Class Extractor

Design Document

Elliot Liskin

November 2015

Class Extractor is a study tool for students that uses natural language processing algorithms to determine what the most important topics in a course are, ranking and organizing them to help prioritize what should be studied using a combination of lecture recordings, student notes, and class PowerPoint presentations. Class Extractor is meant to ease the studying process, helping students organize and prioritize all of the information they have learned over the course of a semester. This goal also reduces the pressure on students, enabling them to perform better on exams and recall more information more readily. Students who are less stressed may score better in classes and lead a healthier life outside of classes, as the amount of pressure in-class is reduced.

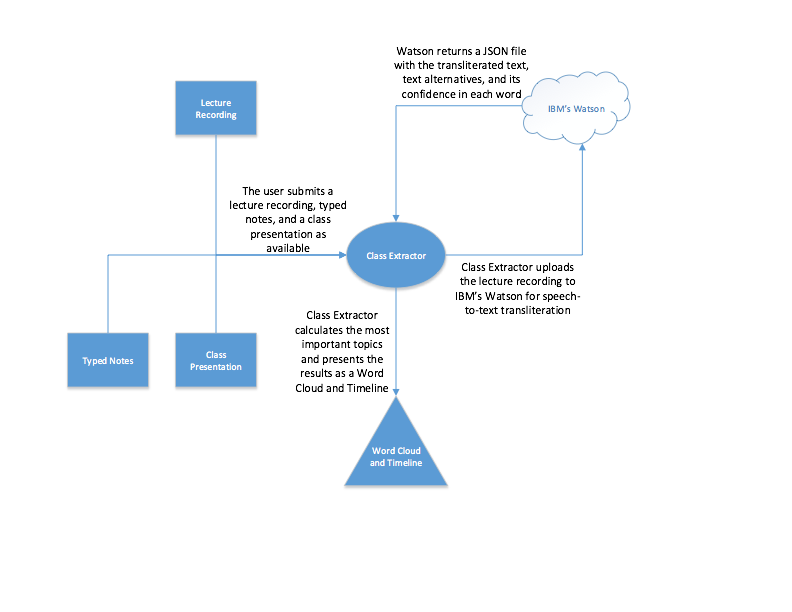
The primary users of this application are students who have difficulty staying organized or who have trouble prioritizing information, with other users including professors and other students. Students who have difficulty prioritizing information can use Class Extractor to help them, as it can greatly increase their efficiency and effectiveness by showing them which information and topics should be prioritized. Information is organized logically, making it easier for these students to digest the material and refer to lecture notes and class recordings of where the material was discussed. Other students are students who may not have these difficulties, but could still benefit from using the application. This group is amorphous, hence the name “other students,” but it includes any student who wants or needs help studying, wants to learn more from his or her courses, or generally wants to perform better on exams. Professors, the third group, can benefit from Class Extractor by using the application to determine what is perceived to be the most important concepts from class. Professors can determine if they are overemphasizing a topic they did not mean to be that important, underemphasizing something when it should have been of higher priority, seeing if they have too many irrelevant tangents in discussions, or neglecting to talk about a concept entirely. Using perceived important topics, professors can alter what they say in class and how they say it to better communicate information and course concepts to students.

For users of the first and second groups, students who have difficulties organizing information and other students, the usage patterns and use cases are similar, so they will be discussed together. Typically, these users fall into two groups: those who will use the application on a regular basis to stay on top of material taught throughout the semester, and those who will only use it when studying for an exam. In either situation, the student chooses a lecture recording, a text document containing his or her notes, and a class presentation for analysis (any combination of these resources are accepted, however the more there are the more accurate the analysis will be). The lecture recording and the presentation contain what the professor thinks is important, since they came from the professor, and the notes are what the student thinks is important, since they came from the student. The notion that increasing numbers of inputs makes the resulting analysis more accurate is hinged on the fact that the student takes diligent and careful notes. If there are concepts missing because the student spaced out, the student forgot to type something, or the student simply did not think something was important, gaps in concepts will confound the analysis. However, the notes are still a valuable input for two reasons. The first reason is that if the student took thoughtful notes, they can still help the analysis in figuring out the most important topics, just like the other two inputs. The second reason is that student typed notes can show what the student was confused by or wanted to revisit at a later time (that is, the student took note of something specifically because he or she did not understand it). As such, displaying those topics in the application can help students view the material and show them where the material is discussed in the lecture recording and in the class presentation.

The third group, professors, can use Class Extractor to learn what the topics they are implicitly telling their students are important. Over the course of a given semester, some topics are critical to the class, and some are not, yet sometimes emphasis might be misplaced, leading students to study and learn the wrong material, leading to worse performance on tests and less relevant knowledge gained for use after the semester ends. When presenting topics in front of an audience, presenters can find it difficult to gauge what the audience is absorbing and what they are seeing as important. It is easy to go on a tangent that may be relevant in the mind of the professor, but not so much in terms of the class material or to the students. Class Extractor can be an invaluable tool for professors to see exactly what they are asking the students to remember. What is taught in a class is a contact between the professor and the students; the contract says that all of this material may be on the exam, that all of this material is relevant to the subject matter of the course, and that, if a student pursues this subject matter after the semester ends, the material will continue to be relevant to him or her. This contract means that tangents and anecdotes that are neither part of the course material nor of help to students for remembering course material are detrimental to the learning experience. Class Extractor can make these unproductive discussions evident to the professor, as the professor can upload a lecture recording to view what the program sees as the most important topics of a lecture, and if the most important topics are not what he or she was expecting, the professor can alter future lectures to realign the concepts actually taught with the concepts intended to be taught.

For all three groups, the workflow is largely the same. The user gives Class Extractor his or her available resources, and Class Extractor performs the analysis. All students of the first and second groups will upload all of the resources they have, whereas professors in the third group would upload only a recording of the lecture and a presentation. Professors might opt to only use a recording because it is obvious looking at a presentation if the wrong topics are emphasized, whereas that is not necessarily the same for a lecture recording because everything may seem relevant to the professor since he or she was the one who said it. Once the relevant resources are chosen, Class Extractor performs its analysis, and returns to the user a visual interface that is broken down into two components: a Word Cloud and a Timeline. The user navigates through the Word Cloud by clicking on clouds of interest. The size of each cloud directly corresponds to the importance of the topic in the cloud, and clicking on it zooms into the cloud to display a Word Cloud of subtopics. Each cloud also contains links to the spots in each of the resources where the topic is discussed. The second component of the topic navigator, the Timeline, allows the user to visually see where each of the topics occur in the lecture recording, how topics overlap, where subtopics come in, and how different concepts interrelate. Students can use this interface to see how concepts are mapped over the course of a lecture, and professors can validate that they are discussing the topics they want to in the order they intend. The functional components, in these use cases, are the uploading mechanism that allows the user to tell Class Extractor which resources to perform the analysis on, as well as the dual-component interface that the users can look at to see the results of the analysis.

Non-functional components of the application include the performance and responsiveness of the natural language processing analysis. The speech-to-text transliteration of the lecture recording is done by IBM’s Watson, which entails a lot of I/O. This process should be done on background threads to prevent freezing of the interface and ensure responsiveness. Likewise, much of the other associated processes, such as splicing audio files, converting them to the appropriate file types, and analyzing the returned transliterated text all require a lot of CPU time. As such, these processes should also be done in the background to decrease the amount of time between users submitting their resources and them receiving the resulting analysis.

Below is a data-flow diagram of how Class Extractor is expected to work. The user gives Class Extractor his or her three resources as available or as relevant (in the professor’s case). Class Extractor sends the lecture recording to Watson for speech-to-text transliteration, which returns the text that Class Extractor can combine with the other inputs for analysis. After determining what the most important topics and concepts are, Class Extractor presents the results as a Word Cloud and as a Timeline as discussed above. In the diagram, rectangles represent inputs, ovals represent Class Extractor, clouds represent servers, and triangles represent outputs.