A BEGINNERS GUIDE TO SSI

Nathan Reiff

Introduction to SSI

Hello! If you're reading this, then you're probably like me: you're trying to learn Social Signal Interpretation (SSI), but have little coding experience. SSI can be complex and difficult to learn even with coding expertise, so don't worry if it makes no sense at first. When I started to learn SSI, I knew a bit of HTML, CSS, and JavaScript, but nothing that was directly related to SSI. Even after 10 weeks of attempting to work with SSI, there are still many parts that are unclear to me. So, through this documentation, I hope to speed up the process of learning the underpinnings of SSI in the hopes that future non-coders will be able to excel faster and farther than I was able to.

In basic terms, SSI syncs multiple 'pipelines' of data collection for easy visualization and analysis of multi-modal data. In the context of tiilt's work, the key concept here is referred to as Multi-modal Learning Analytics (MmLA). MmLA is used to study learning in and out of classrooms using sensors like the ReSpeaker, Pozyx, Empatica, MYO and Tobii Eye-Tracker. Together, these tools compose the multi-modal portion of this data, as they all provide researchers and educators with different forms of data (audio, video, physiological, location). SSI comes in when researchers are using these tools together. SSI is powerful because it is able to synchronize data collection between all of these tools to provide researchers with a cohesive picture of all of the data from each sensor at any given moment.

Before starting to work on SSI, I was working on a project aimed at using ReSpeakers to capture audio data to provide educators with real-time multi-modal learning analytics regarding student discussions in their classes. As the project developed, we found ourselves adding another mode of data collection: video. Instead of trying to manually sync the two data streams, we can use SSI to easily and reliably sync the data collection. This is one of many applications SSI has to research endeavors.

Online Documentation

The first thing to do when trying to delve into SSI is to check out the official online documentation. This can be found at: https://rawgit.com/hcmlab/ssi/master/docs/index.html. The documentation gives an in-depth explanation of SSI, which can be overwhelming and confusing for someone with no coding background. So, I'm going to give a brief explanation of some key components of a SSI pipeline (still read the entire documentation though).

A pipeline is a document written in Extensible Markup Language (XML) that contains the components which tell SSI what data to collect. XML is an easily usable and readable programming language that is similar to HTML, so don't worry about learning it. There are three

main components that build a pipeline: a sensor, a transformer, and a consumer. The sensor is used to produce a data stream. This section tells SSI what is being used to collect data and how it works. You can find already made sensors and the components that can be added onto them in the SSI API, which I will discuss in more detail later. One example of a sensor you could use is a Mouse sensor. This provides data on the position and button state of the mouse. Check out the official documentation under section 4.1.3 for the actual coding structure of the mouse sensor.

After the sensor, a pipeline needs a transformer. In layman's terms, a transformer takes the data collected by the sensor and converts it to an easily readable form for the consumer. For example, transformers can be used to change the rate at which SSI consumes data. This is especially useful if the streams are not fully synced. In my opinion, transformers are the most complicated component of SSI, but read the documentation and don't worry too much about fully understanding them (sensors and consumers are much more important).

Finally, a pipeline has a consumer. Consumers allow us to access the data that was collected and stored by the sensor. Consumers, such as SignalPainter can visualize data and allow for easy access to synchronized data streams. Like with sensors, you can use the SSI API to see a wide variety of pre-made consumers and the components that can be used with them.

Along with the official SSI documentation page, the SSI API is another extremely helpful resource, which can be found in the SSI GitHub repository (https://github.com/hcmlab/ssi). The API has a list of all of the pre-made sensors and consumers (and some other things). Under each sensor and consumer, the API also lists components that can be added to those sensors and consumers. For example, the SignalPainter consumer has components such as title, pos, and type. These components define the title of the generated graph, the position/size of the graph on the screen, and the type of graph that is generated. These components are a helpful way to customize data collection and visualization. Clicking through the API to see what is already made is a helpful way to understand the groundwork of an SSI pipeline.

Python & SSI

The pre-made sensors and consumers on the API site are a great way to get started with SSI, but sometimes you'll want to create a custom sensor from scratch. To do this, you have to write a Python script that can be passed to SSI to tell it what to record. Going into learning SSI, I had no Python experience. Here, I'm going to tell you about some resources I used to learn Python.

The first resource is Dive Into Python. You can get this from Amazon or find it free here: https://www.cmi.ac.in/~madhavan/courses/prog2-2012/docs/diveintopython3/index.html. This book is an extremely detailed and helpful resource for beginning to learn Python. The nature of the Dive Into Python website also allows for readers to easily find concepts that they need a quick refresher on.

While Dive Into Python is a great resource, I found it difficult to actual learn to program with Python by just reading on a screen. So, I mostly used Codecademy to learn Python. I already had a Codecademy subscription, but if you don't there's a free trial that you can use to see if it's the right fit for you. On Codecademy, I used the *Analyze Data with Python* course. This provided me with helpful information on the basics of Python and how to use it to analyze large data sets. In order to write a custom sensor with Python, you only really need to know functions, loops, conditionals, strings, and dictionaries. With Codecademy, it is easy to quickly learn these concepts with interactive lessons that really ensure that you understand these key concepts. Then, you can always refer back to Dive into Python when you need to remember a little detail about how to do something. I would also suggest taking detailed notes when doing Codecademy. The nature of the lessons make it seem like you'll remember how to do everything, but I found that taking notes really solidified the material for me and gave me another resource to refer back to when needed.

Tutorials

After you learn the basics of Python, you can find SSI tutorials on the GitHub repository. Simply reading through these will give you a clearer picture of how to build and run pipelines on your SSI machine. For example, the first tutorial (Hello World) walks users through a step by step process of building a pipeline, writing it to the disk, adding a transformer, adding and running events, and sharing data. From there, the tutorials get more and more complicated and really give you a sense of the architecture of SSI. The Python tutorial is particularly helpful in understanding Python's connection with SSI and how to use it properly. I found it useful to copy the example code from the tutorials into my own documents. Writing them out myself really helped me understand the structure of the XML code and allowed me to dissect it on my own time.

Application to Research and Learning Sciences

I touched on this a bit earlier, but it is easy to see how SSI can be used to ease data collection in the study of learning. One example that we recently studied was video footage from an after-school program for high schoolers. This video footage was shown at MmLA for IA, a workshop hosted by tiilt. As a group, we were conducting interaction analysis on the video footage. At one point, a participant in the video stepped back and put his hands behind his back. This seemed to be an important gesture for the group and we spent a while talking about what it might mean. Some thought it showed uncomfortability, others thought it was anger. There was a whole conversation about the meaning of the gesture, but no consensus was come to.

Now, say we added in ReSpeakers to capture audio data. This could tell us things like keywords the participant had said, how long he had been talking for, and, most importantly,

discussion features from word choice and prosodic tones. This data might tell us that he had high anger, confusion, analytic thinking, or any other of the features available through the Harvard Inquirer or the Linguistic Inquiry and Word Count (LIWC) library.

Next, if we added more cameras around the room to capture different angles, we could use pose estimation to track his body movements. This may tell us if the arms behind the back movement is regular for this individual, or other data that might shed light on his gesture.

Finally, with a sensor like the Empatica, we could track physiological factors, like heart rate and body conductivity. This data could hint at emotional responses and reactions to the conversations he was having, which were highly charged.

Now, with all of these various sensors in play, SSI can be used to synchronize the data streams. Thus, we could look at aggregated data for the exact moment the student steps back and puts his arms behind his back. We still would not be able to know the exact emotional response he was feeling at that moment, and that may never be possible. However, we definitely have more data for that moment in time than we did with just video footage. Understanding these minute instances in vital to researching students' learning paths and responses. Thus, SSI and MmLA gives researchers a much clearer window into the bodies of students and has a large potential to change the way Learning Sciences (and other disciplines) research is conducted.