Gesture

An app to translate sign language into spoken word

Student: Scott Sievert *Advisor:* Jarvis Haupt

Contents

List of Figures

Abstract

The complete backend for our project has been developed, including communicating with both armbands simultaneously and moving data to a machine suitable for prototyping. We have cleared humans providing data with the IRB and have connected with several individuals who use ASL on a daily basis. We have a clear picture of what to do next and what algorithms to use and have installed the libraries necessary for this. The algorithm that seems to be most appropriate with certain simplifying assumptions are convolutional neural networks.

1 Introduction

This project involves translating sign language to spoken word through a smartphone app with associated hardware. A complete diagram of the system can be found in Figure 1.

2 Backend

Hardware is needed to obtain the data of interest and comes in the form of armbands. These armbands provide limited EMG data, acceleration and orientation data and communicate with a smartphone via Bluetooth.

To even communicate with the armbands, a smartphone app had to be developed. This smartphone app is only used to collect data and does not perform any advanced computation. The data collected is either individual signed words or a complete signed sentence.

In order to collect this data, we had to communicate with the Myo armbands via Bluetooth. This required that we conform to a third party API. These armbands are very new meaning the API is not completely stable and there were certain changes that had to be adapted to. Additionally, communicating with one armband was relatively easy and had extensive examples but communicating with two armbands had far less documentation and fewer examples.

This project is a prototype as it naturally should be and prototyping on a mobile platform is far from ideal. While the device may be capable of complex computations, it does not have other infrastructure necessary for prototyping. Capabilities to plot, easily inspect variables and rich library support are all missing on mobile platforms.

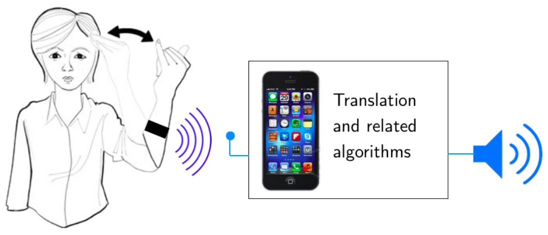

Figure 1: A diagram of our project complete with armbands and smartphone app.

To circumvent this, we have allowed for transmission of raw data between the smartphone and computer. This is possible by writing a file to the iOS file system then using the “File Sharing” feature of iTunes to transfer these files across.

The language of choice while developing on a personal laptop (or possibly a server) is Python. This is a high level language similar to Matlab with similar prototyping capabilities. It is easy to visualize data or inspect variables. Additionally, Matlab and Python with their low-level libraries are roughly an order of magnitude faster then the seemingly underpowered mobile device.

It should be noted that the long term vision of this project is to eventually port this prototype to a mobile platform. The current state of technology is ready for this as it is common to carry powerful and small computing devices in pocket. However, porting this algorithm to a mobile device is in the distant future as the prototype needs to be completed.

3 Human data

This experiment involves collecting data from humans and therefore required IRB approval. While this project does not require IRB approval, we still had to check with the IRB to ensure it didn’t.

Additionally, we felt that it would be best if we obtained data from a deaf person as that is the population we’re trying to help. To do this, we had to talk with the Disability Resource Center (DRC) and have linked with two deaf individuals, John Wilson and Kaity Hagen.

This process was more involved than first thought. I expected it to be simple and easy but have since learned that this process is very involved.

4 Previous work

At first, automatic sign language recognition (ASLR) seems similar to automatic speech recognition (ASR). Both involve computing translating some signal into text or spoken word. However, the hardware used to collect sign language data is present on 12 different channels. The algorithm behind modern ASR, the hidden Markov model (HMM) has been generalized to multiple input channels [1].

The biggest challenge in ASLR is that one sign can correspond to many different words, invalidating the assumption behind HMMs. Previous work has been done on ASLR but they had many limitations [2]. For example, one system could effectively translate only if there were short sentences that followed a very rigid syntax.

The systems mentioned in [2] tended to rely on speech recognition concepts, especially the algorithms of modified HMMs and neural networks. ASLR is a very difficult problem with many constraints (e.g., not ASLR to English is not 1-to-1) and accordingly the algorithms used had modifications from versions used in ASL.

5 Next steps

We do not want to take a giant magic step and have the entire project rest on the outcome of one algorithm. Instead, we want to take small steps and gradually improve the algorithm. As part of this, we will assume that sign language motions correspond one-to-one with spoken words. As time goes on, we hope to further expand upon this and include a more natural translation that does not make the assumption of a one-to-one translation.

Since sign language data has not been obtained, I am currently tackling a similar problem of digit recognition. Digit recognition is similar to ASLR by having multimodal inputs that can be shifted in time (or space). According to my advisor, convolutional neural networks (CNNs) are best suited for ASLR because CNNs are shift invariant.

For CNNs to be fast to be effective and require use of the GPU and low level mathematical libraries and just-in-time compiled functions. This requires that a third party library is used. Several libraries and languages were looked at including Torch7 (Lua), Caffe (Matlab/Python) and Theano (Python). I attempted to install all of these libraries and eventually settled on Theanets, a library built on top of Theano that can also include other customization as necessary.

The first small step is to use this library to classify the handwritten digits using CNNs, a problem similar to ASLR through matrix inputs and shift-invariance necessity. This will also ensure that the library is practical for our purposes. This is currently being developed.

6 Timeline and budget updates

There are no updates to the budget.

The backend is done and work on the algorithm will begin. We expect this to be done in mid-April with time enough to polish the end result and write the final report.

7 Conclusion

The entire backend for this project has been developed. Both armbands can be communicated with simultaneously via an iOS app, the information can be transferred from iOS to a local machine for prototyping purposes and we have connected with individuals who use ASL on a daily basis.

The developed product will be a prototype and only demonstrate ASLR. The final product will likely be further polished and presented as a mobile app. This will not be a trivial undertaking and will require significant time and effort to port the algorithm of choice to the mobile platform.

References

[1] Samy Bengio. Multimodal speech processing using asynchronous hidden markov models. *Information Fusion*, 5(2):81–89, 2004.

[2] Sylvie CW Ong and Surendra Ranganath. Automatic sign language analysis: A survey and the future beyond lexical meaning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(6):873–891, 2005.