Sign Language Recognition Analysis using Multimodal Data

Abstract

Voice-controlled personal and home assistants (such as the Amazon Echo and Apple Siri) are becoming increasingly popular for a variety of applications. However, the benefits of these technologies are not readily accessible to Deaf or Hard-ofHearing (DHH) users. The objective of this study is to develop and evaluate a sign recognition system using multiple modalities that can be used by DHH signers to interact with voice-controlled devices. With the advancement of depth sensors, skeletal data is used for applications like video analysis and activity recognition. Despite having similarity with the well-studied human activity recognition, the use of 3D skeleton data in sign language recognition is rare. This is because unlike activity recognition, sign language is mostly dependent on hand shape pattern. In this work, we investigate the feasibility of using skeletal and RGB video data for sign language recognition using a combination of different deep learning architectures. We validate our results on a large-scale American Sign Language (ASL) dataset of 12 users and 13107 samples across 51 signs. It is named as GMUASL51. 1 We collected the dataset over 6 months and it will be publicly released in the hope of spurring further machine learning research towards providing improved accessibility for digital assistants.

Introduction

According to The National Institute on Deafness, one in thousand infants is born deaf. An additional one to six per thousand are born with hearing loss at different levels BIBREF0. Sign language is commonly used by Deaf and Hard-of-Hearing (DHH) persons to communicate via hand gestures. An automatic sign language recognizer enables an ASL user to translate the sign language to written text or speech, allowing them to communicate with people who are not familiar with ASL. There is a tremendous rise in the popularity of personal digital assistants; available on user's personal and wearable devices (Google

Now, Amazon Alexa and Apple Siri, etc.) and also in the form of standalone devices (Amazon Echo and Google Home smart speakers). These devices are primarily controlled through voice, and hence, their functionality is not readily available to DHH users. An automatic sign recognizer can also enable the interaction between a DHH user and a digital assistant.

Most current systems have capability of ASL recognition with RGB video data BIBREF1, BIBREF2, BIBREF3. An ASL sign is performed by a combination of hand gestures, facial expressions and postures of the body. Sequential motion of specific body locations (such as hand-tip, neck and arm) provide informative cues about a sign. Using video data, it is difficult to extract different body locations and associated motion sequences from a series of RGB frames. Microsoft Kinect is a 3D camera sensor which can use the depth information of a person to capture 3D coordinates of his/her body location across a video. This sequence of 3D body location is referred by skeletal data BIBREF4. To the best of our knowledge, there is no publicly available skeletal dataset in literature for ASL recognition.

With skeletal data, an ASL sign can be seen as a sequence of 3D coordinates or a 3D time series BIBREF5. Recurrent neural networks (RNN) have shown strong performance for sequential modeling BIBREF6. In this work, we investigate the impact of RGB video data in recognition accuracy when combined with skeletal data. We also propose a combined RNN network with a simple spatial data augmentation technique. In summary, the contributions of this work are:

We propose an RNN architecture with a novel spatial data augmentation technique.

We propose an architecture which uses both RGB and skeletal data to improve recognition accuracy.

We introduce and publicly release a multi–modal dataset for ASL called GMU-ASL51.

Literature Review

Most sign language recognition systems use RGB video data as input. These approaches model sequential dependencies using Hidden Markov Models (HMM). Zafrullah et al. BIBREF7 used colored gloves (worn on hands) during data collection and developed an HMM based framework for ASL phrase verification. They also used hand crafted features from Kinect skeletal data and accelerometers worn on hand BIBREF8. Huang et al. BIBREF1 demonstrated the effectiveness of using Convolutional neural network (CNN) with RGB video data for sign language recognition. Three dimensional CNN have been used to extract spatio-temporal features from video BIBREF2. Similar architecture was implemented for Italian gestures BIBREF9. Sun et al. BIBREF3 hypothesized that not all RGB frames in a video are equally important and assigned a binary latent variable to each frame in training videos for indicating the importance of a frame within a latent support vector machine model. Zaki et al. BIBREF10 proposed two new features with existing hand crafted features and developed the system using HMM based approach. Some researchers have used appearance-based features and divided the approach into sub units of RGB and tracking data, with a HMM model for recognition BIBREF11.

Compared to RGB methods, skeletal data has received little attention in ASL recognition. However, in a closely similar human action recognition task, a significant amount of work has been done using body joint location related data. Shahroudy et al. BIBREF12 published the largest dataset for human activity recognition. They proposed an extension of long short term memory (LSTM) model which leverages group motion of several body joints to recognize human activity from skeletal data. A different adaptation of the LSTM model was proposed by Liu et al. BIBREF13 where spatial interaction among joints was considered in addition to the temporal dynamics. Veeriah et al. BIBREF14 proposed a LSTM network to capture the salient motion pattern of body joints. This method takes into account the derivative of motion states associated with different body joints. Some have treated the whole body as a hierarchical configuration of different body parts and proposed a hierarchical RNN to recognize human activities

BIBREF5. Several attention based models were proposed for human activity analysis BIBREF15, BIBREF16. Some prior works converted skeleton sequences of body joints or RGB videos into an image representation and then applied state-of-the-art image recognition models BIBREF17, BIBREF18. Motivated by the success of skeletal data in human activity recognition, we investigate its suitability for recognizing ASL signs.

Dataset

ASL recognition with skeletal data has received little attention, resulting in a scarcity of public datasets. There exists one dataset for ASL recognition with skeletal data BIBREF19. This dataset has 9800 samples from 6 subjects and more than 3300 sign classes. The number of samples per class was small for use in deep learning based models. Adding to this, the samples were collected in controlled settings with uncluttered background. In contrast, GMU-ASL51 has 13107 samples for 51 word level classes from 12 distinct subjects of different height, build and signing (using sign language) experience.

Figure FIGREF6 shows the T-SNE representation of a subset of samples from GMU-ASL51. It was performed on output vectors from a trained RNN model for each sign example in the subset. The used model, AI-LSTM, is described in section SECREF19.

Dataset ::: Collection Protocol

The data was collected with a Microsoft Kinect version 2.0 depth camera positioned in front of the signer. For each sign (a single class like Air Condition or AC) we collected 24 samples continuously; and the process was repeated for every sign (51 classes in total). Due to time and availability constraints, for some subjects we could not collect the samples for all the classes resulting in a total of 13107 samples. The distance between the subject and the sensor was varied in the range from 10 to 15 feet to simulate

practical scenarios. No constraints were imposed on performers' posture and lighting condition of the room.

To gather individual samples from the continuous data, segmentation marks were interleaved through a user interface. This was later used to segment individual samples. These samples were further segmented using motion calculation of the wrist joint co-ordinates from skeletal data. Figure FIGREF7 (a) illustrates the distribution of number of samples per gesture class in GMU-ASL51 dataset. Figure FIGREF7 (b) shows the distribution of duration of videos in our dataset.

Dataset ::: Data Modality

All of our experiments on ASL recognition were done with RGB video data and/or skeletal data. Skeletal data is a multivariate, multidimensional time series input where each body part acts as a variable and each of them have 3D coordinate data at each time step. The skeletal data provides motion trajectory of different body parts such as wrist, elbow and shoulder (total 25 such body parts) over whole video frames. This process is called skeletal tracking. Skeletal data provides high level motion of different body parts. These are useful for capturing discriminant features associated with different types of gestures. However, for better modeling of sign language, hand shape is crucial, as different signs may have similar motion but different hand shapes and orientation. Figure FIGREF10 presents one such example where the sign pair Alarm and Doorbell have exact same motion pattern according to skeletal data but have different hand shapes. We observe similar situation for sign pairs such as Kitchen/Room, Time/Movie, Quote/Camera, Lock/Stop and many more.

We hypothesize that hand shape is useful in situations where skeletal data has similar dynamic motion pattern for different sign classes. Due to this fact, we extract and use hand shape patterns from RGB video data.

Our Approach

Inspired by the success of deep learning approaches in computer vision BIBREF20, we applied different deep learning architectures to model sign languages from both input modes (RGB and skeletal). Unlike traditional image classification or object detection models where neural networks learn hierarchical spatial features from data, sign recognition requires capture of temporal body motion.

Our Approach ::: Recurrent Neural Networks (RNN)

RNN has shown success in modeling sequential pattern in dataBIBREF6. It can capture temporal dynamics in data by maintaining an internal state. However, the basic RNN has problems dealing with long term dependencies in data due to the vanishing gradient problem BIBREF21. Some solutions to the vanishing gradient problem involve careful initialization of network parameters or early stopping BIBREF22. But the most effective solution is to modify the RNN architecture in such a way that there exists a memory state (cell state) at every time step that can identify what to remember and what to forget. This architecture is referred to as long short term memory (LSTM) network BIBREF23. While the basic RNN is a direct transformation of the previous state and the current input, the LSTM maintains an internal memory and has mechanisms to update and use that memory. This is achieved by deploying four separate neural networks also called gates. Figure FIGREF12 depicts a cell of an LSTM network which shows input at the current time step ${x_t}$ and the previous state ${h_{t-1}}$ enter into the cell; and get concatenated. The forget gate processes it to remove unnecessary information, and outputs ${f_t}$ which gets multiplied with the previously stored memory ${C_{t-1}}$ and produces a refined memory for the current time.

Meanwhile, the input and update gate process the concatenated input and convert it into a candidate memory for the current time step by element–wise multiplication. The refined memory and proposed

candidate memory of the current step are added to produce the final memory for the current step. This addition could render the output to be out of scale. To avoid that, a squashing function (hyperbolic tan) is used, which scales the elements of the output vector into a fixed range. Finally ${o_t}$, the output from output gate gets multiplied with the squashing function and produces the current time step output. Figure FIGREF12 shows an LSTM cell. The forget, input, update and output gates are represented by four circles and symbolized as f_t, i_t, $\tilde{C_t}$ and o_t, respectively . Equation DISPLAY_FORM13 shows LSTM functions; where $\oplus $ and $\otimes $ represent element wise addition and multiplication respectively; $\times $ represents matrix multiplication, $concat$ process means a concatenation of its input.

Our Approach ::: 3D Convolutional Neural Network

Traditional convolutional neural network (CNN) is two dimensional in which each layer has a stack of 2D feature maps generated from previous layer or from inputs in case of first layer. A layer also has a certain numbers of filters which are rectangular patches of parameters. Each filter convolves over the stack of 2D feature maps at previous layer and produces feature maps (equal to the number of filters in the current layer) at current layer. The operation is given by Equation DISPLAY_FORM17 where $F_{i,j}^{l}$ denotes the value of feature map at l^{th} layer at location (i,j). $\odot $ represents dot product of filter W and associated feature map patch in previous layer.

Standard CNN fails to capture the temporal information associated with data, which is important in video or any type of sequential data representation. To solve this problem, 3D convolution was introduced in BIBREF2. The key difference is that kernels are 3D and sub sampling (pooling) layers work across three dimensions.

Equation DISPLAY_FORM18 shows 3D convolution function. In this case from each filter we get a 3D

feature map and $F_{i,j,k}$ denotes value at (i, j,k) location after convolution operation. The dot product is between two three-dimensional matrices (also called tensors).

Our Approach ::: Axis Independent LSTM

Given a sample skeletal data of $R^{T \times J \times 3}$, where T denotes time axis, J is the number of body joints and the last dimension is the 3D coordinates of each joint. We flatten every dimension except time and at each time step we can feed a vector of size $R^{3 \times J}$ as input. However, we have empirically verified that learning a sequential pattern for each coordinate axis independently and combining them later shows stronger classification performance. Based on this, we trained three different 2 layer LSTMs for data from x, y, and z coordinates separately; and concatenate their final embedding to produce softmax output. In this setting, each separate LSTM receives data as $R^{T \times J}$ and final embedding size is $R^{3\times S}$ where S is the state size of LSTM cell. Figure FIGREF15 (a) shows the architecture where as a sample arrives, just before entering into main network, data along separate axis is split and entered into three different LSTM networks. The model concatenates the final state from each of the separate LSTM networks; followed by feeding this into the softmax layer for classification. This approach is referred by Axis Independent Architecture (AI-LSTM). Implementation details such as values of T and J are provided in the `Experiments' section.

Our Approach ::: Spatial AI-LSTM

AI-LSTM, described in last section, works by modeling temporal dynamics of body joints' data over time. However, there can be spatial interactions with joints at a specific time step. It fails to capture any such interaction among joints in a given time. To incorporate spatial relationship among joints, we propose a simple novel data augmentation technique for skeletal data. We do this by origin transfer. For each frame in a gesture sample, we use each wrist joints as origin and transform all other joints' data by subtracting

that origin from them. In this way spatial information is added to the input. We refer this model with spatial data augmentation as Spatial AI-LSTM. This augmentation technique is depicted in Figure FIGREF21. A sample data of form $R^{T \times 6 \times 3}$ results in a representation of $R^{T \times 5 \times 3}$ after subtracting left wrist joint (origin transfer). After this augmentation process, each sample is a $R^{20 \times 16 \times 3}$ matrix. Hence, each separate LSTM networks in our Spatial AI-LSTM network receives an input of $R^{20 \times 16}$.

Our Approach ::: Combined Network

We hypothesize that, some signs that have mostly similar skeletal motion pattern could be distinguishable using hand shape information. We propose a combination of LSTM and 3D CNN networks. We call this Max CNN-LSTM network. Figure FIGREF15 (b) represents the the Max CNN-LSTM. The details of 3D CNN module is shown in Figure FIGREF14. This architecture has two parts: one for left hand patches and other for right hand patches. Each part has four 3D convolutional layers (second and fourth layers have following maximum pooling layers) followed by 2 fully connected layers. Final embeddings from these two parts are concatenated and by using a softmax layer, from which a classification score is produced. The other AI-LSTM network is fed with skeletal time series data. At the final time step, the LSTM state vector is taken and using a softmax layer another probability score is produced. The final classification score is created by taking element wise maximum of the output scores from the two networks. During back–propagation, both networks are trained on their own score. The combined network acts like a model ensemble and some sign classes which are confused by RNN network alone might have an improved recognition accuracy with this approach.

Experiments

Naturally each sign has different frame length after segmentation because each subject does a sign at

different speed. It is possible that the same subject may do the same sign at different speeds at different times which makes the recognition challenging. Further, neighboring frames contain redundant information; and all joints will not have equal amount of motion or pattern in case of skeletal data.

Experiments ::: Skeletal Data

Most of the signs do not involve all the 25 joints' information provided by Kinect sensor; specifically, joints involved with the two hands convey most information. Based on this, we consider only 6 joints (wrist, elbow, shoulder) from both as input to the LSTM network. Figure FIGREF22 shows an example where 7 frames were sampled from a sign video of class Air Condition and the bottom panel shows the skeletal configuration across those 7 frames. From each sign video we sampled some number of frames uniformly and took joints' data associated with those frames. We verified empirically that picking 20 frames uniformly works best for skeletal data. For samples with less than 20 samples we convert them to 20 frame signs by interleaving existing frames uniformly. Thus skeletal data for each sample is a vector in $R^{20 \times 6 \times 3}$.

Experiments ::: Video Data

Since ASL involves specific hand shape patterns, we crop both hand regions at each frame. Using 2D coordinates of hand joints on a video frame as center, we do a 100×100 crop to generate hand patches. To reduce motion blur, we calculate velocity of joints at each video frame using skeletal coordinates and then sample from frames which have less motion. We sampled 15 frames from each sign video resulting in a vector of $R^{15 \times 100 \times 100 \times 3}$ for each hand patch.

Experiments ::: Training Details

To deal with over-fitting, dropout was used for all networks except convolutional layers with probability of 0.5. In addition to dropout, L2 regularization was used for LSTM networks and for dense layers; $\beta $ was set to 0.008 which controls the impact of regularization on the network. State size and number of layers of LSTM networks were 50 and 2, respectively. Learning rate for Max CNN-LSTM and LSTM networks were set to 0.00001 and 0.00005, respectively. We used Adam Optimizer for training our networks BIBREF24. All networks were run for a certain number of epochs (200-300) with a batch size of 64. We developed all of our models with Tensorflow 1.10 (python). Average time taken to train an AI-LSTM and an Spatial AI-LSTM are 25 and 30 minutes on an Intel(R) Core(TM) i5-7600 (3.50GHz) processor respectively. We trained 3D CNN and Max 3D CNN models on GPU (Tesla K80) and each model took around 20 hours to train.

Experiments ::: Baseline Methods

We use support vector machines and random forest for baseline comparison. The baseline models utilize skeletal data in each axis for every joint in building the following features per sample: Mean, Area, Skew, Kurtosis, Motion Energy, Range and Variance over the frames BIBREF25. We have 6 upper body joints and 3 axes per joint and 7 features for each giving a total of 126 $(7 \times 6 \times 3)$ features per sample.

Experiments ::: Experimental Results

Table TABREF28 shows the comparative results among our proposed architectures and baselines. Overall, we use data from 12 subjects for our experiments which sum up to 13107 sign gesture samples in total. To evaluate model performance on a specific subject (test subject), we adopt cross subject evaluation criteria. Suppose, X is the test subject. We train our networks with all sign samples except those are from subject X. We use subject X's data as test split to evaluate the performance of the

networks. Table TABREF28 shows the average test accuracy for all 12 subjects. We can see that 3D CNN network alone performs worse than simpler baselines. But when coupled with AI-LSTM as Max CNN-LSTM, it shows an increase in recognition accuracy by 2% from AI-LSTM alone. This is because some of the signs are confused by the AI-LSTM network because of similar skeletal motion pattern. Incorporating spatial relationship among joints leads to a significant accuracy gain. The Spatial AI-LSTM is trained only on skeletal data but outperforms the combined network by 6%.

Figure FIGREF30 shows three confusion matrices for a subset of twelve sign classes for a subject. The top matrix is for AI-LSTM network, middle one is for Max CNN-LSTM and bottom one is for Spatial AI-LSTM. As seen in Figure FIGREF10 the sign pairs Alarm/Doorbell are similar in skeletal motion but have different hand shapes. Since Max CNN-LSTM includes hand shapes, it can successfully recognize it while other two models struggles. Same is true for some other signs like Email, Event, List, Order and Weather . Some other signs are better recognized by Spatial AI-LSTM network. It should be mentioned here that accuracy listed in Table TABREF28 shows average accuracy across all test subjects, while Figure FIGREF30 presents confusion matrix for a single test subject. For this particular subject overall test accuracy is 58%, 70% and 69% for AI-LSTM, Max CNN-LSTM and Spatial AI-LSTM network respectively.

Experiments ::: Effect of Same Subject Data in Training

In addition to having the cross subject accuracy described in section SECREF29, we also want to know the impact of adding a test subject's data to the training process. It is obvious that adding test subject's data to the training must increase the accuracy of the network for the subject. However, we want to know how much or what fraction of data is necessary for significant improvement in performance. This is important for assessing the practial usability of a recognition system. In other words, we want to know how quickly or with what amount of data, the current system can be adapted for a subject completely

unknown to the system. To do that, we first pick a test subject and train a model for the test subject with data from all other subjects in our dataset. Then we retrain the model with some fraction of data from the test subject. We keep increasing the fraction of data being used from the test subject in the retraining process up to 50%. The other half of the test subject's data is used for testing the model.

Figure FIGREF32 shows the effect of added training data from test subjects in the retraining on six subjects from our dataset in case of Spatial AI-LSTM model. We see that, adding data from a test subject increase recognition accuracy for all of the subjects shown. It is interesting to observe that adding even 10% of data from a test subject gives significant improvement in recognition accuracy (close to 95%) for almost all of the subjects shown.

Conclusion

We present a deep learning based approach for ASL recognition that leverages skeletal and video data. The proposed model captures the underlying temporal dynamics associated with sign language and also identifies specific hand shape patterns from video data to improve recognition performance. A new data augmentation technique was introduced that allowed the LSTM networks to capture spatial dynamics among joints. Finally, a large public dataset for ASL recognition will be released to the community to spur research in this direction; and bring benefits of digital assistants to the deaf and hard of hearing community. For future research direction, we are looking into the problem of sentence level ASL recognition. We also plan to use other data modality such as wifi signals which can be complimentary to video in sign language recognition.

Acknowledgments

This work was supported by Google Research Award. Some of the experiments were run on ARGO, a

research computing cluster provided by the Office of Research Computing at George Mason University, VA. (URL:http://orc.gmu.edu)