Journal of Modern Education Review, ISSN 2155-7993, USA
May 2022, Volume 12, No. 5, pp. 391–397
Doi: 10.15341/jmer(2155-7993)/05.12.2022/009

http://www.academicstar.us

Intelligent Algorithms to Enhance Teaching

Michael Chourdakis
(Music School of Alimos, University of Athens, Greece)

Abstract: This paper presents a set of methods to support/enhance the typical schoolwork of a high school student, especially when undergoing remote learning due to Covid-19 restrictions. We have created algorithms in various languages (typically C++ and PHP) and we have presented them in forms of web-applications to the students in Music School of Alimos during two academic years, 2019–2020 and 2020–2021.

Key words: C++, E-learning, PHP, algorithm, anti cheating, video editing, ear training

1. Introduction

Recently, the day of a high school student and/or their teachers is not only restricted to typical old-fashioned learning/teaching, but also requires substantial office support, paperwork, and e-applications to aid the whole process.

In our school, we have faced a list of problems, both simple and advanced, to deal with. Here we will discuss five of them:

- Managing the evaluation of the students,
- Limiting the possibilities that the students do not cheat while participating in a remote-learning class,
- Ensuring that the distribution of the assignments to the student is fair and effective,
- Aiding advanced ear training lessons and
- Creating advanced video editing lessons.

In the following chapters we will elaborate on each of the algorithms and the statistical results we have achieved by applying them.

2. Algorithms

2.1 Student Evaluation

According to current rules (Law of Greek Schools Operation and Law of Greek Music Schools Operation), a student must be evaluated in writing at most once per day and at most three times per week. Before our algorithm was applied, teachers in our school ought to personally ask each other to detect empty spaces within the weekly school timetable, to schedule their evaluation. This process was error-prone (overlapping tests, more than three tests per week, etc.), the table below showing the errors measured when the schedule was probed:

Michael Chourdakis, Ph.D., Music School of Alimos, University of Athens; research areas: digital signal processing, artificial intelligence in sound, image and video applications, and AI in education. E-mail: chourdakismichael@gmail.com, m.chourdakis@music.uoa.gr.

Table 1 Stats Before Algorithm Applied

Teachers	Classes	Tests per period	Error rate
72	18	216	51.8% (112 errors) 45 more than 1 test per day 56 more than 3 tests per week 11 no test taken

More than half of the tests were wrongly scheduled. They had to be rescheduled at the school director's notice. Our algorithm, designed in C++ and presented to our staff in a form of a web application, is described in the following steps:

Table 2 Algorithm Description

Step #	Description
1	Determine which classes need to be evaluated by writing
2	Prioritize class scheduling, considering: • Teacher presence in school (not all teachers are present each day in the school) • Class importance (higher grade classes take a coefficient of 1.2; Middle grade classes take a coefficient of 1.1 and lower grade classes take a coefficient of 1.0) (Williams & Ajinaja, 2019) • Add a coefficient factor depending on previous teacher response of examination scheduling (if any) (Kuldeep Singh, 2003)
3	Examine the teacher timetable to detect empty spaces for scheduling (Bagul et al., 2015)
4	Span probing threads that build an examination schedule according to the above rules (Bagul et al., 2015)

The thread that has the best sparse ratio (i.e., the student has the most available time to prepare for their next exam) is selected and the examination schedule is automatically created.

Table 3 Stats After Algorithm Applied

Teachers	Classes	Tests per period	Error rate	Manual intervention needed
72	18	216	0%	3/216 3 of the 216 tests were too close to others.

The errors were minimal and manageable through manual rescheduling at a much lower complexity.

2.2 Student Anti-cheating Mechanisms

According to current rules, a method to authenticate students in their online presence was not provided by the Ministry of Education. Instead, students clicked on a statically generated link, choose a nickname (which could be anything and could change at any moment) and log in to the e-class. This means that it is practically infeasible to prove a student's verified presence to the courses. Research on our school during the e-courses due to Covid-19 restrictions in 2020 demonstrated the results at the following table:

Table 4 Stats Before Algorithm Applied

Students	Fake nicks	Fake presence/bot detected
398	55	31

Our algorithm is designed in PHP and presented via a web application to teachers. It's described as follows:

Table 5 Algorithm Description

Step #	Description
1	Login students with the GSN (Greek School Network) credentials to avoid fake identification
2	Interoperation with Webex Cisco API for developers to query IP address, Cookies, and Browser footprints: • Compare IP from GSN to IP from WCAPI, • Compare session cookie from GSN/WCAPI/ our referrer network • Compare browser footprints (Jana & Shmatikov, 2012)
3	Implementation of undisclosed action-related methods (Karmand, 2016) to detect abnormal keyboard and mouse activity.
4	Implementation of disclosed action-related methods (Kahtan H. & Kashmar D., 2020) to detect network inactivity.

The results are presented in the following table:

Table 6 Stats After Algorithm Applied

Students	Fake nicks	Fake presence/bot detected
398	12	4

This is an improvement of 10.79% for fake nicks and 6.77% for fake presence.

2.3 Automatic Assignment Distribution

Student evaluation involves not only written exams, but also offline assignments and collaboration exercises. Typically, a teacher assigns approximately the same number of assignments to all students, which leads to an incomplete evaluation of them because of the following:

- Students of a lower knowledge level than the class average would find it too difficult to accomplish many of the assignments and eventually quit their efforts.
- Students of a higher knowledge level than the class average would find many assignments too easy for them and eventually get bored.

In an example class with 22 students, the average is 14 (out of 20), where 5 students belong to the 17–20 area and 4 students belong to the 5–10 area. Initially, all students were assigned the same assignments without considering their knowledge level.

The following table shows the pre-algorithm results:

Table 7 Stats Before Algorithm Applied

Students	Above average	Below average	Bored	Quit
22	5	4	3	3

This means that in at least 6 students the course failed to exploit all their capabilities.

Table 8 Algorithm Description

Step #	Description
1	Evaluation of all student grades within 6 years to determine a more widen average
2	Evaluation of all student exams within 6 years to determine differences between oral and written examination
3	Evaluation of the teacher's grades within 6 years to estimate the grading style of the teacher
4	The teacher selects a difficulty level to the assignments (1–5)
5	Recommendation to the teacher, from the specified assignments, how many and which assignments should be given to each student

The statistical methods used are common to any sort of data (Ali & Bhaskar, 2013), including the mean ($\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$), the standard deviation ($s_n = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i - x)^2}$), the linear regression ($y = ax + b$), the T – Testing and a determination of the sample size equal to 6 years.

Table 9 Stats After Algorithm #3 Applied

Students	Above average	Below average	Bored	Quit
22	7	2	0	1

Not only the above average of the class raised, but also only one student with a lower-than-average knowledge level quit their assignments.

2.4 Advanced Ear Training

With current laws of education, basic ear training is taught in every music school in the country. This includes tone detection, interval comparison, chord detection, chord progressions, mode detection and rhythmic exercises.

In our school however, we occasionally encounter advanced-level students who are already quite familiar with the above basic training. Therefore, a higher level of ear acoustics is required, designed for sound editors and engineers. This includes theory and practice on difficult courses like equalizer patterns, compression ratio and threshold levels, 2D and 3D pan detection, song pitch shifting, delay and distortion detection and reverb probing.

Current implementations of e-class and online teaching (Webex, Zoom, Teams, Skype etc.) are not designed for such requirements. They assume that the main audio stream passed is speech and optimize accordingly, applying heavy equalization, filtering, and compression methods to provide optimal experience even with a low bandwidth, or to preserve bandwidth for higher demand like webcam video. Therefore, those applications cannot help us in advanced ear training courses. In known exercises, our students performed badly when tested:

Table 10 Stats Before Algorithm Applied

Students	Successful pan detection	Successful equalizer test	Successful compression test	Successful distortion test
22	70%	20%	18%	85%

Our solution implements the following algorithm:

Table 11 Algorithm Description

Step #	Description
1	Usage the OPUS open-source algorithm in full frequency mode (no speech) to deliver the full frequency range online (Carôt, 2010)
2	Caches sound to induce a small delay in exchange for better quality
3	Presents in a web form the answers so the test can be automated
4	Performs pitch detection (Chourdakis, 2007) to enhance signal equalization exercises based on the pitch

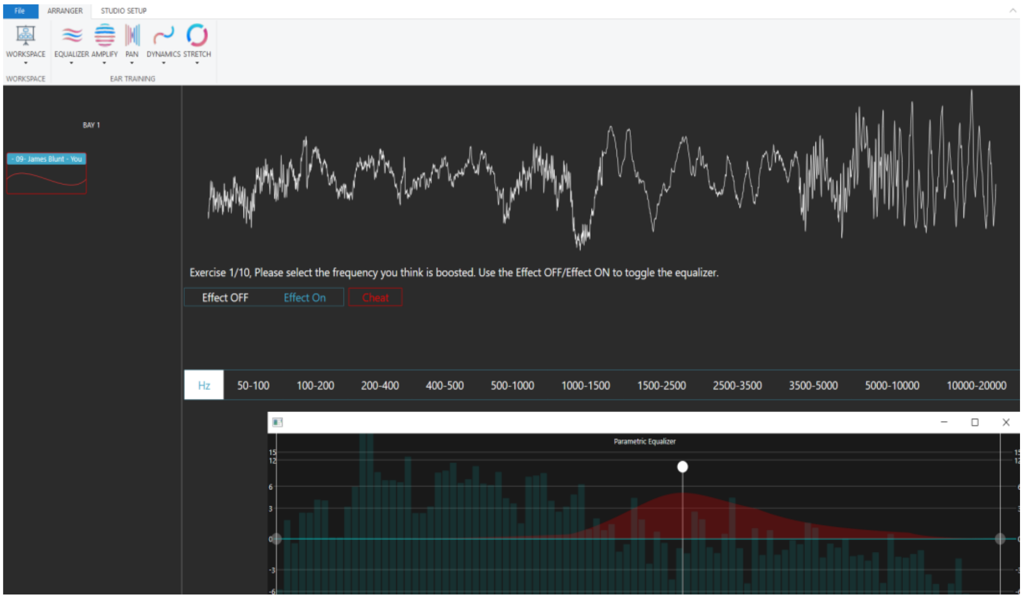

Figure 1 Equalization Test via Virtual Class

Table 12 Stats After Algorithm Applied

Students	Successful pan detection	Successful equalizer test	Successful compression test	Successful distortion test
22	70%	80%	78%	85%

As we see, in the equalizer and compression tests there was a much higher success rate.

2.5 Project: Video Editing

From 2012 and until 2020, a course named "Project" used to exist in the higher classes of Greek high schools. The idea was to train students into high levels of academic research based on a free pick of whatever research subject they would wish to work on, supported by their teacher. This course was eventually abandoned in 2020 in favor of another similar education scheme targeted to lower-level education.

In our school the average level or the students is high enough to support advanced courses. Since 2013 we have been teaching advanced topics to our most capable students, including acoustics, image editing, audio sequencing and C++ programming.

In 2019 we attempted to teach video editing and the issues in the screen sharing in typical e-learning applications, were similar to the ear training issues:

- Resolution lowering had been applied to the video so the color quality was reduced, this would prevent efficient color grading, work on high FPS (Frames per second), HLSL debugging of GPU code is impossible,
- Differential compression had been applied to the video and this did not allow our students to detect glitches in the sequencing (Microsoft, 2013),
- Simultaneous editing of video and audio together was difficult.

Table 13 Stats Before Algorithm Applied

Students	Color grading	Sound editing	Effect authoring	Quality result
15	15%	20%	18%	20%

Our final algorithm in Virtual Class uses the following:

Table 14 Algorithm Description

Step #	Description
1	HTTP Live Streaming (at the time most used, now to be replaced by MPEG-DASH) to ensure optimal performance for the delivery of high-quality video stream depending on the bandwidth capacity of each student individually (Bentaleb et al., 2018).
2	H.265 compression to minimize network bandwidth (Nguyen et al., 2013).
3	Interactive desktop sharing (not only view, but interact at vector level) so the users can access the software presented from their own system
4	Vector-based differential compression that minimizes bandwidth on small screen changes (Microsoft, 2013).

The tradeoff is increased latency which would prevent real time audio transmission and, all this technology is only supported in Windows.

Table 15 Stats After Algorithm Applied

Students	Color grading	Sound editing	Effect authoring	Quality result
15	35%	15%	38%	42%

This final table demonstrates what we have achieved in terms of color grading, effect authoring and overall image quality result at the expense of simultaneous sound.

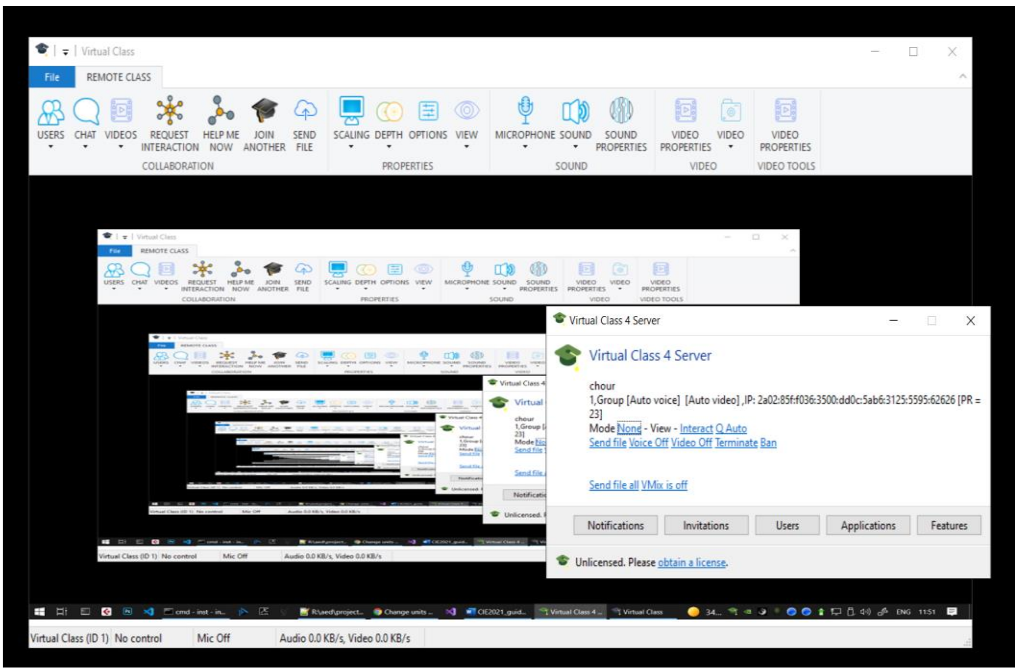

Figure 2 Virtual Class Interface With Desktop Sharing Mirrored

3. Summary

In this paper we presented our algorithms to enhance the teaching process and resolve many common issues of various teaching facts. Advanced technology and increased computing power in the last years combined with the higher level of knowledge of many of our students make our algorithms plausible and efficient.

References

Bagul M., Chaudhari S., Nagare S., Patil P. and Kumavat K. (2015). "A novel approach for automatic timetable generation", *International Journal of Computer Applications*, Vol. 127.

Bentaleb A., Taani B., Begen A., Timmermer C. and Zimmermann R. (2018). "A survey on bitrate adaptation schemes for streaming media over HTTP", *IEEE Communications Surveys & Tutorials (IEEE COMST)*, Vol. 1, No. 1, pp. 1–1.

Carôt A. (2010). "Low latency audio streaming for internet-based musical interaction in streaming media architectures: Techniques and applications: Recent advances". *IGI Global*, pp. 362–383.

Craft A. (2003). "The limits to creativity to education: Dilemmas for the educator", *The Open University British Journal of Education Studies*, Vol. 51, No. 2, pp. 113–127.

Chourdakis M. (2007). "Time domain pitch recognition", in: *4th SMC International Conference UOA*, Lefkada, pp. 307–311.

Chourdakis M. (2016). "Virtual class II", in: *3rd conference NIT*, Athens.

Chourdakis M. (2020). "Artificial intelligence algorithms in school administration", *Teachers in Foreground*, Athens, pp. 47–51.

Greek Ministry of Education, Law of Education in Schools and Music Schools (M.D. 13423/CD4/2021- GG 491/B/9-2-2021, M.D. 20923/D2/23-02-2021 GG B΄ 878/05-03-2021).

Jana S. and Shmatikov. V. (2012). "Memento: Learning secrets from process footprints", in: *2012 IEEE Symposium on Security and Privacy*, pp. 143–157.

Kahtan H. and Kashmar D. (November 2020). "A comparison study for CAPTCHA security", *Research Gate*.

Karmand H. and Kaya M. (December 2016). "An evaluation of different types of captcha: Effectiveness, user-friendliness, and limitations", *Research Gate*, No. 2, pp. 12–19.

Kezar M. and Lester J. (2009). *Organizing Higher Education for Collaboration: A Guide for Campus Leaders*, Jossey-Bass.

Microsoft (2013). "DFS replication", pub. 16 October 2006, updated 2013.

Simonson M., Smaldino S. and Zvacek S. M. (2014). *Teaching and Learning at a Distance: Foundations of Distance Education* (6th ed.), Information Age Publishing.

Nguyen T., Helle P., Winken M., Bross B., Marpe D., Schwarz H. and Wiegand T. (2013). "Transform coding techniques in HEVC", *Journal of Selected Topics in Signal Processing*, Vol. 7, No. 6, pp. 978–989.

Viola P. and Jones M. (2004). "Rapid object detection using a boosted cascade of simple features", Mitsubishi Electric Research.

Wexelblat A. and Maes P. (1997). "Footprints: History-rich web browsing", *MIT Media Lab*, Software Agents Group E15-305, 20 Ames St., Cambridge, MA 02139.

Williams M. and Ajinaja K. (Jan. 2019). "Automatic timetable generation using genetic algorithm", *Journal of Information Engineering and Applications*.

Zulfiqar A. S. and Bala, B. (September 2016). "Basic statistical tools in research and data analysis", *US National Library of Medicines*, pp. 662–669.