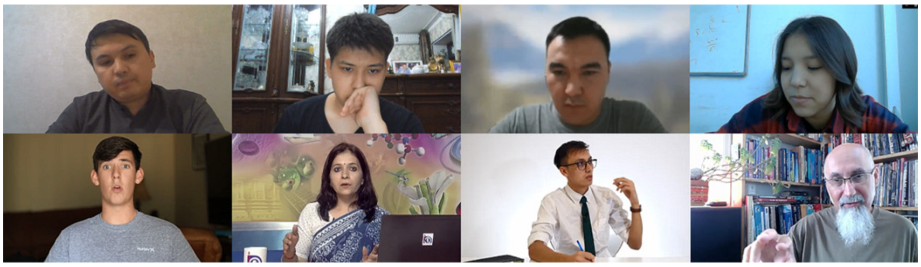


Fig. 1. Screenshots from the dataset of videos

Once the dataset is collected, next we split it as a train set and a test set. To this end, we first create 5-second and 7-second videos by cutting each video in the dataset with 50% overlaps. Then, we randomly allocate 10% of these short videos into the test set, and the remaining 90% is allocated for augmentation and training purposes. In this way, we obtained 130 short videos for testing the performance of the models and 1164 videos for augmentation and training. 66 out of 130 videos in the test set are labeled as 'reading' while in the train set there are 583 videos labeled as 'reading'.

With the aim to improve the classification performance and avoid possible overfitting, two data augmentation techniques, namely cropping and increasing brightness, were applied to the train set with 1164 videos while the test set has not modified. For cropping, we clearly need to preserve the person appearing in the videos, especially his/her eye region. Similarly, when brightness is increased, we should still be able to see and follow the eye movements of the person. One copy of 1164 train videos is generated by applying random cropping according to a uniform distribution. More specifically, we take random numbers x, y, z, and w from uniform distribution in the intervals [0, 0.3], [0, 0.1], [0.8, 1], and [0.8, 1] respectively.

We recall that the original videos had a height of 1280 and a width of 729 pixels. Then, a cropped video was obtained from a given video in the train set by removing $1280x$ portion from the left, $729y$ portion from the top, $1280z$ portion from the right, and $729w$ portion from the bottom. To apply cropping to the next video, the numbers x, y, z, and w were generated again. Another copy of 1164 train videos is generated by increasing the brightness to a random number in the interval [1.5, 2.0]. As a result of augmentation techniques, the train size is increased to 3492 short video clips.

3.2 Proposed deep learning models

To carry out the experiments, we consider two different deep learning architectures and compare their performances implemented in Keras. First, we formulate the problem statement in mathematical terms. A video consists of a sequence of images known as video frames. Depending on the world region a standard video may consist of 24–30 frames per second (fps). At any given time, the frame X is regarded as a stack of P number of $M \times N$ matrices, where entries of the matrices emphasize the intensity of the image for a given measurement. For example, for a 10 × 20 pixel RGB image, $P = 3$, $M = 10$, and $N = 20$. Thus, a video is a time series $X_1, X_2, \ldots, X_t$ belonging to the vector space $\mathbb{R}^{P \times M \times N}$.

Now, the reading recognition problem statement amounts to finding

$$\arg\max_{x \in L} Pr(x \mid X_1, X_2, \ldots, X_t)$$

Where *Pr* stands for the probability function and *L* consists of the labels "reading" or "no reading".

1) Convolutional Long Short-Term Memory architecture. Convolutional neural networks were developed to extract spatial features of the data [30], [31]. In particular, they perform well dealing with tasks related to still images in Computer Vision [32] including object recognition and image classification. Despite various cutting-edge CNN-based models developed in CV, they are not well suited to problems of temporal nature. Examples include time series analysis and natural language processing tasks. To address such sequence models, Recurrent neural network architectures were developed including Long Short-Term Memory (LSTM) [33], BLSTM [34], Gated Recurrent Networks (GRU) [35], Encoder-decoder models [36], and so on. There are situations when the given data has both spatial and temporal characters. Video-based action recognition, weather forecasting, and spectrogram-based speech recognition are some of examples of spatiotemporal tasks. In addressing spatiotemporal problems, it is better to combine the CNN-based models together with RNN-based models. Shi et al. [37] developed a novel Convolutional LSTM to deal with precipitation nowcasting to predict the upcoming rainfall intensity for a short period of time. Their proposed model is proven to outperform fully connected LSTM developed by [38].

Our first model to train the dataset will be Convolutional LSTM (ConvLSTM). To make the work self-contained we provide details of ConvLSTM adopting the notations from [37], [39]. To this end, let σ denote the sigmoid activation given by

$$\sigma(x) = \frac{1}{1+e^{-x}},$$

and *i*, *f*, *o*, *c* are input gate, forget gate, output gate, cell, and cell input activation vectors, respectively. The *W*, *b* represent the weight matrices and the bias, respectively.

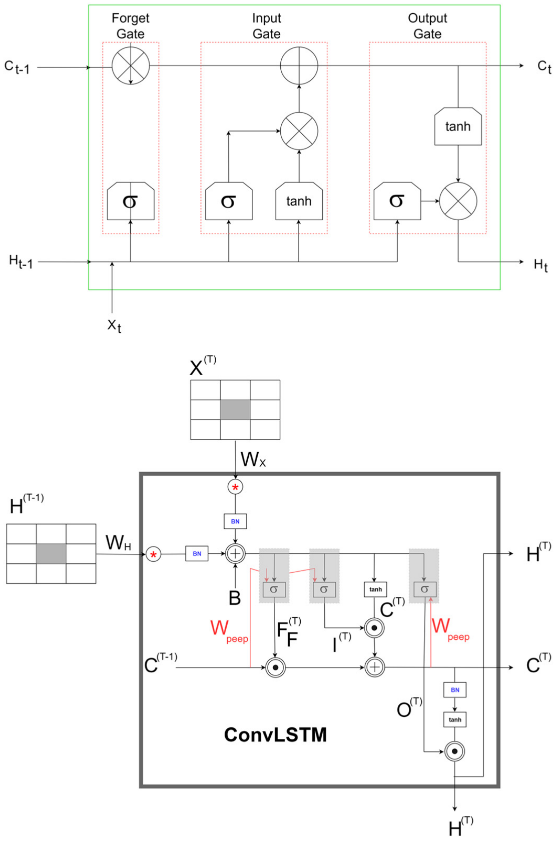


Fig. 2. LSTM vs ConvLSTM Cell architecture

The main difference between fully connected LSTM [38] and ConvLSTM [37] is that the latter considers all inputs X_j'*s*, cell outputs C_j'*s*, hidden states H_j'*s*, and gates i_j, f_j, o_j to be 3D tensors. The assignments governing the ConvLSTM Cell architecture are then given by the following equations:

$$i_t = \sigma(W_{xi} * X_t + W_{hi} * H_{t-1} + W_{ci} \circ C_{t-1} + b_i)$$

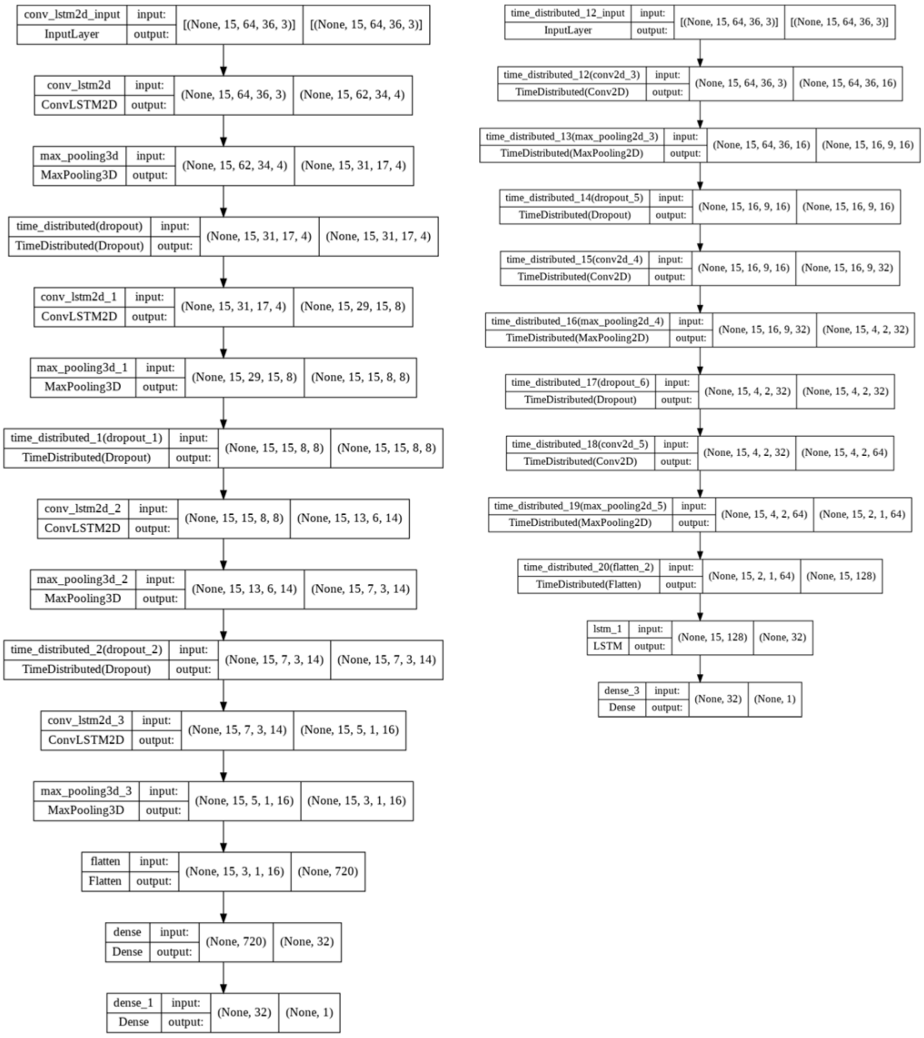
$$f_t = \sigma(W_{xf} * X_t + W_{hf} * H_{t-1} + W_{cf} \circ C_{t-1} + b_f)$$

$$C_t = f_t \circ C_{t-1} + i_t \circ \tanh(W_{xc} * X_t + W_{hc} * H_{t-1} + b_c)$$

$$o_t = \sigma(W_{xo} * X_t + W_{ho} * H_{t-1} + W_{co} \circ C_t + b_o)$$

$$H_t = o_t \circ \tanh(C_t)$$

Where the operations * and ◦ denote the convolution and Hadamard product, respectively. A summary of the ConvLSTM model is depicted in Figure 3a. The model is compiled with binary cross entropy loss using the Adam optimizer.


conv_lstm2d_input InputLayer input: output: [(None, 15, 64, 36, 3)] [(None, 15, 64, 36, 3)]
conv_lstm2d ConvLSTM2D input: output: (None, 15, 64, 36, 3) (None, 15, 62, 34, 4)
max_pooling3d MaxPooling3D input: output: (None, 15, 62, 34, 4) (None, 15, 31, 17, 4)
time_distributed(dropout) TimeDistributed(Dropout) input: output: (None, 15, 31, 17, 4) (None, 15, 31, 17, 4)
conv_lstm2d_1 ConvLSTM2D input: output: (None, 15, 31, 17, 4) (None, 15, 29, 15, 8)
max_pooling3d_1 MaxPooling3D input: output: (None, 15, 29, 15, 8) (None, 15, 15, 8, 8)
time_distributed_1(dropout_1) TimeDistributed(Dropout) input: output: (None, 15, 15, 8, 8) (None, 15, 15, 8, 8)
conv_lstm2d_2 ConvLSTM2D input: output: (None, 15, 15, 8, 8) (None, 15, 13, 6, 14)
max_pooling3d_2 MaxPooling3D input: output: (None, 15, 13, 6, 14) (None, 15, 7, 3, 14)
time_distributed_2(dropout_2) TimeDistributed(Dropout) input: output: (None, 15, 7, 3, 14) (None, 15, 7, 3, 14)
conv_lstm2d_3 ConvLSTM2D input: output: (None, 15, 7, 3, 14) (None, 15, 5, 1, 16)
max_pooling3d_3 MaxPooling3D input: output: (None, 15, 5, 1, 16) (None, 15, 3, 1, 16)
flatten Flatten input: output: (None, 15, 3, 1, 16) (None, 720)
dense Dense input: output: (None, 720) (None, 32)
dense_1 Dense input: output: (None, 32) (None, 1)
time_distributed_12_input InputLayer input: output: [(None, 15, 64, 36, 3)] [(None, 15, 64, 36, 3)]
time_distributed_12(conv2d_3) TimeDistributed(Conv2D) input: output: (None, 15, 64, 36, 3) (None, 15, 64, 36, 16)
time_distributed_13(max_pooling2d_3) TimeDistributed(MaxPooling2D) input: output: (None, 15, 64, 36, 16) (None, 15, 16, 9, 16)
time_distributed_14(dropout_5) TimeDistributed(Dropout) input: output: (None, 15, 16, 9, 16) (None, 15, 16, 9, 16)
time_distributed_15(conv2d_4) TimeDistributed(Conv2D) input: output: (None, 15, 16, 9, 16) (None, 15, 16, 9, 32)
time_distributed_16(max_pooling2d_4) TimeDistributed(MaxPooling2D) input: output: (None, 15, 16, 9, 32) (None, 15, 4, 2, 32)
time_distributed_17(dropout_6) TimeDistributed(Dropout) input: output: (None, 15, 4, 2, 32) (None, 15, 4, 2, 32)
time_distributed_18(conv2d_5) TimeDistributed(Conv2D) input: output: (None, 15, 4, 2, 32) (None, 15, 4, 2, 64)
time_distributed_19(max_pooling2d_5) TimeDistributed(MaxPooling2D) input: output: (None, 15, 4, 2, 64) (None, 15, 2, 1, 64)
time_distributed_20(flatten_2) TimeDistributed(Flatten) input: output: (None, 15, 2, 1, 64) (None, 15, 128)
lstm_1 LSTM input: output: (None, 15, 128) (None, 32)
dense_3 Dense input: output: (None, 32) (None, 1)

(a) ConvLSTM **(b) LRCN**

Fig. 3. Deep learning architectures

2) Long-term Recurrent Convolutional Networks. We now introduce our second architecture based on [25]. The convLSTM-based model discussed above uses several stacks of convLSTM layers added in sequential order where one convLSTM layer includes both CNN and LSTM functionality. On the other hand, Long-term Recurrent Convolutional Networks (LRCN) proposed by Donahue [25] uses stacks of CNN layers flowed by LSTM layers, see Figure 4. This approach proved to outperform previous results [22], [40].

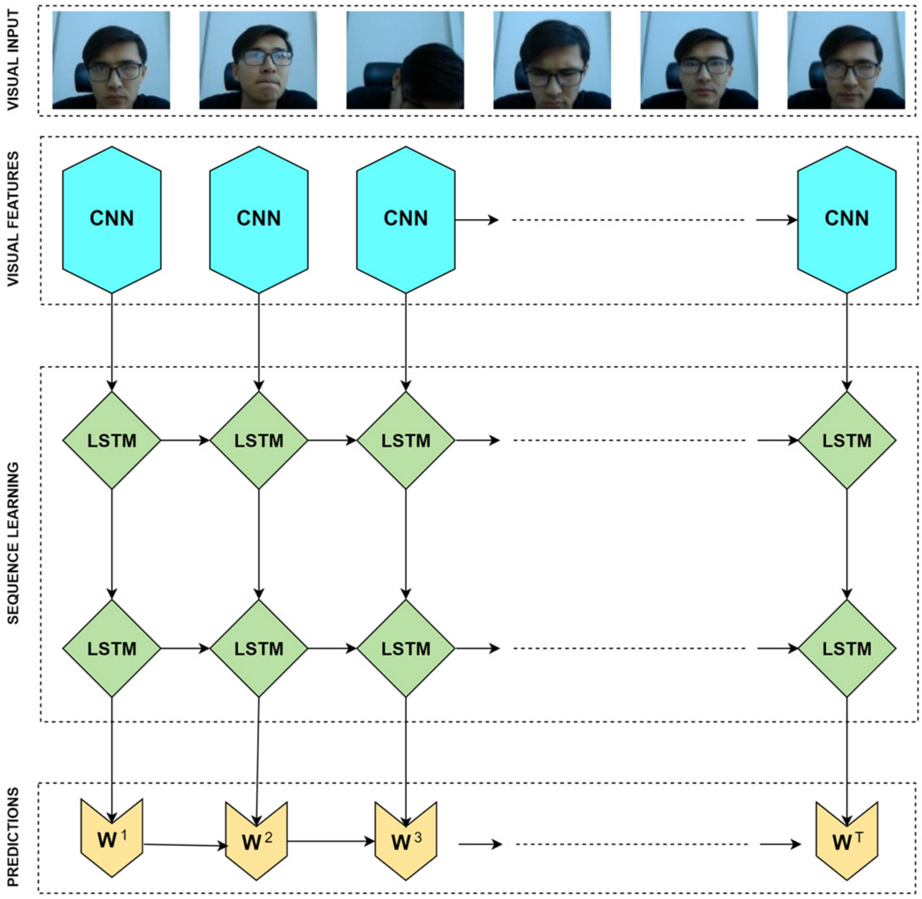


Fig. 4. Long-term recurrent convolutional networks

The compiled Long-term Recurrent Convolutional Networks (LRCN) is given in Figure 3b.

In LRCN each video frame X_i belonging to the vector space $\mathbb{R}^{P \times M \times N}$ is passed through CNN layers to produce a fixed-size feature vector representation. These sequences of feature vectors are then fed into LSTM layers followed by the sigmoid activation in the output to predict the class the video belong to.

3.3 Training process and accuracy metrics

To input the train set into the models for training, each short video is converted into (15, 64, 36, 3)-shaped arrays of 15 RGB frames with sizes 64 × 36. This makes the area of the frames twenty times smaller than the original dataset which in turn increases the speed of training and recognition and avoids using excess memory. Depending on the length of the video we have 150 or 210 frames in each short video and those sequence of 15 representative frames are extracted so that the duration between any two consecutive frames have equal duration. Then, the arrays are normalized through diving each entry by 255.

It is customary to allocate a small portion of the train set to validation in order to fine-tune the model parameters and monitor the accuracy. To this end, we allocated 5% of the augmented train set to validation. Moreover, we considered two different scenarios, in one occasion we only used the original 1164 videos for training and validation. In the second experiment, we used all 3492 videos.

Once the models are trained, the performance of the binary classifiers is measured with performance metrics accuracy, precision, recall, and F-score (F_1) using a test set with 130 videos. We recall that the accuracy is obtained by taking the ratio of correctly classified labels to the number of all labels, that is 130. Precision is the ratio of correctly identified reading cases to the number of all cases predicted as reading. On the other hand, recall is the ratio of correctly identified reading cases to all true reading cases. Finally, F-score is the harmonic mean of precision and recall, given by the formula

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall}.$$

Moreover, confusion matrices are reported to better visualize the mismatches between true labels and model predicted labels.

4 Results of experiments

In this section we report the finding of our experiments. We have two different models ConvLSTM and LRCN and each model was trained twice, first with original train data of 1164 short videos and on the second occasion with 3492 short videos obtained through augmentation. Batch sizes of 64 and 4 were employed to train and validation sets, respectively. The experiments show that ConvLSTM is performing well within 30 epochs and then overfitting occurs while LRCN reaches the best validation accuracy within 40 epochs, see Figure 5 where Figures 5a, 5b show the accuracy for the original train data, and Figures 5c, 5d show the accuracy for the augmented train data. In all situations, best-performing model weights were saved with respect to validation accuracy.

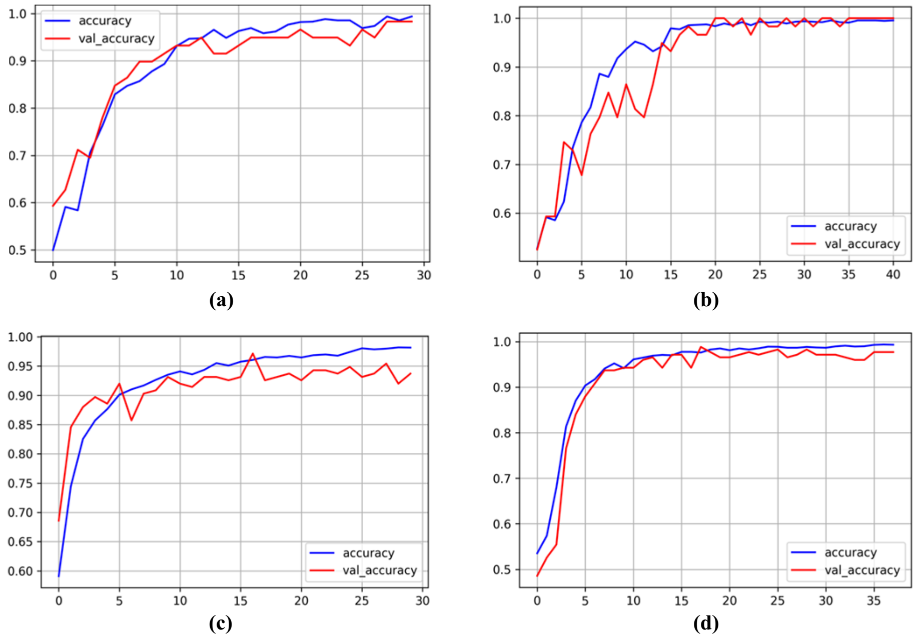


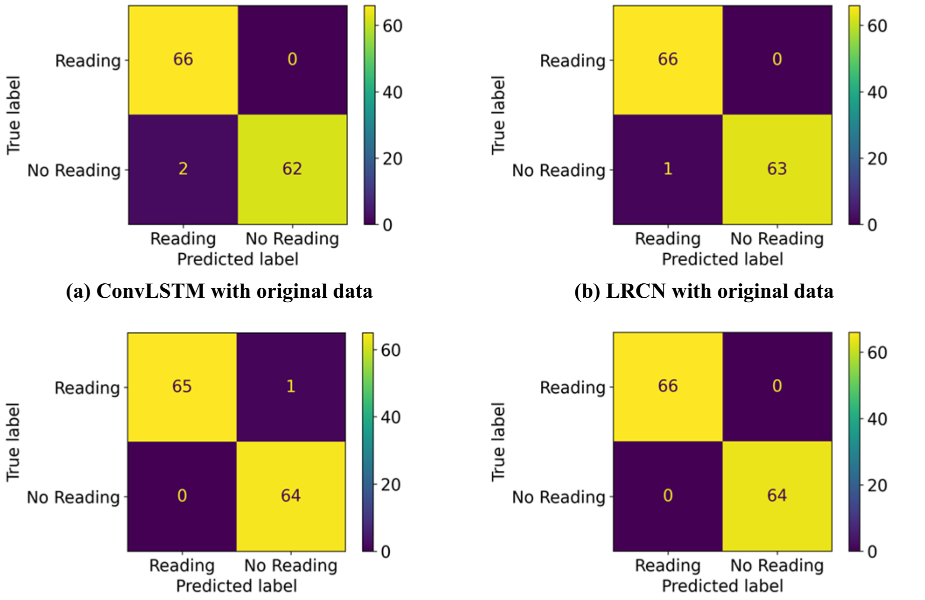
Fig. 5. The accuracy vs. epochs

Table 1 summarizes the performance metrics of all four experiments that are carried out in Google Colab with GPU. Here we also report the time it took to train the models per epoch and prediction time per video. Comparing ConvLSTM to LRCN we see that the latter is outperforming both in terms of four different performance metrics and in terms of train time per epoch.

Table 1. Summary of performance metrics for our experiments

Classifier	**Train Data**	**Train Time**	**Prediction Time**	**Accuracy**	**Precision**	**Recall**	**F1**
ConvLSTM	Original	16s	0.029s	98.46%	97.06%	**100%**	98.50%
LRCN	Original	3s	0.005s	99.23%	98.50%	**100%**	99.25%
ConvLSTM	Augmented	31s	0.023s	99.23%	**100%**	98.48%	99.24%
LRCN	Augmented	8s	0.003s	**100%**	**100%**	**100%**	**100%**

To visualize the performances of the models we provide the confusion matrices of the classifiers on the test set in Figure 6.



(a) ConvLSTM with original data

(b) LRCN with original data

(c) ConvLSTM with augmented data

(d) LRCN with augmented data

Fig. 6. Confusion matrices

5 Discussion

This work is devoted to the study of automated reading detection from the camera in an online exam. Two state-of-the-art deep learning models, ConvLSTM and LRCN, were adapted to address the reading detection task. To train the models, we collected our own dataset of videos with a total length of thirty-seven minutes and 10% of the dataset was allocated to test the performance of the proposed models. For each model, the training was carried out in two different cases where in the first case no data augmentation is carried and in the second case, several video augmentation techniques were used with the aim to improve the performance. In all cases, the models performed excellently as shown in Table 1 with LRCN reaching 100% accuracy for the augmented dataset in all four metrics. We also see that LRCN is at least four times faster compared to ConvLSTM.

Since the performances of the models are satisfactory, we have not used other possible approaches. In particular, skeleton or biometric-based action recognition may be adapted for similar tasks. However, we note that this particular approach is more likely to take a longer recognition time.

Our model was trained for color videos with frames converted into RGB format. While the model is robust to color videos, we have not tested it for grayscale videos. During data augmentation, a copy of the training dataset was generated by increasing the brightness up to 200%. Aside from addressing the overfitting, this makes the model robust against the different lighting conditions. Since we had no restriction on the video quality, we initially had various resolutions which are then converted into 720p as explained in section 3.1. Then, for training, the frame sizes reduced significantly which in particular reduced the resolution. Hopefully, this will make the model robust against videos with various resolutions.

We see from Figure 5 that the model performs equally well on both train and test sets. This indicates that there is no overfitting and the size of the dataset is sufficient for the binary classification task studied. However, to train the model for more complex tasks [41] one may need to collect more data.

Our model is designed to detect a possible screen reading detection. So, our model needs to see the person's eye to be able to say if the person is reading or not from a source on the computer screen or somewhere nearby. However, if a person is reading a book that is on the table, then it cannot detect it. A book reading detection task could be an interesting question to study in the future.

Since the accuracy is very high, there are very few cases when the model mislabels as shown in the confusion matrix in Figure 6. This makes it difficult to draw any insights into why the model mislabeled those particular clips. Maybe, increasing the test data could generate more mislabeled clips which could help one to analyze the possible reasons.

6 Conclusion

In this work, we trained a binary classifier for an online proctoring system. Clearly, there are other aspects of online proctoring related to the test taker's face and eye region. For example, in certain exams, the test takers may be required to look at the computer screen all time. In these situations, training the model to detect if a person is staring at the screen or not is important. In future work, the proposed model can be trained to carry more general action recognition tasks related to exam proctoring.

7 Acknowledgment

The authors acknowledge the support by a grant from the Ministry of Education and Science of the Republic of Kazakhstan within the framework of the Project AP14869776 – "Diophantine approximation and artificial intelligence". The second author also acknowledges the support by a grant from the Ministry of Education and Science of the Republic of Kazakhstan within the framework of the Project AP08051987-Irregular sets in Dynamical Systems".

8 References

[1] N. Muhammad and S. Srinivasan, "Transition from In-Class to Online Lectures During a Pandemic," in *International Conference on Interactive Collaborative and Blended Learning*, pp. 307–314, 2020. https://doi.org/10.1016/j.chbr.2021.100130

[2] S. Srinivasan, J. A. L. Ramos, and N. Muhammad, "A Flexible Future Education Model—Strategies Drawn from Teaching During the Covid-19 Pandemic," *Education Sciences*, vol. 11, no. 9, p. 557, 2021. https://doi.org/10.3390/educsci11090557

[3] N. Muhammad and S. Srinivasan, "Online Education During a Pandemic-Adaptation and Impact on Student Learning," *Int. J. Eng. Pedagog.*, vol. 11, no. 3, pp. 71–83, 2021. https://doi.org/10.3991/ijep.v11i3.20449

[4] F. Geng, S. Srinivasan, Z. Gao, S. Bogoslowski, and A. R. Rajabzadeh, "An Online Approach to Project-Based Learning in Engineering and Technology for Post-secondary Students," in *Interactive Mobile Communication, Technologies and Learning*, pp. 627–635, 2022. https://doi.org/10.1007/978-3-030-96296-8_56

[5] M. Alavi and S. Srinivasan, "Introduction of Online Labs to Enhance the Quality of the Real-Time Systems Course," in *Interactive Mobile Communication, Technologies and Learning*, pp. 856–864, 2022. https://doi.org/10.1007/978-3-030-96296-8_77

[6] H. Noprisson, "A Survey of the Online Learning Implementation During COVID-19 Outbreak," *International Journal of Recent Contributions from Engineering, Science & IT (iJES)*, vol. 8, no. 4, 2020. https://doi.org/10.3991/ijes.v8i4.17913

[7] M. Ouadoud, N. Rida, and T. Chafiq, "Overview of E-Learning Platforms for Teaching and Learning," *Int. J. Recent Contributions Eng. Sci. IT*, vol. 9, no. 1, pp. 50–70, 2021. https://doi.org/10.3991/ijes.v9i1.21111

[8] F. Noorbehbahani, A. Mohammadi, and M. Aminazadeh, "A Systematic Review of Research on Cheating in Online Exams from 2010 to 2021," *Education and Information Technologies*, 2022. https://doi.org/10.1007/s10639-022-10927-7

[9] C. S. Indi, K. V. Pritham, V. Acharya, and K. Prakasha, "Detection of Malpractice in E-Exams by Head Pose and Gaze Estimation," *International Journal of Emerging Technologies in Learning*, vol. 16, no. 8, 2021. https://doi.org/10.3991/ijet.v16i08.15995

[10] A. Tweissi, "The Effects of a Debate-Based Awareness Lecture on Cheating in Online Exams," *International Journal of Emerging Technologies in Learning*, vol. 17, no. 11, 2022. https://doi.org/10.3991/ijet.v17i11.30031

[11] C. S. Campbell and P. P. Maglio, "A Robust Algorithm for Reading Detection," in *Proceedings of the 2001 Workshop on Perceptive User Interfaces*, pp. 1–7, 2001. https://doi.org/10.1145/971478.971503

[12] Y. Kong and Y. Fu, "Human Action Recognition and Prediction: A Survey," *International Journal of Computer Vision*, 2022. https://doi.org/10.1007/s11263-022-01594-9

[13] S. Ji, W. Xu, M. Yang, and K. Yu, "3D Convolutional Neural Networks for Human Action Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, 2013. https://doi.org/10.1109/TPAMI.2012.59

[14] C. Schüldt, I. Laptev, and B. Caputo, "Recognizing Human Actions: A Local SVM Approach," in *Proceedings – International Conference on Pattern Recognition*, 2004, vol. 3. https://doi.org/10.1109/ICPR.2004.1334462

[15] H. Jhuang, T. Serre, L. Wolf, and T. Poggio, "A Biologically Inspired System for Action Recognition," 2007. https://doi.org/10.1109/ICCV.2007.4408988

[16] M. Yang, F. Lv, W. Xu, K. Yu, and Y. Gong, "Human Action Detection by Boosting Efficient Motion Features," 2009. https://doi.org/10.1109/ICCVW.2009.5457656

[17] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006, vol. 2. https://doi.org/10.1109/CVPR.2006.68

[18] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning Spatiotemporal Features with 3D Convolutional Networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4489–4497, 2015. https://doi.org/10.1109/ICCV.2015.510

[19] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A Dataset of 101 Human Actions Classes from Videos in the Wild," *arXiv preprint arXiv:1212.0402*, 2012.

[20] H. Wang and C. Schmid, "Action Recognition with Improved Trajectories," 2013. https://doi.org/10.1109/ICCV.2013.441

[21] Y. Jia *et al*., "Caffe: Convolutional Architecture for Fast Feature Embedding," 2014. https://doi.org/10.1145/2647868.2654889

[22]K. Simonyan and A. Zisserman, "Two-Stream Convolutional Networks for Action Recognition in Videos," in *Advances in Neural Information Processing Systems*, vol. 1, January, 2014.

[23] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: A Large Video Database for Human Motion Recognition," 2011. https://doi.org/10.1109/ICCV.2011.6126543

[24] L. Wang *et al*., "Temporal Segment Networks: Towards Good Practices for Deep Action Recognition," in *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016, vol. 9912 LNCS. https://doi.org/10.1007/978-3-319-46484-8_2

[25] J. Donahue *et al*., "Long-Term Recurrent Convolutional Networks for Visual Recognition and Description," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, 2017. https://doi.org/10.1109/TPAMI.2016.2599174

[26] J. Y. H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond Short Snippets: Deep Networks for Video Classification," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015. https://doi.org/10.1109/CVPR.2015.7299101

[27] F. T. Keat, S. Ranganath, and Y. v. Venkatesh, "Eye Gaze Based Reading Detection," in *IEEE Region 10 Annual International Conference, Proceedings/TENCON*, 2003, vol. 3. https://doi.org/10.1109/TENCON.2003.1273294

[28] A. Bulling, J. A. Ward, H. Gellersen, and G. Tröster, "Robust Recognition of Reading Activity in Transit Using Wearable Electrooculography," in *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2008, vol. 5013 LNCS. https://doi.org/10.1007/978-3-540-79576-6_2

[29] S. Ishimaru, K. Hoshika, K. Kise, A. Dengel, and K. Kunze, "Towards Reading Trackers in the Wild: Detecting Reading Activities by EOG Glasses and Deep Neural Networks," 2017. https://doi.org/10.1145/3123024.3129271

[30] Y. LeCun *et al*., "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, vol. 1, no. 4, 1989. https://doi.org/10.1162/neco.1989.1.4.541

[31] L. C. Ngugi, M. Abelwahab, and M. Abo-Zahhad, "Recent Advances in Image Processing Techniques for Automated Leaf Pest and Disease Recognition – A Review," *Information Processing in Agriculture*, vol. 8, no. 1, 2021. https://doi.org/10.1016/j.inpa.2020.04.004

[32] K. Mrhar, L. Benhiba, S. Bourekkache, and M. Abik, "A Bayesian CNN-LSTM Model for Sentiment Analysis in Massive Open Online Courses MOOCs," *International Journal of Emerging Technologies in Learning*, vol. 16, no. 23, 2021. https://doi.org/10.3991/ijet.v16i23.24457

[33] S. Hochreiter and J. Schmidhuber, "Long Short Term Memory. Neural Computation," *Neural Computation*, vol. 9, no. 8, 1997. https://doi.org/10.1162/neco.1997.9.8.1735

[34] M. Schuster and K. K. Paliwal, "Bidirectional Recurrent Neural Networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, 1997. https://doi.org/10.1109/78.650093

[35] J. Chung, "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling arXiv: 1412 . 3555v1 [cs . NE] 11 Dec 2014," *International Conference on Machine Learning*, 2015.

[36] A. Vaswani *et al*., "Attention is All You Need," in *Advances in Neural Information Processing Systems*, 2017.

[37] X. Shi, Z. Chen, H. Wang, D. Y. Yeung, W. K. Wong, and W. C. Woo, "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting," in *Advances in Neural Information Processing Systems*, 2015.

[38] N. Srivastava, E. Mansimov, and R. Salakhutdinov, “Unsupervised Learning of Video Representations Using LSTMs,” in *32nd International Conference on Machine Learning*, ICML 2015, 2015, vol. 1.

[39] A. Graves, “Generating Sequences with Recurrent Neural Networks,” *arXiv Preprint arXiv:1308.0850*, 2013.

[40] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and F. F. Li, “Large-Scale Video Classification with Convolutional Neural Networks,” 2014. https://doi.org/10.1109/CVPR.2014.223

[41] A. Shdaifat, R. Obeidallah, G. Ghazal, A. A. Sarhan, and N. A. Spetan, “A Proposed Iris Recognition Model for Authentication in Mobile Exams,” *International Journal of Emerging Technologies in Learning (iJET)*, vol. 15, no. 12, pp. 205–216, 2020. https://doi.org/10.3991/ijet.v15i12.13741

9 Authors

Bakhitzhan Kadyrov is a senior PhD student at Suleyman Demirel University, Faculty of Engineering and Natural Sciences, Department of Computer Sciences in Kaskelen, Kazakhstan. His research interests are exam proctoring systems, data science, web development (email: 202107003@stu.sdu.edu.kz).

Shirali Kadyrov is a Professor at Suleyman Demirel University, Faculty of Engineering and Natural Sciences, Department of Mathematics and Natural Sciences in Kaskelen, Kazakhstan. He obtained his PhD in 2010 from the Ohio State University, Columbus, USA. His research interests are dynamical systems, mathematics education, and data science (email: shirali.kadyrov@sdu.edu.kz).

Alfira Makhmutova is a Senior Instructor at Suleyman Demirel University, Faculty of Education and Humanities, Kaskelen, Kazakhstan. She obtained her PhD in 2022 from Nazarbayev University, Nursultan, Kazakhstan. Her research interests are education, language maintenance (email: alfira.makhmutova@sdu.edu.kz).

Article submitted 2022-06-13. Resubmitted 2022-08-05. Final acceptance 2022-08-15. Final version published as submitted by the authors.