Continuous Face Detection and Verification During Online Examinations

Dominik Kurasbediani

Warsaw University of Technology

Author Note

[Include any grant/funding information and a complete correspondence address.]

Abstract

I present a prototype of a system used to continuously detect and verify faces of examinees during online examinations, as well as a method of representation of the results of such verifications, and overall architecture of the system. The system consists of three modules: the client-side face detection, the serverside face verification, and the exam supervisor’s dashboard. Face detection occurs on each client’s computer using Single Shot Multibox Detection (SSD). The resulting image is aligned and cropped, then encoded into base-64 format and sent to the verification server via a POST request. Upon receiving such request, the server decodes the base-64 image, feeds it to the VGG-Face model to get a vector representation of the face and compares the Euclidean distance between said image and some predefined base image. The results of the last operation are sent to the exam supervisor’s dashboard via a POST request. The exam supervisor’s dashboard’s task is to receive and process such requests and represent the updated data in real time. In addition, the exam supervisor’s dashboard is responsible for configuration of the exam session, as well as generation of the clientside executables to allow easy and seamless process for the students.

Keywords: Real-Time Face Recognition, Client-Server Architecture, Online Examination.

## Motivation

The current solutions for online examination are variants of the following approach: the students connect to a third party-provided online meeting (e.g., Zoom, Microsoft Teams), turn on their cameras, proceed to complete the exam. This method is flawed in terms of security against academic offenses. The most prevalent one, in my experience as a student, is the abuse of the fact that the exam supervisor has never seen students’ faces before the exam due to it not being required during the semester, as well as the absence of identity verification as a prerequisite to writing the exam. In other words, students may outsource (ask their friends to write the exam for them, pay a professional to do it, etc.) and get a passing grade regardless of whether they truly know the material presented during the semester. During the COVID-19 pandemic, when there was no other option other than online education and online exams, the students were incentivized to abuse this system and commit academic offenses and get a passing grade without opening the course textbook once during the semester. This effectively lowers the overall education level, potentially creating severe errors, that may be lethal in some cases, due to students-turned-professionals being unqualified. One may argue that the proportion of students cheating is marginal at best, however this problem is a hole in a ship that might lead to it sinking.

With an existing state-of-the-art face recognition technology, it is possible to prevent these problems by building and integrating security applications into existing infrastructures of academic institutions.

## Existing Procedures

Normally examinations are held in-person, which prevents students from committing academic offenses under the watchful eyes of the exam supervisors and their assistance. The identity of examinees is established and verified either on entrance to the exam room or during the exam itself: the examinees are asked to present a student identification card that contains the image of their face, first and last name(s), and the student ID number. This information is verified against a list of students registered for this exam, and the images in the identification card is matched with the face of the owner of the card. If these conditions are met, the examinee is cleared to write the exam. Examinees are not allowed to leave the exam room until they have turned in their exam papers, or their identity is verified once again on re-entry.

The described above procedure is put in place with security in mind, to prevent the problem mentioned in the previous section. It has been successful in preventing most cases of identity fraud in the context of examinations. However, currently there is no such procedure being used for online examinations.

## Proposed Procedure

The procedure I use in my application is modeled after the existing in-person exam protocol:

1. Before the start of the exam session, exam supervisors are required to create a list of students allowed to take the exam (first and last names, student numbers, optional base image).
2. The client executables (access point for the students) are generated and distributed to the students.
3. The exam supervisor starts the exam session.
4. The students launch the executables, input their identification data (first and last names, student number)
5. The system verifies their faces throughout the duration of the whole exam.

The described above procedure replicates the in-person procedure except for the last step. Step 6 covers an edge case that is only possible during online examinations: it is not possible to fake one’s identity once it has been verified without leaving the examination room first. However, during online examinations it is possible. By ensuring continuous verification, the application prevents it from happening.

# Chapter 1. System Architecture

Any face recognition pipeline consists of at least four parts: face detection, alignment, face representation, and verification. The system consists of three modules: client-side face detection, server-side face verification, and exam supervisor’s dashboard. The pipeline stages are spread across the three modules to optimally distribute the stress and utilization of computational resources required of each stage.

## Exam Supervisor’s Dashboard

The Dashboard module is responsible for configuration of the session, executing compilation scripts, starting the verification server, and presentation of the verification results. It is a web application that the examination supervisor exclusively will have access to. The module consists of two components: the frontend and the backend.

### Frontend

This component is responsible for the visual representation of the data and allowing the supervisor to utilize all the tools provided efficiently and with ease. The User Interface (UI) is designed to be as minimalistic and intuitive as possible to minimize the training required to become acclimated to the system. It is built with React JS, which is a JavaScript framework.

Timeline

Description automatically generated with medium confidence

Figure 1. Default UI of the Dashboard

#### Add Student

This button redirects the user to a separate page that contains a form that allows for addition of new students to the database. The required components are the first and last names, and the student identifier. There is also an optional field to add a base image of the student. This base image is uploaded to the backend server and saved locally for the verification purposes. The ‘Cancel’ button redirects the user back to the previous page.

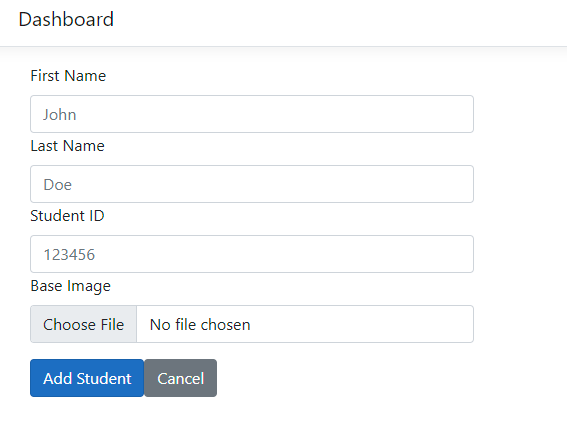


Figure 2. Add Student Form

When the form is submitted, the contents are sent to the backend via a POST request, where it is processed further.

### Backend

This component is responsible for all the data management, script execution and operation of the dashboard. It is a REST API built with C# and ASP.NET Core.

#### Database

The module is implemented in a way that omits configuration and maintenance of dedicated databases and instead utilizes an in-memory database built in runtime. This solution reduces the cost of maintenance in the long-term at the cost of building the database every session (which is compensated by the ability of the user to export the session data and load sessions from a .json file). The solution also decreases the amount of time required for the changes to occur due to the lack of the need to send and receive data from a dedicated database. However, it is configured in a way that would not take a significant amount of development time to implement support of a dedicated solution due to utilization of .NET’s Entity Framework.

#### API

Due to the limitations of the design, it is impossible to utilize some of the features of React efficiently. Instead of only refreshing the contents when there is change automatically, the contents are manually refreshed with the given frequency. This results in a consistent amount of GET requests to the backend

[The body of your paper uses a half-inch first line indent and is double-spaced. APA style provides for up to five heading levels, shown in the paragraphs that follow. Note that the word Introduction should not be used as an initial heading, as it’s assumed that your paper begins with an introduction.]

# [Heading 1]

[The first two heading levels get their own paragraph, as shown here. Headings 3, 4, and 5 are run-in headings used at the beginning of the paragraph.]

## [Heading 2]1

[To add a table of contents (TOC), apply the appropriate heading style to just the heading text at the start of a paragraph and it will show up in your TOC. To do this, select the text for your heading. Then, on the Home tab, in the Styles gallery, click the style you need.]

### [Heading 3].

[Include a period at the end of a run-in heading. Note that you can include consecutive paragraphs with their own headings, where appropriate.]

#### [Heading 4].

[When using headings, don’t skip levels. If you need a heading 3, 4, or 5 with no text following it before the next heading, just add a period at the end of the heading and then start a new paragraph for the subheading and its text.] (Last Name, Year)

##### [Heading 5].

[Like all sections of your paper, references start on their own page. The references page that follows is created using the Citations & Bibliography feature, available on the References tab. This feature includes a style option that formats your references for APA 6th Edition. You can also use this feature to add in-text citations that are linked to your source, such as those shown at the end of this paragraph and the preceding paragraph. To customize a citation, right-click it and then click Edit Citation.] (Last Name, Year)

References

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). Ssd: Single shot multibox detector. In European conference on computer vision (pp. 21-37). Springer, Cham.

Zhang, K., Zhang, Z., Li, Z., & Qiao, Y. (2016). Joint face detection and alignment using multitask cascaded convolutional networks. IEEE signal processing letters, 23(10), 1499-1503.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems, 28.

Gao, X., Xu, J., Luo, C., Zhou, J., Huang, P., & Deng, J. (2022). Detection of Lower Body for AGV Based on SSD Algorithm with ResNet. Sensors, 22(5), 2008.

Serengil, S. I., & Ozpinar, A. (2020, October). Lightface: A hybrid deep face recognition framework. In 2020 innovations in intelligent systems and applications conference (ASYU) (pp. 1-5). IEEE.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Footnotes

1[Add footnotes, if any, on their own page following references. For APA formatting requirements, it’s easy to just type your own footnote references and notes. To format a footnote reference, select the number and then, on the Home tab, in the Styles gallery, click Footnote Reference. The body of a footnote, such as this example, uses the Normal text style. (Note: If you delete this sample footnote, don’t forget to delete its in-text reference as well. That’s at the end of the sample Heading 2 paragraph on the first page of body content in this template.)]

Tables

Table 1

[Table Title]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column Head | Column Head | Column Head | Column Head | Column Head |
| Row Head | 123 | 123 | 123 | 123 |
| Row Head | 456 | 456 | 456 | 456 |
| Row Head | 789 | 789 | 789 | 789 |
| Row Head | 123 | 123 | 123 | 123 |
| Row Head | 456 | 456 | 456 | 456 |
| Row Head | 789 | 789 | 789 | 789 |

Note: [Place all tables for your paper in a tables section, following references (and, if applicable, footnotes). Start a new page for each table, include a table number and table title for each, as shown on this page. All explanatory text appears in a table note that follows the table, such as this one. Use the Table/Figure style, available on the Home tab, in the Styles gallery, to get the spacing between table and note. Tables in APA format can use single or 1.5 line spacing. Include a heading for every row and column, even if the content seems obvious. A default table style has been setup for this template that fits APA guidelines. To insert a table, on the Insert tab, click Table.]

Figures title:

Figure 1. [Include all figures in their own section, following references (and footnotes and tables, if applicable). Include a numbered caption for each figure. Use the Table/Figure style for easy spacing between figure and caption.]

For more information about all elements of APA formatting, please consult the APA Style Manual, 6th Edition.