Model Analysis Tool for Sign Language Video Recognition and other HAR Tasks

Ishan Chadha

Table of Contents

Introduction 3

Literature Review 4

- Sign Language Annotation Tools 4
- Forced Alignment 4
- Complex Model Visualization 5
- HAR Task Analysis 5

Design 7

Results 8

- Recognition System 9
- Experiments 10
- Conclusions 10

References 11

Appendix A 12

Appendix B 14

Introduction

Currently, over 95% of deaf children are born to hearing parents, many of whom do not learn sign language. Lack of communication between deaf children and their parents can lead to language deprivation syndrome (LDS), where the child does not develop important language skills such as short-term memory and basic communication abilities. The CopyCat project aims to address the learning gap that stems from LDS by designing a video game that accepts videos of deaf children signing simple phrases and employs computer vision models to recognize the phrases being signed and verify that they are correct. The task of American Sign Language recognition (ASLR) fits into the broader field of human activity recognition (HAR), which focuses on predicting and comprehending sensor readings from a movement or pattern generated by humans. A key problem in HAR is selecting features that are representative of different patterns, where features are hand, face, and body pose elements such as fingertips or eyes that help to distinguish signed phrases. A tool that helps to select features characteristic of individual words is crucial for developing a robust HAR framework, especially when distinguishing commonly confused signs such as “in” (right hand moves into left hand) and “above” (right hand hovers over left hand).

Elan is a powerful visualization tool that has been utilized in past sign language visualization tasks, so it was applied to the construction of a feature selection tool for HAR tasks. This feature selection tool displays log likelihoods for each feature based on models and Gaussians passed into the framework as well as the video itself to find where feature selection issues are occurring. An example of this is the signed words “in” and “above” being confused since they are very similar in their motion; the visualization tool can help to find a certain feature, such as the nondominant hand’s fingertips, that allows those two signed words to be distinguished.

The ELAN tool can be generalized to other HAR tasks since feature selection is key for locating distinct characteristic patterns for recognition. However, the focus of this thesis is to develop functionality for the ELAN tool to improve feature selection for the CopyCat project, as visualization of the features being used in the models will allow for better manual feature selection as well as easier implementation of alternative feature selection methods and metrics.

Literature Review

This study investigates possible diagnostic tools for sign language recognition and verification models, especially those used for visualization and annotation of sign language. The overall goal is to use these findings to design a plugin for troubleshooting the CopyCat project's current issues as well as select the best features at each step based on algorithmic analysis. Since modeling sign language videos employs aspects of both computer vision and speech recognition, metrics from both fields should be used in conjunction during their analysis.

Sign Language Annotation Tools

Currently, three outstanding platforms used in sign language annotation are Elan, SignStream, and Noldus Observer, all of which have individual benefits. Elan hosts functionalities including enabling the creation of tiers and annotation levels for different hands, features, or however else the researcher wants to segment data. Elan also allows the researcher to display a side panel of statistics which can be configured programmatically. Lastly and most importantly, Elan provides the researcher with the ability to edit the source code, which means that a plugin could be designed to incorporate even more features than the ones listed [3]. SignStream was designed in the late 1990s for the specific purpose of making sign language annotation easier on a project for applying computer vision to the task of sign language video annotation. A major advantage of SignStream is the abundance of features that correspond directly to the task at hand, such as the ability to separate out all occurrences of a certain sign. However, one key issue is that this system is only supported on MacOS, whereas the vast majority of CopyCat is built on a Linux system [7]. Noldus Observer XT is a tool designed for integrating and analyzing multiple data streams and supports extensive annotation of videos [4]. Despite being designed for research, though, Noldus Observer XT requires payment and does not have open source code, which makes it less malleable in the case that more features are needed. Based on the barriers to entry for SignStream and Noldus Observer XT, the best tool for this analysis is Elan.

Forced Alignment

Forced alignment is an example of a metric widely used in the field of speech recognition, alongside more basic ones such as accuracy and loss. It is a common method utilized in speech recognition used "for aligning orthographic transcriptions to audio recordings;" an example implementation can be found in the Kaldi toolkit. In the scope of this project, forced alignment can be used to locate the sign boundaries within each video and compare them to the phrase segmentation that the models are outputting [1]. In his Master's Thesis, Benovic compares three forced alignment approaches: HMM-GMM, deep learning, and hybrid. The HMM-GMM approach utilizes hidden Markov models and Gaussian mixture models, where hidden Markov models are a mathematical representation of probabilistic time-series data, and Gaussian mixture models are combinations of discrete and continuous

probability density functions. Benovic found that the hybrid approach worked best, and briefly discussed how factors such as accent and gender were not controlled during his analysis [1, 4, 6]. His work was based on creating an application for developing forced alignment tools for YouTube videos, but it focused too greatly on building the codebase and application surrounding rather than the forced alignment analysis itself. Factors of accent, gender, fluency, and age might need to be considered as well. In comparison to other application-based papers in this literature review, Benovic's work was neither very theoretical, nor was it describing a mature framework, which caused the thesis to lack information about why functions were included as opposed to how they were constructed.

Complex Model Visualization

Further exploration into approaches for implementing hybrid models such as HMM-GMM and HMM-DNN show the need for a model topology visualization element to this tool [8]. Swietojanski *et al.* examine how HMM-GMM and HMM-DNN methods of automatic speech recognition on a dataset of TED Talks work in a complementary manner. The HMM-GMM model is designed similarly to that of Benovic's Master's Thesis, but the HMM-DNN model uses a deep neural network to estimate the likelihoods of context-dependent parameters in the hidden Markov model [1]. The final system in Swietojanski et al. uses a form of model averaging which is more complex in structure than an HMM-GMM by itself, which introduces the need for model visualization. Tensorflow Graph Visualizer, used as an accompaniment to the open source machine learning framework from Google called Tensorflow, demonstrates a powerful yet relatively simple framework for visualizing data flow through different machine learning architectures [9]. This framework applies graph construction techniques including building directed graphs in a hierarchical way to represent deep learning models, which will be useful for displaying model architecture through annotations in Elan.

HAR Task Analysis

It is difficult to determine which elements of speech recognition and computer vision analysis should be applied to the task of sign language recognition. By analyzing the different angles from which we can view the task of sign language recognition, we can understand the most relevant features. Hiremath and Ploetz examine various HAR tasks and objective measures of their complexity. Although calculating the complexity of a HAR task does not have a direct connection with feature selection, metrics such as linear separability of features can help in determining correlated features, and techniques such as k-means clustering or computing the minimum spanning tree among data points can give an idea of which features are most relevant. Overall, Ploetz and Hiremath conducted a thorough analysis of the problem they were tackling, and their methods are relevant to the task of sign language recognition [5]. Cisotto *et al*'s work, in contrast, looks at the issue of feature selection for gesture recognition problems directly. Their work depicts an algorithm called consensus clustering, which consists of using two separate

clustering algorithms, followed by the use of a "consensus" filtering and more clustering. The paper does a sufficient job of both describing how the algorithm works and why the different steps were chosen. While the study's chosen data, EEG and EMG waves, are not very similar to the features for sign language video processing at first glance, they can all be fashioned as two-dimensional time-series data waves, where EEG and EMG data uses electric signals and sign language data uses X and Y coordinates of keypoints on the hands and face. One aspect that is slightly unclear in this paper, though, is the significance of the different uses of cross validation, as there is both an inner and outer iteration. A similar approach to the one described in Cisotto *et al*'s paper is used in this project since feature selection is a key indicator of performance of the models and can improve performance of models, so it is a type of metric. In their paper, they discuss how selecting fewer features for gesture recognition tasks would significantly "minimize data traffic over a communication network;" similarly, reducing the feature set for sign language video analysis would improve real time verification, which is the end goal [2].

Based on the literature in both sign language visualization tools and activity recognition metric calculation algorithms, the task of designing a diagnostic tool for the task of sign language recognition is not only possible but can contribute to existing tools. Cisotto *et al.* states that the pipeline constructed for feature selection did not perform as well as hoped for due to lack of data, and Ploetz and Hiremath's paper only attempts to quantify the complexity of two HAR task classes. This means that creating a diagnostic tool would not only help the CopyCat project itself but also other activity recognition frameworks and can possibly be generalized to other activity recognition problems [2, 5].

Design

The design of this human activity recognition (HAR) analysis framework has two requirements: displaying CopyCat data in ELAN and investigating if this improves model performance.

Annotation and Feature Viewers

The first requirement for the design of the HAR analysis app was to ensure that necessary functionality is included. The first component is converting a file containing state boundary and transition information to annotations in ELAN, or the "Annotation Viewer." Figure 1 shows how the state and word boundaries for CopyCat are displayed when words are loaded in different tiers and states are loaded as annotations on their respective tiers. States are labelled with the letter "s" and their order number; for example, the fourth state for the word "blue" in the phrase "alligator above blue wagon" is shown on the fifth tier with the label "s4" in Figure 1.

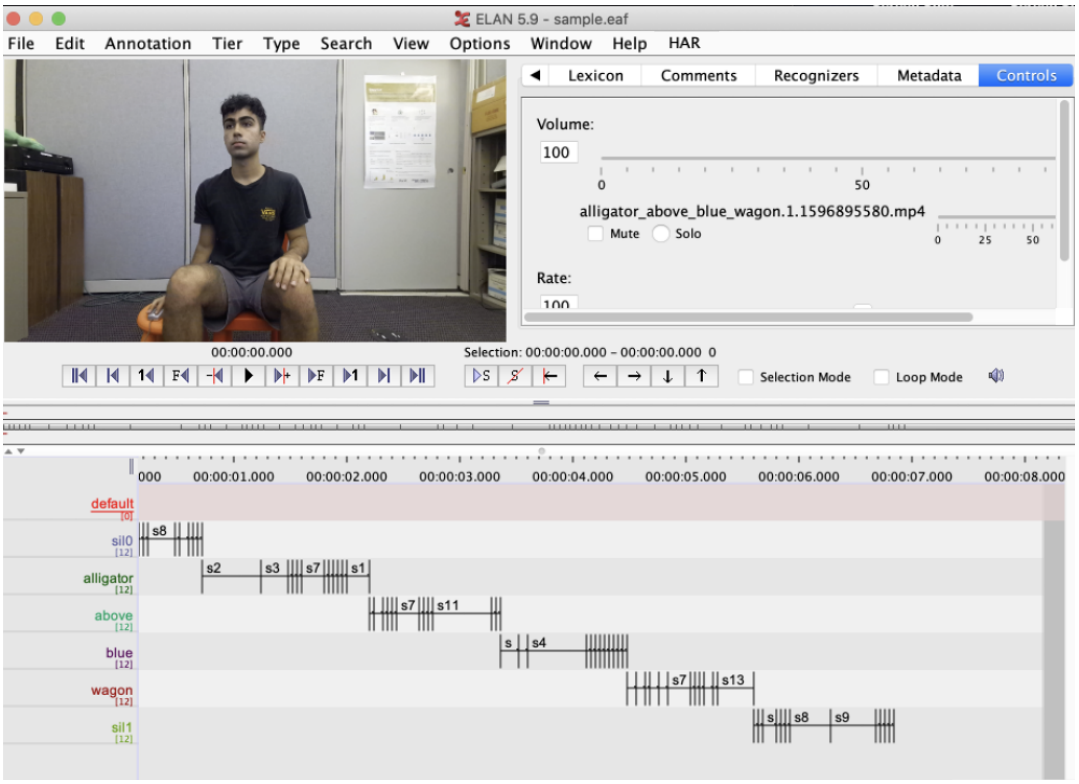

Figure 1. Word and State Boundaries displayed in the Annotation Viewer.

In Figure 2, the timeseries viewer is shown with its "Feature Viewer" functionality; essentially, features values as well as other metrics can be plotted alongside the video and its models' states. In order to test this on CopyCat, numerous scripts were written in order to convert the feature data into log-likelihoods of those features occurring based on the models. Given a feature x at state s, a log likelihood for that feature occurring is shown in Equation 1, where μ_s and σ_s are the mean and standard deviation of that feature for the given state. These log likelihoods were plotted for the words "in" and "above" across every feature for all videos. Features were then ranked based on which feature had the highest difference in median of log

likelihood across all videos. The results of this feature selection process are shown in the results section of this paper.

$$ll_x = log(\frac{exp(-x^2)}{\sqrt{2\pi}})$$

Equation 1. Log likelihood of a feature occurring given model parameters.

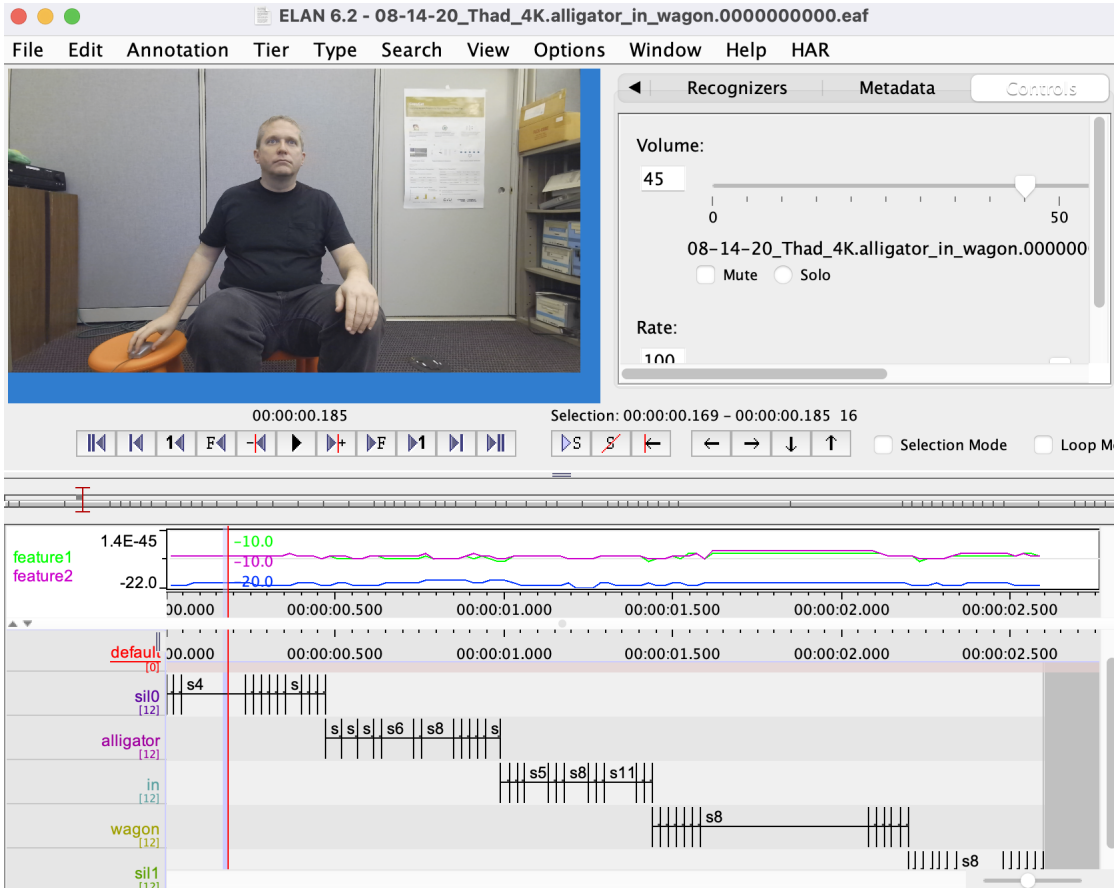

Figure 2. Timeseries Display for Three Example Features.

With these elements, the HAR analysis plugin provides useful insights for tasks such as sign language recognition - as CopyCat works with - or other timeseries analysis tasks such as heart rate tracking or signal processing.

Results

In order to show how feature selection worked, the CopyCat project's recognition system was tested twice - once with previous feature sets, and once with a feature set selected based off of visualizations from the ELAN tool.

Recognition System

Experiments were run on the CopyCat recognition system, which records a video of a user and recognizes what the user is signing in ASL. A standard experiment takes in numerous parameters relating to how the data is processed and what kind of model should be run. The recognition system was set up as follows for the two experiments.

First, the recognition system employed MediaPipe for pose estimation, extracting raw body pose features such as left and right eye position [6]. Many of these features were dropped based on what is specified in a separate configuration file; this file specifically was altered during the feature selection experiments described later on. After this, a hidden Markov model (HMM) was trained for each word, with each model containing a left-to-right topology of 14 states with no skip transitions and 10 Gaussian mixtures approximating the distribution at each state for each feature. The training commenced with a flat initialization of all parameters, followed by five iterations of Baum-Welch re-estimation. After the Baum-Welch re-estimation, Viterbi Decoding was used to determine the HMMs that produced the highest likelihood given observed feature values.

Data from three different users was passed in with the dataset for each user containing around 350 videos. In order to test the trained HMMs for each experiment, three-fold cross-validation was utilized, where one of the three users was left out during each fold. This dataset was collected from users that previously volunteered their time to be recorded using the Azure Kinect and will remain anonymous. The dataset is publicly available here: https://github.com/Accessible-Technology-in-Sign/copycat-data-chi

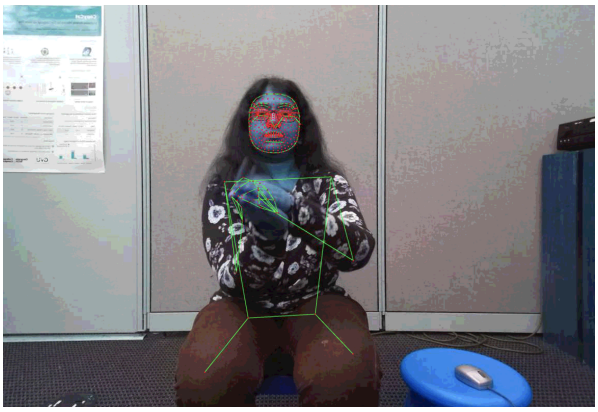

Figure 3. MediaPipe Pose Estimation.

Experiments

To determine the best feature set to distinguish "in" and "above," the log likelihood of a feature occurring at any given time given the trained model from the first experiment were computed for every feature and plotted for all phrases containing the words "in" or "above." After that, the median log likelihood was stored for each feature for the words "in" and "above." Then, the differences in median log likelihood between words "in" and "above" were computed for each feature. Appendix A shows the ranked derived features as described. Based on the rankings, it is evident that features related to the change in motion of the right hand have low log-likelihood differences indicating increased confusion between the two words, so these were removed; Appendix B shows the old feature list as well as the edited feature list. The second experiment was run to test whether the newly selected features improved the correct classification of the word "in."

	Experiment 1	Experiment 2
Classification Accuracy of "in"	.552	.554
Misclassified Proportion of "in" as "above"	.211	.149

Table 1. Classification Accuracy of "in" versus Confusion of "in" with "above".

Conclusions

Based on the results of the feature selection experiments, the accuracy of recognition of "in" in words did not significantly increase, but the misclassification of "in" as "above" did decrease from 21% to 15% misclassification. These experiments show the capacity of this tool to be utilized in the future for feature selection. This tool also gives users the ability to visualize the boundaries of a hidden Markov model's states, which means that users can tell if certain states of a word's model are more important in distinguishing that word than other states so that they can assign a greater weight to that state in the model. This idea, known as state boosting, should be explored to improve the performance of HMMs for the task of sign language recognition

This project has contributed to the visualization of data in the field of human activity recognition, allowing for timeseries data and model/state data to be mapped out in the ELAN tool, given that they are in the correct format. For formatting tools and the general project setup, all of the code utilized is available at the following public source repository: https://github.com/ishanchadha01/CopyCat-HTK.

References

1. Benovic, M. *Forced Alignment via Neural Networks* . Prague, 2020. Diploma thesis. Charles University, Faculty of Mathematics and Physics, Department of Distributed and Reliable Systems. Thesis supervisor Kofroň, Jan.
2. Cisotto, G., Capuzzo, M., Guglielmi, A.V., & Zanella, A. "Feature selection for gesture recognition in Internet-of-Things for healthcare," *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, Dublin, Ireland, 2020, pp. 1-6.
3. Crasborn, O., & Sloetjes, H. (2008). Enhanced ELAN functionality for sign language corpora. In *Proceedings of the 3rd Workshop on the Representation and Processing of Sign Languages: Construction and Exploitation of Sign Language Corpora* (pp. 39-43).
4. Ducharme, D.A., Arcand, I. Using Noldus Observer XT for research on deaf signers learning to read: An innovative methodology. *Behavior Research Methods* 41, 833–840 (2009).
5. Hiremath, S.K. & Plötz, T.. 2020. Deriving Effective Human Activity Recognition Systems through Objective Task Complexity Assessment. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. 4, 4, Article 131 (December 2020), 24 pages.
6. Lugaresi, Camillo, et al. "MediaPipe: A Framework for Building Perception Pipelines." *ArXiv:1906.08172 [Cs]*, June 2019. *arXiv.org*, http://arxiv.org/abs/1906.08172.
7. Neidle, C., Sclaroff, S. & Athitsos, V. SignStream: A tool for linguistic and computer vision research on visual-gestural language data. *Behavior Research Methods, Instruments, & Computers* 33, 311–320 (2001).
8. Swietojanski, P., Ghoshal, A., & Renals, S. "Revisiting hybrid and GMM-HMM system combination techniques," *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 6744-6748, doi: 10.1109/ICASSP.2013.6638967.
9. Wongsuphasawat, K. *et al*., "Visualizing Dataflow Graphs of Deep Learning Models in TensorFlow," in *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 1-12, Jan. 2018, doi: 10.1109/TVCG.2017.2744878.

Appendix A

Derived Feature Names	Difference in Medians between Log Likelihood for “in” and “above”
delta thumb left to nose y	2.8029862
delta hand left to nose y	2.7684686
delta handtip left to nose y	2.66457228
delta wrist left to nose y	2.48273314
delta elbow left to nose y	1.91186298
delta wrist right to nose y	1.60134778
delta handtip right to nose x	1.39228694
delta thumb right to nose x	1.32882823
delta hand right to nose y	1.19749331
delta handtip left y	1.11814567
delta thumb left to nose x	1.03648656
delta hand right to nose x	1.03452863
delta elbow right y	0.93705755
delta thumb left x	0.79280365
delta wrist right to nose x	0.77067687
delta elbow left y	0.75620695
delta thumb right y	0.63472446
delta wrist left to nose x	0.63152984
delta hand left to nose x	0.60202106
delta thumb right to nose y	0.55113822
delta wrist left x	0.51854211
delta thumb right x	0.48515404
delta elbow right x	0.48310083
delta hand right y	0.42080042
delta handtip right y	0.41539807
delta hand right x	0.34979547
delta handtip left x	0.32121592

delta hand left x	0.31749068
delta thumb left y	0.27441119
delta handtip right x	0.12586787
delta elbow right to nose x	0.12421918
delta elbow right to nose y	0.10361924
delta wrist right x	0.10127393
delta handtip left to nose x	0.08124838
delta wrist right y	0.07963044
delta hand left y	0.07863644
delta elbow left to nose x	0.04614015
delta elbow left x	0.04035305
delta wrist left y	0.03840295
delta handtip right to nose y	0.02056878

Appendix B

Old Base Features		**New Base Features**	
Right Shoulder X	Delta Right Shoulder X	Right Shoulder X	Delta Left Shoulder X
Right Shoulder Y	Delta Right Shoulder Y	Right Shoulder Y	Delta Left Shoulder Y
Right Elbow X	Delta Right Elbow X	Right Elbow X	Delta Left Elbow X
Right Elbow Y	Delta Right Elbow Y	Right Elbow Y	Delta Left Elbow Y
Right Wrist X	Delta Right Wrist X	Right Wrist X	Delta Left Wrist X
Right Wrist Y	Delta Right Wrist Y	Right Wrist Y	Delta Left Wrist Y
Right Pinky X	Delta Right Pinky X	Right Pinky X	Delta Left Pinky X
Right Pinky Y	Delta Right Pinky Y	Right Pinky Y	Delta Left Pinky Y
Right Index X	Delta Right Index X	Right Index X	Delta Left Index X
Right Index Y	Delta Right Index Y	Right Index Y	Delta Left Index Y
Right Thumb X	Delta Right Thumb X	Right Thumb X	Delta Left Thumb X
Right Thumb Y	Delta Right Thumb Y	Right Thumb Y	Delta Left Thumb Y
Left Shoulder X	Delta Left Shoulder X	Left Shoulder X	Delta Horizontal Hand Distance
Left Shoulder Y	Delta Left Shoulder Y	Left Shoulder Y	Delta Vertical Hand Distance
Left Elbow X	Delta Left Elbow X	Left Elbow X	
Left Elbow Y	Delta Left Elbow Y	Left Elbow Y	
Left Wrist X	Delta Left Wrist X	Left Wrist X	
Left Wrist Y	Delta Left Wrist Y	Left Wrist Y	
Left Pinky X	Delta Left Pinky X	Left Pinky X	
Left Pinky Y	Delta Left Pinky Y	Left Pinky Y	
Left Index X	Delta Left Index X	Left Index X	

Left Index Y	Delta Left Index Y	Left Index Y	
Left Thumb X	Delta Left Thumb X	Left Thumb X	
Left Thumb Y	Delta Left Thumb Y	Left Thumb Y	
Horizontal Hand Distance	Delta Horizontal Hand Distance	Horizontal Hand Distance	
Vertical Hand Distance	Delta Vertical Hand Distance	Vertical Hand Distance	