Improving American Sign Language Recognition with Synthetic Data

Jungi Kim
SYSTRAN Software, Inc.
jungi.kim@systrangroup.com

Patricia O'Neill-Brown
U.S. Government
po17b@icloud.com

Abstract

There is a need for real-time communication between the deaf and hearing without the aid of an interpreter. Developing a machine translation (MT) system between sign and spoken languages is a multimodal task since sign language is a visual language, which involves the automatic recognition and translation of video images. In this paper, we present the research we have been carrying out to build an automated sign language recognizer (ASLR), which is the core component of a machine translation (MT) system between American Sign Language (ASL) and English. Developing an ASLR is a challenging task due to the lack of sufficient quantities of annotated ASL-English parallel corpora for training, testing and developing an ASLR. This paper describes the research we have been conducting to explore a range of different techniques for automatically generating synthetic data from existing datasets to improve the accuracy of ASLR. This work involved experimentation with several algorithms with varying amounts of synthetic data and evaluations of their effectiveness. It was demonstrated that automatically creating valid synthetic training data through simple image manipulation of ASL video recordings improves the performance of the ASLR task.

1 Introduction

In everyday life, there are situations in which there is the need for deaf and hearing individuals to communicate with one another without the aid of an interpreter. To address this need, we are developing ASL-English MT that enables signers and non-signers to communicate with one another using mobile devices such as smartphones and tablets. The concept is that the signer of ASL signs into the device and the video images are captured, automatically recognized, translated and rendered into both speech and text for the speaker of English. Conversely, using this application, the speaker's speech is automatically recognized and an avatar signing the machine translation in ASL is displayed, which appears along with the English text. This paper outlines our work on the first critical aspect of the problem, which is the development of an automatic sign language recognizer (ASLR). Specifically, we address our research in the area of generating valid synthetic data, a requirement dictated by the lack of sufficient amounts of large-scale annotated data for ASL to English for testing, training and developing ASLR algorithms.

ASL is a visually perceived language based on a naturally evolved system of articulated hand gestures and their placement relative to the body, along with non-manual markers such as facial expressions, head movements, shoulder raises, mouth morphemes, and movements of the body (ASL: A brief description - Lifeprint.com). This language is structured like Japanese: it is a topic-comment language and does not have articles (Nakamura, 2008). See Speers (2002) for an excellent detailed linguistic description of ASL. The challenges involved with the recognition of sign language are akin to those of automatic speech

recognition (ASR) (Dreuw et al., 2007). As with speech, for sign language, allophonic variation must be taken into account, since each time a person makes a particular sign, they make tiny variations in how they position their arms and hands. In addition, hand and arm sizes and shapes vary between signers. An effective recognizer must be able to handle these variances as well as the differences in background colors and lighting. To account for these variations, similar to an acoustic range for each type of sound such as each specific vowel, we have developed the concept of a 'bounding box' within which each sign needs to be made for it to be considered that sign as judged by native signers and interpreters. Our plan is to automatically discover the bounding box for each sign through training on images of different signers signing each sign. However, since the datasets available for training algorithms to automatically recognize sign language do not contain a sufficient quantity of signer variation, this research focuses on the automatic creation of valid synthetic data to accurately capture those variations.

In this paper, we discuss our research into the area of automatically generating synthetic data for the ASLR component of an ASL-English MT system. Synthetic data generation has proven to be effective in solving problems such as image classification (Krizhevsky et al., 2012), ASR (Ko et al., 2017), and MT (Sennrich et al., 2016). The approach of using synthetic data to train and test machine learning algorithms is a newly emerging topic of interest and an area of active research in the field of Artificial Intelligence (AI). A popular approach in the attempt to solve this problem is to utilize the generative model in the Generative Adversarial Network (GAN) architecture (Antoniou et al., 2017; Gurumurthy et al., 2017; Bousmalis et al., 2017). However, we experimented with GANs and we found that they generated valid as well as non-valid signs and we were unable to constrain them to automatically generate only valid signs. Therefore, we looked for alternative approaches. This paper discusses our work on developing and testing different techniques for automatically generating valid synthetic data and determining whether synthetic data increases classification accuracy. In particular, we address two key research questions: a) given a very small amount of annotated data for training, how much synthesized data can we utilize for augmenting the training data without hurting classification performance, and b) would different synthetic data generation methods produce different outcomes for the model performance and if so, what techniques are most suitable and why.

2 Previous Work

A review of the literature reveals different approaches to the development of ASLR. Starner et al. (1998) implemented two Hidden Markov Model based real-time systems for ASLR where one system utilized a desk-mounted camera and the other utilized a hat-mounted camera. Lang et al. (2012) made use of Microsoft's Kinect for recognizing German signs. Chuan et al. (2014) used the palm sized Leap motion sensor for American finger spelling recognition. Dong et al. (2015) designed a color glove-based technique on the Kinect depth sensor for hand segmentation. Tharwat et al. (2015) developed the Arabic Sign Language recognition system where the scale invariant feature transform is used to perform the sign recognition using Neural Network, K-Nearest Neighbors, and Support Vector Machine. Wu et al. (2016) utilized an inertial measurement unit and surface electromyography devices for the recognition of 80 ASL signs. Dai et al. (2017) used gyroscope and accelerometer sensors running on a smartwatch for the recognition of 103 ASL signs. Ma et al. (2018) utilized WiFi packets to estimate hand and finger movements for ASL sign recognition.

There was the approach of developing a sign language recognition system through the training of very large data sets of video clips recorded by multiple signers using a large vocabulary (Koller et al., 2015). Koller's DeepHands model took an unsupervised approach to training a Convolutional Neural Network (CNN) model with 1 million unlabeled hand-shape images and successfully used it to classify Danish, New Zealand, and German signs (Koller et al., 2016). Since it modeled the hands, it could recognize all signs made using the hands and not just those that are finger-spelled, so we leveraged this work to develop a baseline prototype ASLR.

Anantha Rao et al. (2018) implemented Indian Sign Language recognition running in real-time on a mobile phone using hand image segmentation and a feedforward neural net-work. Huang et al. (2018) developed a CNN-based Hierarchical Attention Network with Latent Space in a sequence-

to-sequence fashion. In particular, the following studies are similar to our work on the aspect of augmenting training data to improve the performance of CNN models. Molchanov et al. (2015) experimented with the deformation of input data by augmenting reversed ordering and mirroring in the off-line and by augmenting rotating, scaling, shifting, and random dropout in the online for the hand gesture recognition using 3D CNN. Bheda and Radpour (2017) developed CNN-based ASL recognition with data augmentation (rotating and horizontal flipping). These projects applied data augmentation techniques developed for CNN model training; however, they did not evaluate the effectiveness of the techniques nor provide in-depth analyses of their methods. Tao et al. (2018) implemented an ASL alphabet recognition system with a CNN, equipped with a multi-view augmentation and inference scheme. This approach differs from our work in that they exploited a 3D motion capture device and the 3D modeling capability to virtually generate views from different angles, while we opted to utilize a generic input device and 2D techniques. Our approach is to develop the solution so that it is not reliant on special equipment and can run on any tablet, laptop or smartphone.

3 Approach

3.1 Overview of the Baseline System

For this work, we began with a baseline prototype[1] ASLR that we used for developing and testing the hypothesis of data augmentation based on DeepHands using the Kinect Sensor and a graphical user interface to capture the video recordings of people signing in ASL. The Kinect uses multiple cameras in order to capture motions in three dimensions, and utilizes the Kinect 2.0 SDK library, to outputs 25 body joints and their 3D coordinates (Microsoft, 2014). The demo system managed the recording of ASL sign videos of registered users with true labels which were annotated by signers. The demo system was developed by training the system to recognize 50 different signs using these datasets. The recognizer was trained to recognize a single-sign video clip as one of the 50 signs it was trained on. The baseline ASL recognizer consists of the following components: Kinect 2.0 as a video input device, Kinect SDK for feature extraction, and Kmeans clustering for classification.

For this work, we modified the original baseline system so as to remove the dependency on the Kinect input device to enable the system to train and classify on any 2D video feed or recordings. This enabled us to carry out experiments on a set of ASL video recordings of native ASL signers along with annotations that are made publicly available (Neidle et al., 2012). Being able to do away with specialized recording hardware also opened up the possibility of easily adding or creating more annotated data for training and evaluating the system. As shown in Figure 1, in place of the Kinect device, we utilized OpenPose (Cao et al., 2018) as the input video analysis module of the baseline system. OpenPose[2] is open source software that implements the state-of-the-art multi-person keypoint detection approaches for body, face, hands, and feet. In our preliminary experiments, it was verified that system performance was not degraded when using the features prepared from OpenPose output instead of Kinect output.[3]

3.2 Feature Extraction

OpenPose provides pretrained pose, face, and hand detection models trained on publicly available datasets. We used the pretrained 25-point body pose and 20-point hands detection models. OpenPose models produce the body and hands keypoints for each successfully analyzed frame of an input video clip. Two types of features are extracted for all frames: a) hands 2D coordinate feature and b) DeepHand hand-shape feature. A simple python script was written to automate the feature extraction process from a video clip and its OpenPose output with a tensor flow version of the DeepHand model. This code is available at https://github.com/ Dragonfly-ASL to make our work easily reproducible.

Hands 2Dtracking feature: In the OpenPose analysis output for each video frame, two coordinates (index 4 and 7) out of 25 body keypoints correspond to the right and the left wrists. The two coordinates were normalized with regard to the coordinate of the neck as the origin and the distance between neck and nose as the unit vector. The

[1]The baseline prototype and demo systems were developed by the Massachusetts Institute of Technology Lincoln Laboratories (MITLL) under a government contract.

[2]https://github.com/CMU-Perceptual-Computing-Lab/openpose

[3]Top 1 accuracies of the recognizer with Kinect and OpenPose were 61.8% and 61.5%, respectively.

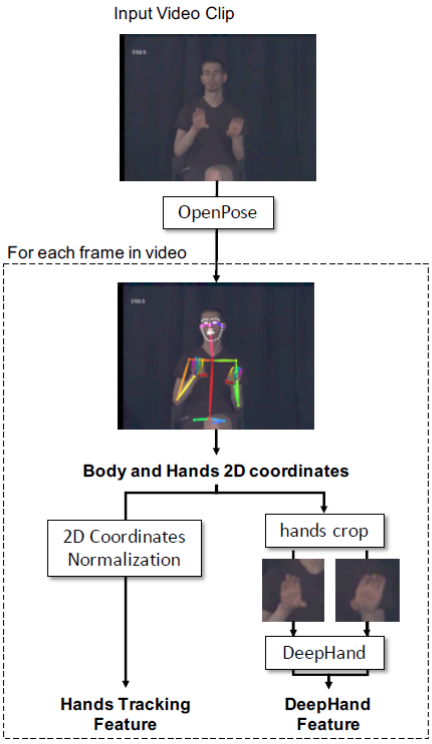


Figure 1: Feature extraction using OpenPose and DeepHand (Example shown from (dataset: ASLLRP-ASLLVD, signer: Brady, sign: CHEM-ISTRY+)[4])

normalized left and right hand coordinates of all frames combined consist of the half of the hands 2D tracking feature. The other half of the hands 2D tracking feature is calculated as the derivative of the normalized 2D coordinates across the time dimension with window size 5. The total dimension of the hands 2D tracking feature is then NumFrames 8, where each hand in a frame is described with a normalized 2D point and its derivative.

DeepHand hand-shape feature: To create handshape features, our baseline system utilizes DeepHand models (Koller et al., 2016), a CNN-based sign language recognizer trained on 1 million hand images. The model was trained to classify input images into 60 fine-grained hand-shape classes. The model performed with 62.8% accuracy on a manually labelled dataset with 3361 images that cover 45 hand-shape classes. The DeepHand takes its architecture after GoogLeNet (Szegedy et al., 2015). The model contains 22 layers, mixed with convolutional, pooling, fully-connected layers. The baseline system utilizes the activation output of one of the internal fully-connected layers, as the compact and abstract representation of the input image. The activation is of 1,024 dimensions for each hand, resulting in NumFrames $\times$ 2048 in total.

3.3 Classification with K-means Clustering

The baseline ASL classifier was developed using K-means clustering. There were two types of feature representations used for the training data: hands 2D tracking and DeepHand handshape. The K-means clustering algorithm was applied to each of the feature types for all of the video frames in the training data. K-means requires a distance measure between the clustered elements. A simple Euclidean distance was employed: $\mathrm{dist}(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$ where p and q are feature representations and n is the feature dimension. Once the clustering was finished, all clusters were calculated for the label probabilities using the distance between all elements in the training data with their annotated labels.

To classify an input video clip, two label probabilities were calculated for each feature type using the K-means clusters. For each feature type, a) the cluster memberships of the input video frames were determined using the same Euclidean distance, b) these distances were multiplied by the cluster's label probabilities, and c) the label probabilities were accumulated and averaged. The final label probabilities for the input video clip were averaged from the label probabilities for each frame. The two label probabilities were then combined as:

$$P_{\text{combined}} = e^{(\ln(P_{\text{HandTracking}})+\ln(P_{\text{HandShape}}))}$$

P_{combined} is in effect a product of $P_{\text{HandTracking}}$ and $P_{\text{HandShape}}$ but the calculation avoids the risk of underflow. Intuitively speaking, the probabilities for both the 2D hand tracking feature as well as the hand shape feature should be high for the combined probability to be high. Otherwise, if either one of the probabilities is low, then the combined probability stays low.

3.4 Data Augmentation

There is only a very small number of video clips each with a single ASL sign and its sign manually assigned. Therefore, our goal was to augment this data with a large amount of new videos synthesized from the original dataset, so we developed a tool that applies a set of image manipulation operations to all of the frames in a provided video clip. We categorized these 2D image manipulation operations according to their influence in the synthesis process as follows:

- Recording environment anticipation
- Sign and Signer variance anticipation

Depending on the actual usage of the proposed system, we took into account that the recording hardware and environment may vary. For example, ASL signers may use our software on different hardware devices such as their PCs, smartphones, or tablets with different cameras with varying field of view and resolutions, and different recording conditions such as varying levels of lighting and camera zoom and angle settings. These kinds of variances in recording environments may be addressed by training the recognition model with additional data generated with such effects applied to the existing training data. We anticipate that the following image manipulation operations account for such occasions: noise addition or removal, image enhancement, brightness changes, vertical and horizontal skews, etc.

The role of other image manipulation categories is to account for the different ways ASL signs are made by each ASLsigner and the different appearances between the signers in the training data and at the test or use time. There are many image manipulation operations that can potentially address such discrepancy in the training and test conditions. We limited the scope of this work to testing the usefulness of data synthesis within ASL sign recognition and focused on the following two image manipulation operations: Rotate and Zoom.

Each manipulation comes with its own set of parameter ranges and the generation space becomes quite big. For the scope of this experiment, we focused on these two broad sets of synthetic types: a) Random manipulations and parameters selection, and b) Controlled parameter selection (Rotate, Zoom).

Our training and test data were recorded in the identical environment and with identical camera settings. Therefore, we will not be able to verify the effectiveness of the corresponding manipulation operations within the scope of current work. However, we can still demonstrate that training an ASL recognizer with additional synthetically generated data, many times larger in quantity than that of the original data, can still yield a valid model. For this purpose, we have a type of synthetic data whose image manipulation operations and its parameters are chosen and configured with randomness (Figure 2).

Manipulation Operation	Parameter Range
shift-x, shift-y	-10 ~ +10, -3 ~ +3
skew-x, skew-y	-10 ~ +10, -10 ~ +10
rotate	-10 ~ +10
zoom	90% ~ 110%
contrast	0 ~ +3
brightness, saturation, hue	85 ~ 115
DE speckle, enhance	50% probability
normalize, quantize	10% probability
gamma, reduce Noise	0 ~ +2
swirl	-15 ~ +15

Figure 2: Manipulation operations and their parameters were randomly selected for each input video.

For the controlled parameter selections of Rotate and Zoom, three sets of varying ranges and increment steps are selected as below:

- Rotate (degrees angle)
  - Rotate1: $-15^\circ \sim 15^\circ$ (step size 3)
  - Rotate2: $-30^\circ \sim 30^\circ$ (step size 6)
  - Rotate3: $-45^\circ \sim 45^\circ$ (step size 9)
- Zoom (%)
  - Zoom1: $95\% \sim 105\%$ (step size 1)
  - Zoom2: $90\% \sim 110\%$ (step size 2)
  - Zoom3: $85\% \sim 115\%$ (step size 3)

With these varying ranges but with the same amount of generated synthetic data, we did work to gather preliminary evidence to support hypothesis that certain types of image manipulations have greater impact and benefit by helping the synthetic data generation process better address the lack of variability in the limited amount of training data. Through experimentation, we learned that there are certain parameters that are more important than others to ensure that the signs generated are valid.

The complete python script that can produce a synthesized video given an input video clip with a set of various image manipulation options is available at https://github.com/Dragonfly-ASL to make our work easily reproducible

4 Experiments

4.1 Experimental Settings

We trained our ASL recognition models using a publicly available annotated dataset American

Sign Language Lexicon Video Dataset (ASLLVD) (Neidle et al., 2012). ASLLVD consists of almost 10,000 ASL signs signed by 6 native ASL signers. The dataset also comes with human-annotated linguistic information such as gloss labels and handshape labels. Using the entire dataset, we selected videos with glosses that belong to our hand-picked 50 ASL signs, each signed by 6 different signers, making the total dataset size 300. ASLLVD provides videos shot from different angles (front view, side view, close-up), but we only used videos with the front view (Figure 3).

To carry out experiments with varying amount of synthetic data, we trained our ASL recognition model with varying amount (0 or 0%, 1 or 100%, 3 or 300%, 5 or 500%, and 10, 1000%) of synthetic data generated with Random operations and parameter selection method. Another set of experiments was performed to demonstrate the effectiveness of two image manipulation operations (Rotate and Zoom), each with three sets of parameter ranges as described in Section 3.4. For each of the parameter selection strategies (Random, Rotate1 . . . 3, and Zoom1...3), each video clip in the ASLLVD dataset was augmented with up to ten additional synthesized data variants (SYN1...10).

In the preliminary experimentations with the baseline system, it was noted that the hyperparameter K, the number of clusters in the Kmeans algorithm, has a significant impact on the classification performance. For the baseline experiments with 300 ˜ 500 annotated data, K was tested with 1,000 ˜ 3,000 with increments of 1,000. To account for the increased size in data (300 ˜ 3,300), we experimented with K with values 1,000, 2,000, 3,000, and 5,000.

To compare the performance of models, we employ accuracy as our main evaluation measure. Each accuracy is averaged over the scores of 6 signers, each tested with 50 signs evaluated in a cross-validation fashion. For example, we pick one signer at a time whose videos are set as a test input, and evaluate against a model trained with videos of the rest of the signers. Synthetically generated data of the test signer are not included in either the test or the training dataset. To eliminate the impact of random initialization in K-means clustering, each signer's score is averaged over three runs with different K-means initial cluster randomization. This is repeated six times for all signers and the performance is then averaged. Therefore, one accuracy score is an average of 18 independent runs.

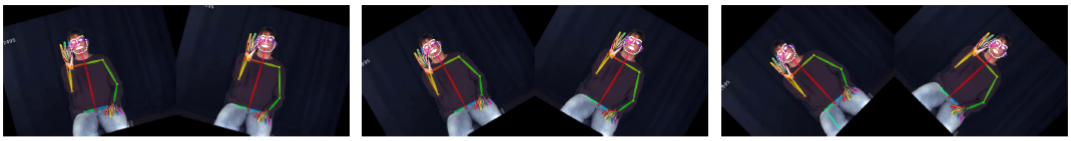
4.2 Experimental Results and Analysis

We present the performance evaluation results of all the experiments carried out in this work in Figure 4.

Scores with statistically significant improvements over the Baseline system, measured with the Wilcoxon signed-rank test (N = 50), are marked with † ($p < 0.15$) and ‡ ($p < 0.05$). Model configuration with 1,000% synthetic data with the K-means cluster size 3,000 performed the best among all configurations we tried for the current work, and it is shown to have improved most statistically significantly ($p = 0.006$). To confirm that the performance improvement did not occur by chance from having a good randomly initialized K-means cluster, we carried out additional experiments with the same configuration (K=3,000, SYN10, Rotate1) but with different random seeds two more times. The outcomes of the additional experiments show similar improvements (71.2% and 69.0%).

Figure 5 shows that, with regard to the varying amount of synthetic data in the train (Random-SYN1, 3, 5, 10, equivalent to 0, 100, 300, 500, and 1,000% synthetic data), the performance improvement is not in a linear relationship to the increasing amount of synthetic data in the train set. Rather, the performance first sharply decreases until 300%, then bounces back at 500% and finally outperforms the baseline at 1,000%. Though confirmed for all sizes of K-means cluster, this behavior is rather counter-intuitive. Due to the limited computing resource capacity (256G of RAM) and the way the baseline system was implemented, we could not utilize more than 1,000% synthetic data.

For runs with 1,000% of synthetic data (SYN10), many configurations of Random, Rotate and Zoom present statistically significant improvements over the Baseline. We also observe that certain configurations of Rotate and Zoom also perform better than Random, though none of the Rotate and Zoom configurations outperform Random with statistical significance. Some Rotate runs are worse than Random or even Baseline, indicating that parameter range for the data manipulation operations should be carefully chosen to ensure that the synthesized data still present valid signs. Another observation is that Random configurations performed reasonably well, and it would make a good go-to strategy in general.



(a) Rotate1: −15° and 15° (b) Rotate2: −30° and 30° (c) Rotate3: −45° and 45°

(d) Zoom1: 95% and 105% (e) Zoom2: 90% and 110% (f) Zoom3: 85% and 115%

Figure 3: Sample images with minimum and maximum values of Rotate1... 3 and Zoom1... 3 annotated with body and hand keypoints (ASLLVD, Signer: Lana, Sign: FOUR, Frame 67)

		Baseline	Random			SYN10						
			SYN1	SYN3	SYN5	Random	Rotate1	Rotate2	Rotate3	Zoom1	Zoom2	Zoom3
	Dataset size	300	600	1,200	1,800	3,300	3,300	3,300	3,300	3,300	3,300	3,300
	Synthetic %	0	100	300	500	1,000	1,000	1,000	1,000	1,000	1,000	1,000
Cluster Size (K)	1,000	62.8	62.0	58.7	61.3	63.0	66.5	65.5	60.7	63.1	65.3	64.5
	2,000	67.0	63.2	62.3	63.5	66.9	69.5	65.7	62.3	66.2	68.2†	67.2†
	3,000	66.7	64.9	64.1	64.6	67.7†	**71.1**‡	66.6	61.9	66.2†	68.7†	67.7†
	5,000	67.1	65.4	64.5	65.1	68.8†	70.5†	68.6†	64.4	67.9†	69.0	68.0†

Figure 4: Top 1 Classification accuracies (%) of the baseline ASL Sign recognizer and recognizers trained with additional synthetic data. Statistically significant improvements over the baseline system with the same cluster size (within the same row) are marked with † ($p < 0.15$) and ‡ ($p < 0.05$). We used the Wilcoxon signed-rank test with N = 50. For experiments using the 1,000% of synthetic dataset (SY N10), none of the Rotate and Zoom runs out-performed Random with statistical significance.

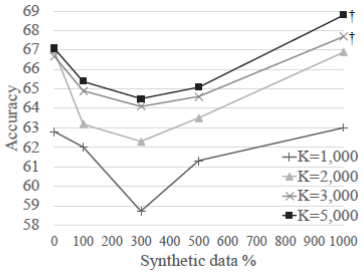


Figure 5: Change in recognition performance with different percentages of synthetic data generated with Random options. † = statistically significant improvement over Baseline (0%).

Though more sensitive to parameter range choices, between Rotate and Zoom, Rotate seems to be more effective in synthetic data generation. We conjecture that this is due to the fact that the 2D coordinates normalization of the hands tracking feature accounts in part for the effect of Zoom. We also speculate that Zoom together with image resizing to make thinner or wider signers should help account for the variances among differently sized and shaped bodies of signers.

We are currently working to further investigate the effectiveness of utilizing greater amounts of synthetic data and combinations of synthetic data generation techniques to identify the most optimal approaches.

In Figure 6, we see the per-sign rank comparison of Baseline and Rotate1 at their best configurations.

Signs with the most positive rank improvement are FOUR, EARTH, and ANY. Signs DEPRESS, CHAT++, and ANSWER were most negatively affected. Figure 7 shows the per-user top 5-ranked signs with their probabilities for input sign FOUR from Baseline (K=5000) and Rotate1 (K=3000). The most-frequently misclassified signs in top 5 rank for input sign 5(a) FOUR from Baseline were 5(b) BEAUTIFUL, 5(c) BLUE, and 5(d) FRIDAY+. Though not shown due to space constraints, the most-frequently misclassified signs in top 5 rank for input sign 6(a) DEPRESS from Baseline (K=5000) and Rotate1 (K=3000) were 6(b) CONFLICT-INTERSECTION, 6(c) DRESS-CLOTHES, and 6(d)EXCITED+. As these figures show, these signs look very similar to each other.

In Figure 8, we observe that the types of motion used by the signers are distinctively different

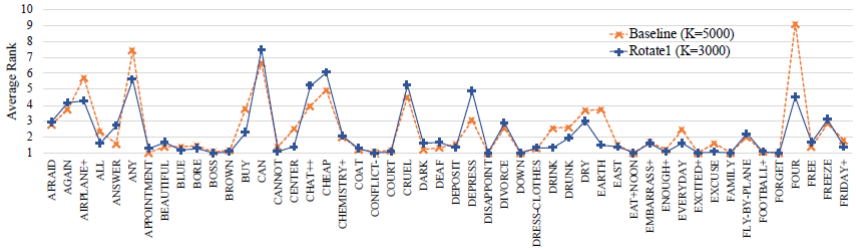
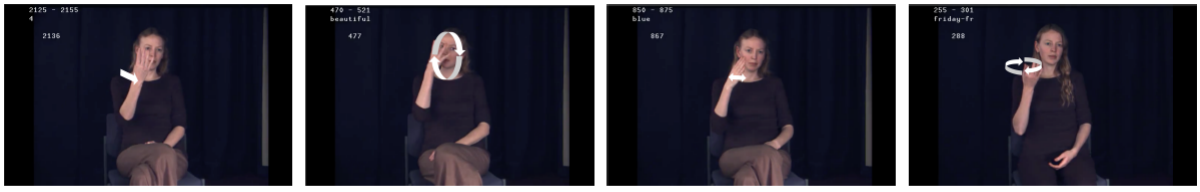


Figure 6: Average ranks of 50 signs for Baseline and Rotate1 (marked with $\times$ and marked $+$, respectively). Lower is better, 24 signs improved in rank (average 0.81), 19 signs degraded (average 0.54), and the ranks for 7 signs did not change.

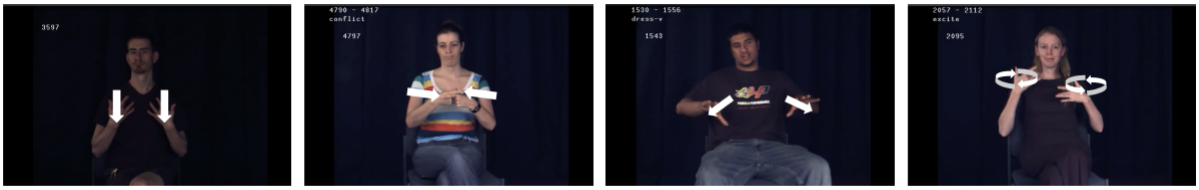
Baseline						
Rank	Brady	Dana	Lana	Liz	Naomi	Tyler
1	FOOTBALL+ 0.590	BEAUTIFUL 0.488	**FOUR** 0.982	BEAUTIFUL 0.887	FRIDAY+ 0.922	**FOUR** 0.549
2	BEAUTIFUL 0.0866	BORE 0.159	DOWN 0.0162	DRINK 0.0468	**FOUR** 0.0178	FRIDAY+ 0.395
3	EAT+NOON 0.0865	BLUE 0.152	BLUE 0.00055	FRIDAY+ 0.0357	BLUE 0.0168	BLUE 0.0482
4	BROWN 0.0724	FRIDAY+ 0.0522	ANY 0.000351	BLUE 0.0146	BEAUTIFUL 0.0117	BORE 0.00167
5	EVERYDAY 0.0527	BOSS 0.0405	FRIDAY+ 0.000287	**FOUR** 0.0117	BROWN 0.00873	DRESS-CLOTHES 0.00103
			...			
6		**FOUR** 0.0336				
			...			
38	**FOUR** 0.000125					
Rotate1						
Rank	Brady	Dana	Lana	Liz	Naomi	Tyler
1	BEAUTIFUL 0.948	**FOUR** 0.886	**FOUR** 0.864	BEAUTIFUL 0.472	FRIDAY+ 0.838	**FOUR** 0.890
2	FOOTBALL+ 0.0301	FRIDAY+ 0.0286	FRIDAY+ 0.131	FRIDAY+ 0.260	BROWN 0.0876	FRIDAY+ 0.0947
3	BORE 0.00654	DOWN 0.0286	DOWN 0.000766	BROWN 0.0989	BEAUTIFUL 0.0357	DRESS-CLOTHES 0.00584
4	BROWN 0.00267	BLUE 0.0269	DRY 0.000759	DRINK 0.0825	**FOUR** 0.0154	DEAF 0.00236
5	DRINK 0.00265	DRUNK 0.0133	EVERYDAY 0.000722	DISAPPOINT 0.0382	BLUE 0.0105	DISAPPOINT 0.00226
			...			
7				**FOUR** 0.00866		
8	**FOUR** 0.00170					

Figure 7: Per-user top 5-ranked signs with their probabilities for test input sign FOUR from Baseline (K=5000) and Rotate1 (K=3000) (Trial 1 result only). For each of the 6 signers, correct rank changed 38→8, 6→1, 1→1, 5→7, 2→4, 1→1.



(a) FOUR **(b)** BEAUTIFUL **(c)** BLUE **(d)** FRIDAY+

Figure 8: Most-frequently misclassified signs in top 5 rank for input sign (a) FOUR from Baseline (K=5000) were (b) BEAUTIFUL, (c) BLUE, and (d) FRIDAY+. (ASLLVD, Signer: Liz)



(a) DEPRESS (b) CONFLICT-INTERSECTION (c) DRESS-CLOTHES (d) EXCITED+

Figure 9: Most-frequently misclassified signs in top 5 rank for input sign (a) DEPRESS from Baseline (K=5000) and Rotate1 (K=3000) were (b) CONFLICT-INTERSECTION, (c) DRESS-CLOTHES, and (d) EXCITED+. (ASLLVD, Signers: Brady, Naomi, Tyler, Liz)

from those of FOUR. However, three out of four signs in Figure 9 move two hands in straight lines parallel to the body. We conjecture that this difference in the variations of hand motion affected the usefulness of adding synthetic data created by rotating videos in the plane parallel to the signer. In other words, for the case of FOUR, added synthetic data helped distinguish similarly looking signs because of the different hand motions for these signs, but in the case of DEPRESS, synthetic data created from signs such as CONFLICT-INTERSECTION and DRESS-CLOTHES did not help because the corresponding hand motions, with rotation, did not help and in some cases hurt differentiating those signs from DEPRESS.

5 Conclusions and Ongoing Work

In this work, we explored different strategies for generating synthetic data with the goal of improving ASLR performance, and we experimented with several techniques for the automatic generation of synthetic data in varying amounts. We demonstrated that creating synthetic training data through the simple image manipulation of each frame in ASL video clips helped improve ASLR performance. We anticipate more benefits from utilizing synthetic data for improving the performance of ASL recognizers.

In addition, we are working to extend our automatic generation of synthetic data strategies to the challenge of moving from the lexical level to machine translating videos of ASL sentences and paragraphs into English. In the course of our experimentation and analyses, we discovered a number of issues requiring further investigation. Next, we will experiment with synthetic data of more than 1,000% to the original data to see at what percentage the performance gains begin to diminish. We will also create a better method for generating valid synthetic data. We plan to do this by defining boundaries of spatial regions that include hand and body motions that constitute a valid sign and developing a synthetic data generation technique from this. Lastly, we will explore adding noise and background variations to the synthetic data generated and verify that these techniques help make ASL systems robust against noisy and poorly lit recording environments.

References

Anantha Rao, G., P. V. V. Kishore, A. S. C. S. Sastry, D. Anil Kumar, and E. Kiran Kumar 2018. Selfie Continuous Sign Language Recognition using Neural Network Classifier. In Proceedings of 2nd International Conference on Micro-Electronics, Electromagnetics and Telecommunications, pages 31–40, Singapore.

Antoniou, Antreas, Amos Storkey, and Harrison Edwards. 2017. Data Augmentation Generative Adversarial Networks. arXiv e-prints, page arXiv: 1711.04340, Nov.

ASL: A brief description - ASL American Sign Language. www.lifeprint.com/asl101/pages-layout/asl1.htm

Bheda, V. and D. Radpour. 2017. Using Deep Convolutional Networks for Gesture Recognition in American Sign Language. arXiv e-prints, October.

Bousmalis, Konstantinos, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. 2017. Unsupervised Pixel-level Domain Adaptation with Generative Adversarial Networks. In Proceedings of the 2017 Conference on Computer Vision and Pattern Recognition.

Cao, Zhe, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2018. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. In arXiv preprint arXiv: 1812.08008.

Chuan, C., E. Regina, and C. Guardino. 2014. American Sign Language Recognition Using Leap Mo-

tion Sensor. In 2014 13th International Conference on Ma- chine Learning and Applications, pages 541–544, Dec.

Dai, Qian, Jiahui Hou, Panlong Yang, Xiangyang Li, Fei Wang, and Xumiao Zhang. 2017. Demo: The Sound of Silence: End-to-end Sign Language Recognition using Smartwatch. In Proceedings of the 23rd Annual International Conference on Mobile Computing and Network-ing, MobiCom '17, pages 462– 464, New York, NY, USA. ACM.

Dong, C., M. C. Leu, and Z. Yin 2015. American Sign Language alphabet recognition using Microsoft Kinect. In 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 44–52, June.

Dreuw, Philippe, David Rybach, Thomas Deselaers, Morteza Zahedi, and Hermann Ney. 2007. Speech Recognition Techniques for a Sign Lan-guage Recognition System. In INTERSPEECH.

Gurumurthy, Swaminathan, Ravi Kiran Sarvadevabhatla, and R. Venkatesh Babu. 2017. DeLiGan: Generative Adversarial Networks for Diverse and Limited Data. In Proceedings of the 2017 Conference on Com-puter Vision and Pattern Recognition.

Huang, Jie, Wengang Zhou, Qilin Zhang, Houqiang Li, and Weiping Li. 2018. Video-based Sign Language Recognition without Temporal Segmentation. In AAAI.

Ko, T., V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur. 2018. A STUDY ON DATA AUGMENTATION OF REVERBERANT SPEECH FOR ROBUST SPEECH RECOGNITION. In 2017 IEEE International Conference on Acouctics, Speech and Signal Processing (ICASSP), pages 5220–5224, March.

Koller, Oscar, Jens Forster, and Hermann Ney. 2015. Continuous Sign Language Recognition: Towards Large Vocabulary Statistical Recognition Systems Handling Multiple Signers. Computer Vision and Image Understanding, 141:108 – 125

Koller, O., H. Ney, and R. Bowden. 2016. Deep Hand: How to Train a CNN on 1 Million Hand Images When Your Data Is Continuous and Weakly labelled. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 3793–3802, June.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of the 25th International Confer-ence on Neural Information Processing Systems - Volume 1, NIPS'12, pages 1097–1105, USA. Cur-ran Associates Inc.

Lang, Simon, Marco Block-Berlitz, and Raul Rojas. 2012. Sign Language Recognition Using Kinect. pages 394–402, 04.

Ma, Yongsen, Gang Zhou, Shuangquan Wang, Hongyang Zhao, and Woosub Jung. 2018. SignFi: Sign Language Recognition Using WiFi. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol., 2(1):23:1–23:21, March.

ASL: A brief description - ASL American Sign Lan-guage www.lifeprint.com/asl101/pages-layout/asl1.htm

Molchanov, P., S. Gupta, K. Kim, and J. Kautz. 2015. Hand Gesture Recognition with 3D Convolutional Neural Networks. In 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 1–7, June.

Nakamura, Karen. 2008. About American Sign Language. In: Deaf Resource Library. http://www.deaflibrary.org/asl.html

Neidle, Carol, Ashwin Thangali, and Stan Sclaroff. 2012. Challenges in Development of the American Sign Language Lexicon Video Dataset (ASLLVD) Cor-pus. In 5th Workshop on the Representation and Pro-cessing of Sign Languages: Interactions between Corpus and Lexicon, LREC 2012.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. 2016. Improving Neural Machine Translation Models with Monolingual Data. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 86–96, Berlin, Ger-many, August. Association for Computational Linguistics.

Speers, D'Armond Lee. 2002. Representation of American Sign Language for Machine Translation. Ph.D. thesis, Georgetown University, Washington, DC, USA. AAI3053310.

Starner, T., J. Weaver, and A. Pentland. 1998. Real-Time American Sign Language Recognition Using Desk and Wearable Computer Based Video. Transactions on Pattern Analysis and Ma-chine Intelligence, 20(12):1371–1375, Dec.

Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Du-mitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In Computer Vision and Pattern Recognition (CVPR).

Tao, Wenjin, Ming C. Leu, and Zhaozheng Yin. 2018. American Sign Language alphabet recognition using Convolutional Neural Networks with mul-tiview augmentation and inference fusion. Engineering Applications of Artificial Intelligence, 76:202 – 213.

Tharwat, Alaa, Tarek Gaber, Aboul Ella Hassanien, M. K. Shahin, and Basma Refaat. 2015. Sift-Based Arabic Sign Language Recognition System. In Abraham, Ajith, Pavel Kromer, and Vaclav Snasel, editors, Afro-European Conference for In-dustrial Advancement, pages 359–370, Cham. Springer International Publishing.

Wu, J., L. Sun, and R. Jafari. 2016. A Wearable System for Recognizing American Sign Language in Real-Time Using IMU and Surface EMG Sensors. IEEE Journal of Biomedical and Health Informatics, 20(5):1281– 1290, Sep