2D and 3D Convolutional Neural Networks for Real-Time Classification of Dynamic American Sign Language (ASL) for Lightweight Applications

Kol A. Crooks and Sean R. McHale

Joel Barlow High School

KEYWORDS: Computer Science. Machine Learning.

ABSTRACT: Using a 2D and 3D convolutional neural network (CNN) this system classifies American Sign Language (ASL) that accounts for the temporal nature of ASL's dynamic gestures. To accomplish this, the system has a web-based application that streams video into a backend server that optimizes the video via pre-processing and then classifies it with the 2D and 3D CNN model after which it streams the predicted word back to the client. The data set used for training is scalable and can be altered in size and labels, allowing words to be easily added to the 'dictionary' of signs. This model of machine learning, and machine learning as a whole, has aided in advances in areas such as detecting written language and audio with high levels of accuracy. Currently, minimal progress has been made in visually recognizing ASL. Many pre-existing research projects demonstrate multiple methods for sign language detection, such as special gloves with sensors for the signer to use, utilizing a Microsoft Kinect or depth detecting cameras to improve feature identification, and 2D convolutional neural networks for static gestures. All these innovations were developed to decrease the language barrier; however, they lack efficiency and mainstream functionality. This project builds upon these studies to examine the effectiveness of a sign language detection model designed for lightweight applications. The model that has been created enables the analysis and classification of dynamic ASL in real time without the cumbersome hardware used in previous studies. Currently, with a dataset of three signs, the system has high levels of accuracy and shows promising results for future development.

I. INTRODUCTION

There are roughly 250,000-500,000 Americans that use ASL. Communication for the deaf is difficult and presents numerous challenges, especially with people with no hearing impairments and the inability to understand sign language. To successfully communicate both parties need to know sign language, and in a culture based on convenience, many do not choose to learn a language which can take years. With the ability to detect signs, communication will be eased between the society of signers and people with no hearing impairments.

This project analyzed a series of studies and builds off them to achieve real-time classification of dynamic American sign language (ASL).The early sign language recognizers involved sensors and gloves. In the study Auslan Sign Recognition Using Computers and Gloves, a glove for the signer is used to recognize sign language. Though able to produce accurate measurements and results, these gloves are expensive and wouldn't create a logical solution for wide-spread sign language identification.

So, for the last two decades, researchers have been using linear classifiers, neural networks, and Bayesian networks to aid in the recognition of ASL. These processes require pre-processing which is very intensive and don't allow for real-time translation as the video needs to be analyzed separately to recognize features. But, in recent years, ASL recognition has evolved. In the study, Real-time American Sign Language Recognition with Convolutional Neural Networks researchers at Stanford have created an application which can identify signed letters in real-time, excluding 'j' and 'z' as those require hand movements to sign. These signs are static and use CNNs to recognize. The main advantage of using CNN is its ability to learn features. However, like many other sign language recognizers, it uses depth.

Many studies use the Microsoft Kinect, Sign Language Recognition with Kinect, to identify the depth of features for aiding in recognition. The Kinect allows researchers to get full RGB video and using its depth indicators, track the full body independently from lighting conditions. However, the Kinect has some limitations, most notably the Kinect has issues in recognizing single fingers. For the project, depth and RGB video is not used. To avoid excess GPU usage the project uses gray-scale videos and meshing together a series of frames with isolated movements to avoid using depth.

Real-time classification of Dynamic ASL has never been achieved before, but researches have gotten close.

Researcher Lionel Pigou in 2014, used a system of Microsoft Kinect, convolutional neural networks and GPU acceleration to identify 20 Italian gestures with a 91.7% accuracy. Sign Language and Recognition using Convolutional Networks shows the proficiency of using CNNs to identify sign language. Though this study was for Italian sign language, CNNs are applicable for ASL recognition. The advantages of using CNN is they are able to capture and learn features from videos and images, otherwise known as feature learning. They are able to analyze multiple frames instead of just one, as demonstrated in Sign Language and Recognition using Convolutional Networks. In addition, they are good for feature extractions. CNNs can be trained with data-sets, with the help of a classifier, in our case OpenCV, and can train it to recognize more specific features, like finger movements, instead of something very broad, like body gestures. However, if you don't have a high GPU they are slow to train and also CNNs require a lot of training data with the help of a classifier, OpenCV, as they have limited ability to process data in their raw form (Deep Learning 2015).

Since the project is in real-time, the classifiers need to know when the sign has ended and another has begun. The study A Real-time Gesture Recognition System for Isolated Swedish Sign Language Signs, led by Kalin Stefanov and Jonas Beskow overcame this problem. While sign language is being classified in real time, after/before every sign they put their arms down, the CNN recognizes this as a new sign beginning to avoid a very long continuous sign. A simple analogy is voice-text, when an individual speaks into a phone to translate the audio into text the final text isn't a mesh of words as the algorithm can differentiate singular words. By Stefanov putting his hands to the side, the classifier and CNN know to begin recognizing a different sign. This is very cumbersome and timely. Sign language is fast paced, and by arms being put to the side of the body after every sign, the speed of communication decreases. An End-to-end 3D Convolutional Neural Network for Action Detection and Segmentation in Videos presents a novel solution to find the end of a sign or action using temporal convolutional neural networks that use frame level proposal algorithm to suggest where actions may end. The downside of this is it is very complex and would require a lot of GPU power.

The development of a machine learning application to analyze sign language is an interesting problem as it requires many technological components as well as a "human-like" intelligence. Progress has been made in recognizing static letters of ASL using data from frames of a sign and recognizing a small number of one-armed gestures. To push the boundaries of sign language detection the project consists of a machine learning applications that are able to recognize dynamic ASL.

In the past, researchers have done similar tasks; however, their systems require an additional dedicated piece of technology that is not optimal for ease of use — for instance, a Microsoft Kinect, which is costly and not mainstream. These additional pieces of technology allow for detection of depth to help achieve accurate recognition and to improve detection of the arms, the body, and hand.

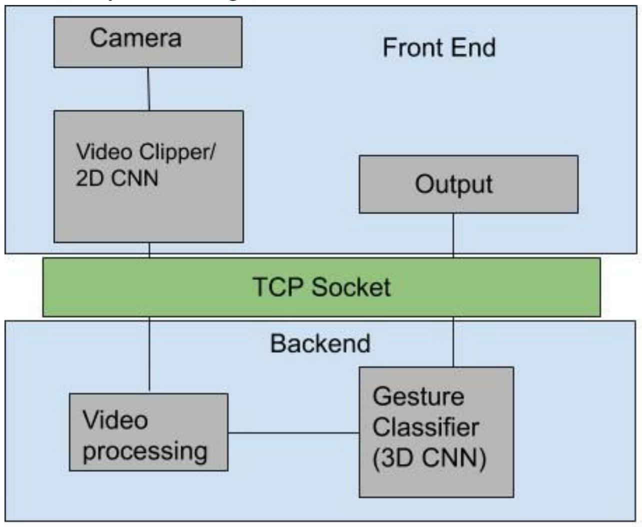
Though it's beneficial to use additional equipment for better preprocessing of training and testing data, it comes at the cost of losing functionality. The Microsoft Kinect allows for background subtraction and hand isolation which is difficult to accomplish otherwise, but would be arduous to implement for mainstream use. By using a conventional web camera this project needs to have a replacement for the benefits provided by additional technologies, which is accomplished with OpenCV.

II. EXPERIMENTAL METHODS

Any type of machine learning relies on large data sets for training. In order to train the convolutional neural network (CNN), videos are crowdsourced from ourselves and other individuals to provide an ample size data set for the neural network to learn from.

The application is composed of a three-layer system: a graphics user interface (GUI) layer, a 2D convolutional neural network layer (CNN), and a 3D CNN. The GUI layer streams real-time video to the computer to be saved as AVI or MP4 files. Individual signs can be isolated from the main video with the record/stop feature of GUI or a 2D CNN which detects when the signer is not moving to segment the signs. With an individual sign, preprocessing occurs to normalize the data set. Then each video is converted into a 5-dimensional array that has equal parameters for the 3D CNN to test off from a pre-learned data set. The dimensions of the array are video, channel, pixelX, pixelY, and pixelZ.

Chart 1. System Design

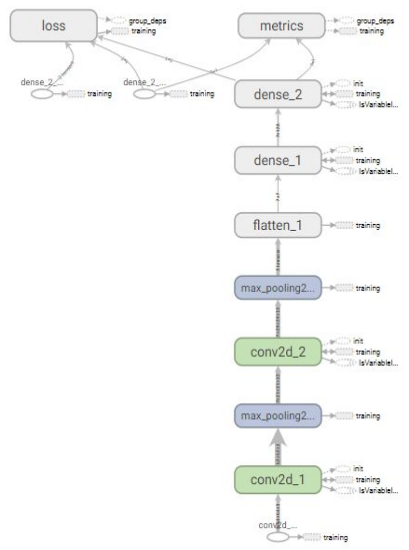


This is the design of the finalized system with the backend running in c# and the front running in javascript.

The front end GUI and the backend server communicate through TCP sockets. TCP sockets were a better choice than UDP in this case because of the higher reliability and packet flexibility. The system has a balance between frontend and backend performance. The front end has the optimal amount of processing while still keeping it runnable on mobile devices. This allows for the system to be viable in terms of business. More processing on the server side correlates to more money for hosting.

Outlined in Chart 1, the data flow for the system starts with the camera. For this project, the camera used was the 12-megapixel rear camera auto-HDR, on the iPhone 8. This camera has the potential to shoot 4k video in 60 FPS, however this is irrelevant as the quality and framerate of the video is downsized in the video preprocessing portion of the system, as seen in Chart 1.

Chart 2. 2D Convolutional Neural Network Design

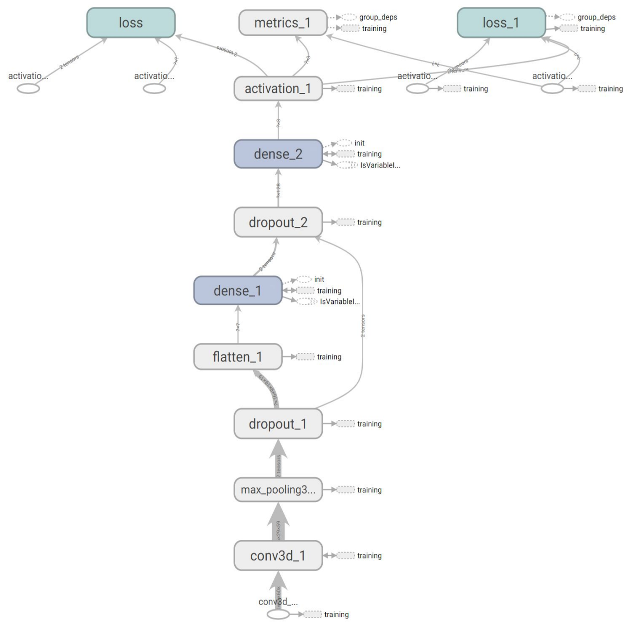


Depicts the layers in the 2D network

After, the camera transfers the series of frames to the system, a 2D CNN is used. This network uses the 2D convolutional network for video segmentation. Segmenting the video when a sign is being signed and when a sign is not being signed. This is important, as the 3D CNN needs very specific and exact data. If the 2D CNN wasn't utilized the 3D CNN would be unable to classify video, as the frames it received would be a mesh of multiple signs. The 3D CNN needs individual signs to classify.
This is how the 2D CNN works. The video frames, once classified are flattened and modified to output 2 values in the same manner as the 3D video classifier. The two values are the probability of the frame being of someone signing and the probability of it not.
The data used for this network is obtained by splitting the training videos for the 3D network into its individual frames, making up the positive set. The negative dataset was made by taking videos of individuals not doing sign language. This network is then able to determine if the user is doing sign language or not.

Then with the video clipped, preprocessing is underway. For the sake of efficiency, pre-segmented signs were manually used, instead of using the segmented video from the 2D CNN. This ensured no problems would occur and the data would stay valid throughout the entire process. Without this, it would be difficult to detect and find the reason if the code was not working properly.
With the segmented video, preprocessing occurs. This normalizes the dataset. For machine learning it is essential that all the data is constant in size and type. The video, regardless of the frame rate or quality, is converted into 30, 50 by 60 pixel frames for the 3D CNN to use. This is done by changing the img_size and img_depth. Additionally, the frames are converted to greyscale, so instead of color being stored in three different RGB values, it is only one. All this preprocessing is used to reduce the complexity of the data, to speed up the classification by using less graphical processing unit (GPU) power.

Chart 3. 3D Convolutional Neural Network Design



This chart has the layers that the data goes through after it is input into the system.

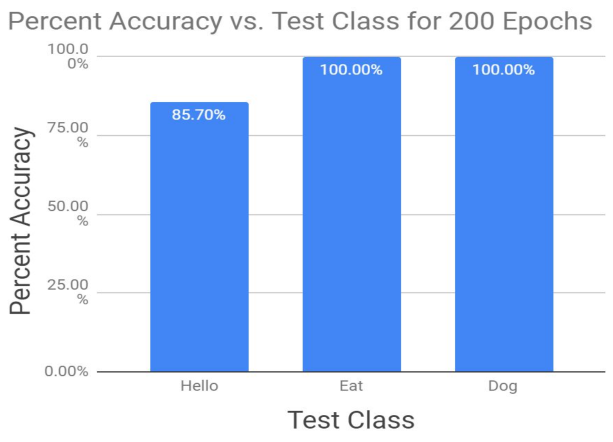
The chosen layer design starts with a 3D convolutional neural network to turn the video into a classified 2D data. This 2D data is then flattened and modified in order for it to get to an array with 3 values. Each of these values is a float with the value representing the percent probability that the classified video is in its respective class. The output is 3 long because the network is trained it off 3 classes. This network is trained off 42 videos total of the 3 classes.

This final result, the predicted sign from the video, is then sent to the front end using TCP sockets. Every piece of data from the camera to the output predicted sign goes through this system.

III. RESULTS

This project shows that this 3D CNN can predict signs with high levels of accuracy for a small number of different signs. Additionally, from the loss function, it is clear that the model is learning, as the number of epochs increases the loss decreases. This is promising information since it shows when the number of signs is increased, the accuracy will continue to be high.

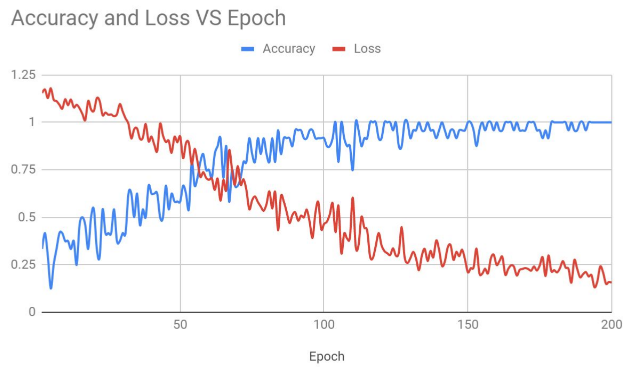
Chart 4. Accuracy of 3D network



This is the accuracy of the 3D classifier based on a sample class of 3 different videos and a sample size of 42 videos.

The accuracy obtained was very promising. As shown in Chart 4, he system achieved 100% accuracy in two of the classes and 85% in the other. As the test dataset increases, the accuracy will go down as edge cases are tested. The accuracy although not 100%, will be high. This larger dataset, would decrease the certainty (how confident the model is on a particular sign) but still would be accurate as the certainty for all signs would be decreased.

Chart 5. Accuracy and loss of 3D network



The models were trained off 200 epochs and achieved an accuracy of ~100% and a loss of ~0.17.

The model was trained with 200 epochs as shown in chart 5. It achieved good results with its accuracy and loss. The loss is calculated through a loss function which determines how well the 3D CNN models the dataset. A lower loss value correlates to a better model, while a higher loss value correlates to a more randomized model. Through testing it was found that 200 epochs at a batch size of 100 was optimal for this model. Adding more epochs would cause overfitting and too little would not be trained enough.

IV. DISCUSSION

Though our project has not achieved the initial goal outlined in the introduction, the project is still a success. This project shows dynamic ASL can be classified using a lightweight application; for this project a smartphone camera was used. Instead of using costly depth detecting cameras, such as the Microsoft Kinect, with the aid of preprocessing, lightweight applications can be just as accurate. Our model achieved a similar high levels of accuracy to Lionel Pigou work in “Sign language recognition using convolutional neural networks” where expensive hardware was used.

Machine learning relies on large sample sizes and testing sets. Since for this project, the videos were constructed by ourselves there was a lack of data to train and test off of. Though the results in Chart 4 show 100% accuracy, it is not possible to say that the accuracy of the model is 100% with some signs. To improve the model increasing the dataset is vital.

On a better note, since the data set which the system uses has a scalable folder, signs can be easily added, creating a ‘dictionary’ of signs. The front end of the system is programmed in JavaScript, enabling the program to be used cross-platform on any device that has browser support. This is the only requirement because all of the heavy lifting is offloaded onto a server for processing the data. The only action that the client does is clipping the input stream into the distinctive signs which is not a very intense application. This will lead to a final system that is able to translate sign language.

This project demonstrates that a 3D CNN for sign language classification is achievable. For the future, with a larger dataset of signs, more preprocessing methods will be used, this includes hand isolation and tracking, to maintain high accuracy. With only the essentials of sign language, the model will have less complex data to train and test off, thus improving its ability to learn features and increasing the accuracy. Additionally, an algorithm to understand higher level language could be implemented to increase the readability of the translated signs by including words like ‘a’, ‘the’ and other words that are not included when using sign language.

This project has important societal impacts for communication among individuals. A successful product could be easily made mainstream as a website or as an applications on a computer. This could take the place of companies such as SignALL, as the expensive hardware they use to provide sign language translation would not be needed. The information and the data gathered from this project can contribute to a more mainstream automatic sign language translator.

AUTHOR INFORMATION

Sean McHale

* Can be contacted at sean.mchale@er9.org

Kol Crooks

* Can be contacted at kol.crooks@er9.org

Author Contributions

The manuscript was written through contributions of all authors. All authors have given approval to the final version of the manuscript. Sean McHale and Kol Crooks have contributed equally.

ACKNOWLEDGMENT

Thank you to our High School for giving us this opportunity to make and present a research project. Along with our encouraging mentors, Paul Testa and Dr. Katherine Nuzzo, who provided their time and advice.

ABBREVIATIONS

ASL, American sign language, 2D, two dimensional 3D three dimensional, CNN, convolutional neural network. FPS, frames per second. GPU, graphical processing unit. GUI, graphics user interface. AVI, audio visual interface. TCP, transmission control protocol.

REFERENCES

Build software better, together. (n.d.). Retrieved from https://github.com/topics/3d-cnn

Garcia, B., & Viesca, S. A. (n.d.). Real-time American Sign Language Recognition with Convolutional Neural Networks [Scholarly project]. In Stanford. Retrieved September 30, 2018, from http://cs231n.stanford.edu/reports/2016/pdfs/214_Report.pdf

Heaton, J. 2019, Installing TensorFlow, Keras, & Python 3.7 in Windows. Retrieved March 1, 201 from https://www.youtube.com/watch?v=59duINoc8GM

Hou, R., Chen, C., & Shan, M. (2015, August). An End-to-end 3D Convolutional Neural Network for Action Detection and Segmentation in Videos [Scholarly project]. In JOURNAL OF CLASS FILES, VOL. 14, NO. 8. Retrieved September 30, 2018, from https://arxiv.org/pdf/1712.01111.pdf

Kadous, M. W. (1998, September). Auslan Sign Recognition Using Computers and Gloves [Scholarly project]. In Semantic Scholar. Retrieved September 30, 2018, from https://pdfs.semanticscholar.org/80d6/54eb9a924eb22d8cd6d72c45f2672cab7c75.pdf

Pigou, L., Dieleman, S., Kindermans, P., & Schruawen, B. (2015). Sign language recognition using convolutional neural networks [Scholarly project]. In Universiteit Gent. Retrieved from https://biblio.ugent.be/publication/5796137

Runhani. (2017, November). Intuitive understanding of 1D, 2D, and 3D Convolutions in Convolutional Neural Networks. Retrieved March 20, from https://stackoverflow.com/questions/42883547/intuitive-understanding-of-1d-2d-and-3d-convolutions-in-convolutional-neural-n

Shah, A. 3D CNN-Action Recognition Part-1(2016, June 19). Retrieved May 31, 2019, from https://www.youtube.com/watch?v=ecbeIRVqD7g

Shah, A. 3D CNN-Action Recognition Part-2(2016, June 19). Retrieved May 31, 2019, from https://www.youtube.com/watch?v=dYVmeJFA_lA

Stefanov, K., & J. (2016). A Real-time Gesture Recognition System for Isolated Swedish Sign Language Signs [Scholarly project]. Retrieved September 30, 2018, from http://www.ep.liu.se/ecp/141/004/ecp17141004.pdf

Yang, H. (2014). Sign Language Recognition with the Kinect Sensor Based on Conditional Random Fields. Sensors, 15(1), 135-147. doi:10.3390/s150100135