CS 6750: Assignment P1

Michael Tong
mtong2@gatech.edu

Question 1

For the discussion topic of an OMSCS student interface, I will be analyzing the Udacity interface. I find that the Udacity interface is unique in that it serves two primary users, us OMSCS students, as well as the general public. Unfortunately, since this discussion is limited in space, I will only be discussing the perspective of us, the student, but it may be important to consider why certain decisions were not made from the perspective of the public user.

Processor Model Perspective

From the perspective of the user as a processor for Udacity, the main objective of the platform is to effectively transfer information from educators to students via video. In order to accomplish this task, the focus should be on the material, how quickly the user can access it, and how efficiently they can absorb the information presented from it. To benchmark the efficacy of Udacity in performing these tasks, a similar and common framework such as YouTube could be used for comparison.

As an OMSCS student, access to the material of interest is simple as our accounts are prefilled with the classes we enroll in. It may not however, be intuitive that we are required to log in with our GT account, if we happen to already have an Udacity account that is already logged in. This requires the user to log out and log back in, then select the Georgia Tech log in option. Since the classes are open to the public, I had initially viewed the class on a private account and had to redo progress on my GT account once realizing the mistake.

Outside of that flaw, the interface does include unique features that improve the access time for the material of interest. One such feature is the added simplicity of returning the user to their last viewed video, and an additional catalog to combine all specific videos for a class, where with YouTube, the playlist feature acts similarly, but requires the user to manual find and add the videos. YouTube does however have Udacity's beat on the required number of clicks to reach a

specific video (not the list). With a simple search bar on the main page, as well as the option to search directly through the address bar, Udacity lacks this feature, and requires the user to click through a few more tabs before arriving at the video, excluding the use of direct links. While the search functionality is much simpler than Udacity's, the layout of the viewer in Udacity is much more minimalistic in comparison with YouTube. The former is much more focused on the video at hand, whereas the latter displays a lot of additional signifiers in the form of advertisements. The minimalistic approach affords the user with less distractions, allowing them to focus on absorbing the information at hand. From the perspective of an OMSCS student, since the task is very specific and the program is built in partnership with Udacity, the platform allows for quick access to the information of interest. To further increase the processer model efficacy, the Udacity homepage could alter based on if an OMSCS student is logged in. Maybe the immediate display of the catalog page would be a better approach, since it's unlikely that a student is interested in additional courseware.

Predictor Model Perspective:

From the predictor point of view, in comparison with YouTube, Udacity's interface is very similar, and actually embeds YouTube's video player as their streaming platform. As such, the expectations for interactions rival each other closely to assist in the prediction of outcomes. Udacity does a good job with contrasting the objects of interest, such as the "My Classroom" tab, which is highlighted in blue, and is a point of attention for OMSCS students. Furthermore, the continue tab for viewing the most recent lecture is a separate blue tab as well. From the prediction perspective, these icons do exactly what is expected, and the environment that each icon takes you to is indicative of the expectations. One of the few flaws that I've encountered with this platform is the menu sidebar while viewing videos. In order to hide the menu bar in the video display, the user must click the icon with three horizontal lines, however, there also exists a button on the far left with an arrow that points to the left, which led me to believe it meant "collapse the menu to the left side." Unfortunately, it acts as the "back to lessons" button. While the horizontal lines are what YouTube uses as the icon for hiding the sidebar, they do not have the additional arrow which for some, may represent a collapse icon.

Luckily, the action clearly leads to the incorrect interpretation, as you're brought away from the video, and can be corrected by returning to the previous page.

Overall, for these two perspectives, it's clear that Udacity attempts to mimic popular video streaming websites for interpretability, but also maintains its own unique functions that improve the experience for OMSCS students. The processer model can improve in its efficiency by providing a separate layout for OMSCS students, which may alleviate the need for searching for the "My Classroom" tab, improving the prediction model through the knowledge of where the homepage link will take the user.

Question 2

Conceptually, in my opinion, one of the most impressive feats of HCI to date is the implementation of computer speech recognition. From automated call centers, to shopping on Amazon, the goal of computer speech recognition is to provide users with a convenient way of interacting with computers outside of the traditional keyboard and mouse. While extraordinarily complex, this interface offers users one of the most organic ways to interaction, aside from them reading our thoughts. As technology improves, we see speech recognition serve more and more spaces, each with a unique environment in which they function. Generally speaking, computer speech recognition typically struggles with generalizing statements if specific key phrases are not used, difficulty processing information in noisy environments, a noticeable delay in processing, accidental usage, language barriers, etc. These limitations constrain usage into specific conditions, but often, the payoff is a much more efficient interface to allow us to communicate what we want the computer to do, especially in an organic way that is similar to how we communicate with other humans. To touch upon the objectively most hindering drawback, generalization is probably the key characteristic that separates human from machines, and it doesn't just occur as an issue in speech recognition. Human's innately adapt their speech to the environment and context, with minimal tuning. Computers on the other hand are often hard coded to interact in a very specific manner, irrespective of the context and environment. This tradeoff is why humans have certain advantages over

machines, and it's often stated as a sacrifice between precision and generalizability. In order for machines to operate well, they require a very large dataset for the computer to retrieve information from, and our current approach in processing that information inherently disables their ability to extrapolate information, leading to both inefficiency and being use case specific.

The obvious solution to this problem would be to create a more generalized interface that can learn, similar to humans, but our technology is currently incapable of doing so. Neural nets are probably the closest we've come to creating a generalized speech recognition interface. An alternative approach may be to design multiple systems for specific environments, which then communicates with a discriminator. The discriminator may act as a "brain," which analyzes the environment in which it currently resides and select the appropriate response from the others. The brain would communicate the environmental variables to other systems, where they process what they believe is the correct statement and return those options to the brain. This would allow for the speech recognition to exist in separate specific domains but respond with context specific information. In doing so, the computer can better mimic a human and respond in a more organic manner, and that familiarity would hopefully result in a more efficient transfer of information through the interface. The more elaborate solution would be to complete ignore the need to even speak directly with the computer and have an interface that can read thoughts directly.

Question 3

The Gulf of Evaluation

Canvas, in comparison with the previous T-Squared system, is actually an extremely intuitive system and the gulf is execution offers many ways for the user to achieve their goal of submitting an assignment, while maintaining a narrow window of execution. As a result, the first two steps of the process are relatively intuitive, and Canvas also offers shortcuts which users may stumble upon. With assignment submission in mind, when the user enters the Canvas dashboard, there is a column for "To Do", and contains clearly labeled hyperlinks, which are

designated with the common hyperlink blue hue. These hyperlinks are labeled with the assignment title, the associated class, and the due date, which gives the user plenty of information to ensure the correct choice is selected. From this “To Do” column, the intentions and actions are clear and concise. The only confusion that I envision may occur is either the user doesn’t recognize that the text is a link, or if the user has a heavy blue light filter on, which may distort the text color. The alternative option to entering the submission page is to identify which class the assignment corresponds to, and to either enter the class page, or selection of the “assignments” icon on the bottom left of the box. The box for the class is sufficiently sized, and clear of its intention, however, the smaller icon is not as intuitive, nor do they stand out. For the first few occurrences, I had been actually entering the class profile through the box first, then selecting the assignments or file tabs, which are clearly labeled on the left side of the page. While this may lead to a few extra clicks, the interface does provide easily discoverable functions, which are reversible with the back key. Each profile is similar to one another, and while the smaller icons are not immediately clear, they do provide users with shortcuts to the areas of interest. Overall, these characteristics result in an easily executable interface for the user, especially for assignment submissions, where multiple options are available, discoverable, and reversible. The gulf of execution for assignment submissions is well integrated.

The Gulf of Evaluation

In regard to the gulf of evaluation, there is not much that is expected from the user for submission of assignments. The user simply wants to know if it was successfully submitted and that that the correct document was submitted. Canvas does this well by providing the user with immediate feedback, which also matches the action, stating a complete submission tagged with the time and date, and additional hyperlinks for details and access to the submission. The feedback also occurs right where the submission button takes place, which is likely where the user will be focused on. Again, a flaw that may occur is the hyperlinks, as stated above, which may either be overlooked, or altered beyond detection. To provide further feedback, the submission status remains intact after the page is re-visited, providing the user with constant feedback regarding what and when the

assignment was submitted. The only additional detail that the previous interface did to further provide feedback is an email notification regarding the submission. Personally, I did not enjoy this and thought it was too much feedback, and often deleted the email without viewing it, but some users may enjoy the additional certainty.

Question 4

The use of generalized algorithms in data science is a daily task for myself, but there exists a large gulf of execution specifically in that each algorithm often has their own syntax, and packages developed to accomplish the same task may differ in their implementation. The interface for my particular situation is Python. Due to the nature of programming, precision is needed to accomplish a specific task. With Python however, there is no standard for argument naming conventions, and the general syntax may even change within the same package. As a result, the user may often have a general idea of what the particular method does but cannot figure out how to execute their actions without either consulting the source docs or Stack Overflow. Since exact calls are required, a user cannot simply use an argument such as “n_estimators” for a method if it is expecting the term to be just “n”, even though either option is reasonable for the user, it is not for the interface. Other instances can lead to even more frustration, such as when an assumed standard is not implemented from package to package. Instances such as how opencv converts all color schemes to BGR, as opposed to the common RGB, which is used in other major packages, such as matplotlib. This implementation difference can go unnoticed until late in development and may shift unexperienced users on a tangent. While the expectation is not for the interface to become more generalizable, the interface can be improved to include standards across packages.

It pains me to admit that R is a lot more diligent with its standardization of packages, especially within the tidyverse methodology. The goal of tidyverse in R is to create a standardization of packages to “share an underlying design philosophy, grammar, and data structures” (*tidyverse*). As a result of this methodology, situations like the ones found with Python above are often

mitigated. This allows for users to be more flexible and able to apply their experience into correctly selecting the appropriate action. In short, tidyverse aims to shrink the gulf of execution by standardizing the actions needed to accomplish a particular task, allowing a more efficient use of the interface.

Python can learn from the design methodology of tidyverse by implementing a standard to be followed, especially in niche spaces such as data science. The standardization of design would be extremely helpful, so that the underlying structure of the algorithm can correctly be assumed, such as the case with the color display order. It would also be useful to assume a certain naming convention for similar classes of methods, instead of having to refer to the documentation; at least for the simpler arguments. Due to the nature of programming, it's important for users to know precisely what actions they have in various situations. Especially in the high-level nature of Python, a lot of ambiguity exists, which leads to an inefficient interface for accomplishing tasks. The standardization of packages may aid in the development of Python through the shrinking of the gulf of execution.

References

1. R packages for data science, *tidyverse*, https://www.tidyverse.org/