International Journal of Computer Vision and Image Processing
Volume 12 • Issue 1

A Deep Learning-Based Approach to Classification of Baby Sign Language Images

Sulochana Nadgeri, Sir Padampat Singhania University, India*
Arun Kumar, Sir Padampat Singhania University, India

ABSTRACT

Baby sign language is used by hearing parents to hearing infants as a preverbal communication, which reduces frustration of parents and accelerated learning in babies, increases parent-child bonding, and lets babies communicate vital information, such as if they are hurt or hungry. In the current research work, a study of various existing sign languages has been carried out as literature, and then, after realizing that there is no dataset available for baby sign language, the authors created a static dataset for 311 baby signs, which were classified using a MobileNet V1, pretrained convolution neural network (CNN). The focus of the paper is to analyze the effect of gradient descent-based optimizers, Adam and its variants, Rmsprop optimizers, on fine-tuned pretrained CNN model MobileNet V1 that has been trained using customized dataset. The optimizers are used to train and test on MobileNet for 100 epochs on the dataset created for 311 baby signs. These 10 optimizers, Adadelta, Adam, Adamax, SGD, Adagrad, RMSProp, were compared based on their processing time.

KEYWORDS

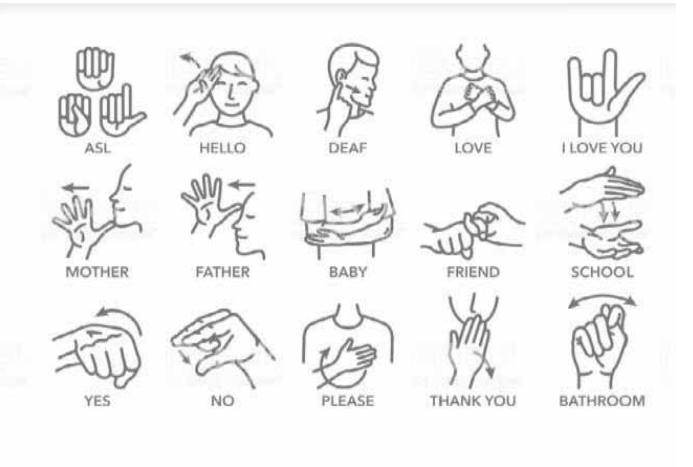
Baby Sign Language, Convolutional Neural Network (CNN), Deep Learning, Gesture Recognition, MobileNet, Optimizer, Sign Language Recognition

INTRODUCTION

Communication helps individuals to convey their thoughts and sentiments simultaneously, encouraging us to get feelings and thoughts of others. There are two methods we use for communication namely verbal and non-verbal communication. In verbal communication we speak with others while Non-verbal communication includes Facial expressions, gesture, posture, hand movements. Sign Language is majorly used by deaf and dumb community to express their emotions, thoughts, ideas with others. Different countries are using different Sign language similar to vernacular language therefore there is no universal Sign Language in existence. American Sign Language is the most widely used sign language in the world. Not only Deaf and Dumb community but also children with disorders like Autism, Apraxia of Speech Down Syndrome also gets benefited. Gesture, posture, eye gaze, facial expression plays a role in conversation using Sign Language. Static gestures or dynamic gestures are used for communication. Static gestures use hand postures made by the signer which are helpful to the signer to express his ideas and communicate with a normal person (Figure 1).

DOI: 10.4018/IJCVIP.2022010104 *Corresponding Author

Figure 1. *A small set of images depicting American Sign Language for words (courtesy:* **https://www.istockphoto.com/***)*

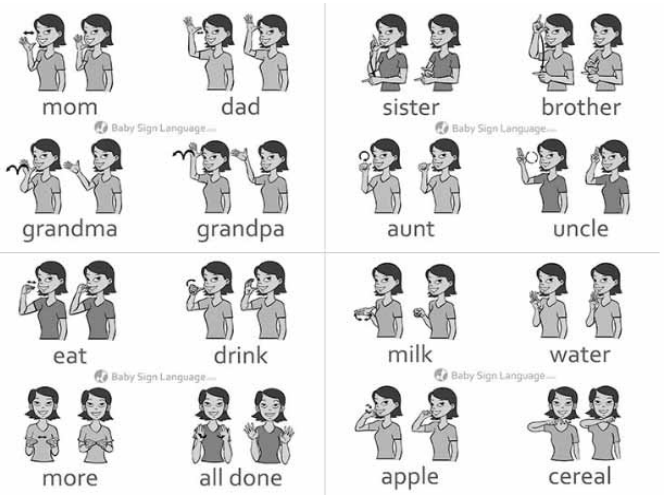


Sign language used by hearing parents of hearing children (aging between 6 months to 6-8 years) to enhance communication is known as Baby Sign Language. As we know infants start interacting through facial expressions with their caregivers at an early age, gazes and vocalizations such as cooing, crying is the primary mode of communication for infants. Crying is effective at evoking a variety of care giving responses, but a limitation of this form of communication is that caregivers must often rely on contextual cues to determine the appropriate response ()(). The signs for babies are different than type of Sign Language used by deaf or hearing hitch community. Baby sign language is a gesture-based communication preliminary to verbal communication, which really establishes connection amongst parents and infants. This early communication sets the establishment for accelerated learning, reduced frustration, and a closer connection between parent and child. Mellisa J. Cesafsky defines Baby Sign Language as a technique in which you and your infant (or toddler) use specific hand shapes, gestures and motions to convey phrases and meanings with each other quickly and easily. Typical hearing parents and infants employ baby Sign Language. In comparison with sign language used by deaf, Baby sign includes keywords and does not contain linguistic complexities as they are to be used with Infants (). The research was recently conducted by Stirling University's psychologist Dr. Gwyneth Doherty-Sneddon, UK on baby signing and confirms that signing enhances the child's vocabulary and mental growth, decreases tantrums and improves relationships with parents, as communication is fundamental for child's development ().

Babies who are taught to communicate using gestures are believed to enhance psychological benefits, such as increased certainty and confidence, for example: feelings of frustration may not occur as often because of an inability to communicate. When a child is too anxious to speak clearly, being able to use signs might be a support system. Teaching sign language along with speech has demonstrated development of spoken communication abilities. Infants use gesture plus speech combinations to form a sign. Sign language does not slow a baby's verbal development rather it boosts his/her enthusiasm to learn multiple communication techniques, including talking (Figure 2).

The paper is as Section 2 briefly summarizes the related research work carried out in this domain. The proposed methodology and architecture of Baby Sign Language has been described in section 3. Section 4 describes the experimental results and finally section 5 concludes the paper.

Figure 2. *A small set of images depicting Sign Language for words (courtesy:* **https://www.babysignlanguage.com/***)*



Related Work

As there is no universal sign language, based on different Sign languages in the world, researchers try to recognize different sign languages either using available datasets or develop their own database. The dataset they use is static or dynamic or in some cases both.

Agapito et al., (2014) and propose an Italian Sign language recognition system recognizes 20 Italian gestures. They used the CLAP14 dataset which is made up of Microsoft Kinect. The system uses the Microsoft Kinect, convolutional neural networks (CNNs) and GPU acceleration that gives 91.7% accuracy.

Chillarige et al., (2019) and proposed Indian Sign Language recognition system for 26 alphabets using a convolutional neural network (CNN). They tested CNN models on two different datasets which they have created and gives accuracy 99.40 when the background is static and only hand gestures are inputted to the system.

Dhruv implemented ASL recognition system for mobile platform which recognizes 24 alphabets from MNIST using Transfer Learning provides an accuracy of 95.03% of accuracy

implemented the i3d inception model to Sign Language Recognition with transfer learning method. 100% training accuracy achieved with 10 words and 10 signers with 100 classes but the validation accuracy is relatively low. They have experimented the dataset for more than one signer as they increased the number of signer accuracy decreases it became 0 for 4 signers and 40 words due to overfitting of the model.

presents a real-time American Sign Language (ASL) hand gesture recognizer which uses a GoogLeNet and AlexNet, pretrained Convolutional Neural Network (CNN) to train a dataset gives accuracy of 95:52% with GoogLeNet, and 99:39% with AlexNet after fine-tuning the model.

Neel Kamal proposed a real time Indian Sign language for alphabets and numbers in which the hand gesture based dataset is captured by the Microsoft Kinect RGB-D camera. They employed 3D construction and affine transformation of computer vision techniques. Convolutional Neural Networks (CNNs) model trained these 36 static gestures and achieved an accuracy of 98.81% on training using 45,000 RGB and depth images.

developed an algorithm based on vision that is used to interpret the sign from video to text that uses deep learning to classify 100 odd signs dynamic ASL performed under different lighting conditions. CNN accepts spatial features in the video and LSTM RNN deals with temporal features of video. The system offers 99% accuracy for the 600-sample dataset.

proposes a pre-trained CNN SqueezeNet model to recognize American Sign Language alphabets from RGB images. Images from dataset were resized and pre-processed before training them with Deep Neural Network model. The model is capable of running on mobile devices, which gives 87.47% training accuracy and 83.29% validation accuracy.

proposed a machine learning based classification in which texture based features were extracted using Gray Level Co-occurrence matrix (GLCM) and classified the prepared dataset of 60 baby signs using KNN and Random Forest algorithm. The accuracy of the system is 73% on the prepared dataset.

employs a Kinect sensor, which integrates the skeleton data, color data, and depth data to detect the palm using the Otsu thresholding method and morphology operators. The numerals and 20 alphabets in the English used in the sign language are predicted by SVM with a recognition accuracy rate of 70.59%

proposes Bhutanese Sign Language digits recognition system using the Convolutional Neural Network (CNN) and a dataset that has 20,000 sign images of 10 static digits collected from different signers. The proposed system has achieved 97.62% training accuracy.

The developed convolutional network () to recognize 24 finger spelling recognition for American Sign Language alphabets which take image intensity and depth data as an input . With the proposed architecture, the first block has two separate parts: One extracts the edges of RGB images; the other extracts the edges of the depth. The features are then combined in the second block. The developed convolutional network outperforms prior findings, with precision of 82% and recall of 80%, according to the evaluation.

proposes a real-time model for ISL gesture recognition, based on the incoming image data from the Kinect, Convolutional Neural Networks (CNNs) were utilized for training 36 static gestures of Indian Sign Language (ISL) alphabets and numbers. The model achieved an accuracy of 98.81% on training using 45,000 RGB depth images.

Methodology

Data

Standard dataset is not available for research purposes so we have to create our own. Baby sign language consists of keywords, which will be used by babies/ infants to convey their feelings, need. So it does not contain the signs for alphabet or number same as other sign language. Baby sign is fewer complexes as it does not follow linguistic structure as babies use it. Signs are made up of one or more than one gesture so while creating the dataset for research we have to acquire the static gesture independently then merge them in order to form a single gesture for a particular keyword. To acquire the image for keywords we have used a 20MP digital camera. Two signers perform each sign. The acquisition process is completed in a controlled environment. RGB images of signers, which represent some signs using hands, face, and head, are captured. The captured images have a resolution of 4608 X 3456 pixels. In different illumination conditions, two signers perform a gesture of each sign ten times. For the point of consistency, the backgrounds in all the images are the same and the duration of this process is almost 30 days. Sample signs from the dataset are shown in Figure 3. As shown in Figure 3 a sign may require one or more gestures.

For research, we considered 311 odd gestures, which are used frequently. The signs that involve a gesture sequence are combined to form a sign the gesture of the Baby Sign not just involves hands but also different parts of the face such as the chin, neck, ear, eyes etc.

Pre-Processing

The proposed framework utilizes a deep learning method for the recognition of the data utilized in the preparation stage, which plays an essential role in the entire system, so simply feeding the images captured in previous phase is not sufficient. We need a good mixture of the data that represents varying conditions and poses, varying sizes, if we expect the network can generalize well during

Figure 3. Glimpse of Baby Sign Language Dataset comprising sign with Single gesture, two gestures, three gestures, four gestures

the validation phase. Deep learning on occasion may run into difficulty where data has inadequate amounts. To get improved generalization of the model we require additional data and as much as discrepancy possible in the data. This is achieved using data augmentation.

Data augmentation is a process for expanding the size of a dataset arbitrarily by creating modified copies of the data in the dataset. Data Augmentation encompasses a suite of techniques that enhance the size and quality of training datasets such that better Deep Learning models can be built using them (). Data augmentation methods, for example, trimming, rotation, brightness, and even flipping are generally used for data augmentation (Figure 4).

Figure 4. Data Augmentation

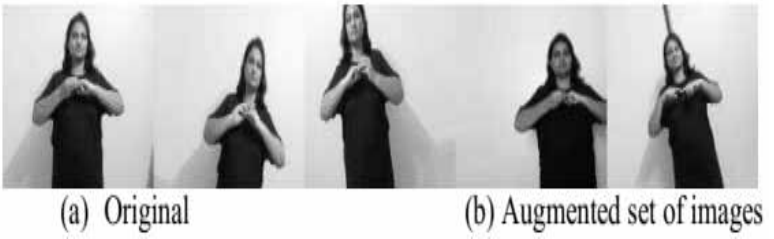
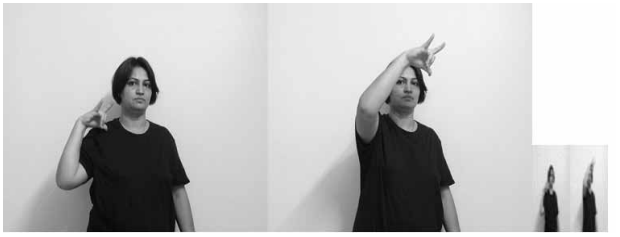


Figure 5. Resizing the image



Each sign is augmented using rotation, brightness, flipping, and transition techniques in order to increase size of the data. The augmentation performed on a single image had generated seven different variations of images.

As we present huge data to the deep learning model so we resize the image to 64x64 which drastically reduces computation time. Captured images are resized not only to reduce computational time but memory consumption also (Figure 5).

The Deep Learning model is supplied the resized image dataset. The dataset is split into training and test data sets, with the former accounting for 80% of the dataset and the latter for 20%.

Convolutional Neural Network (CNN)

Convolutional Neural Network (ConvNets or CNN), neural networks, primarily used with image dataset for image recognition, images classifications, Objects detections, recognition faces etc. ()(), CNN contains three main types of layers: Convolutional Layer, Pooling Layer, and Fully-Connected Layer (Figure 6).

Some popular CNN architectures AlexNet, VGGNet, GoogLeNet, ResNet, MobileNet, SqueezeNet (). In this work, we have chosen the MobileNet V1 model to train for Sign Classification, as it is lightweight. Instead of merging all three channels and flattening them, MobileNet uses depth-separable convolutions that carry out a single convolution on each color channel; these results in less computing time compared to other models.

Figure 6. General CNN Architecture

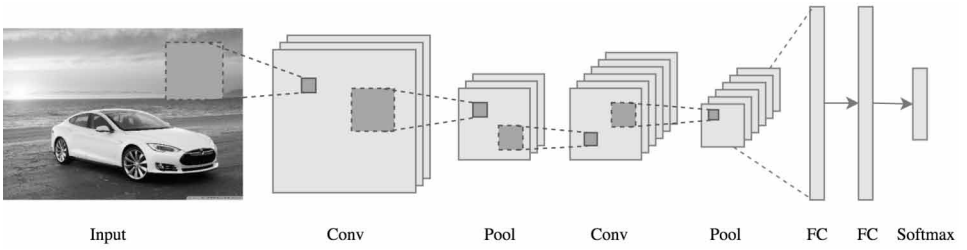
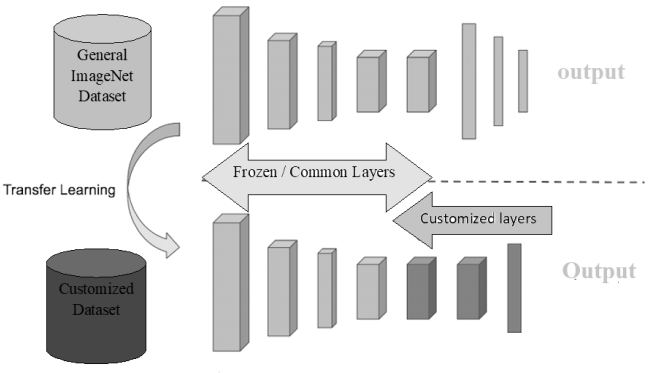


Figure 7. Transfer Learning Process



Transfer Learning

Convolutional Neural Network is very good at image classification" but training for a model computationally heavy (and sometimes not feasible). So we prefer Transfer Learning, a way to build models more accurately, is pre-trained models which are usually trained with Imagenet dataset. In transfer Learning the pre-trained model will behave as a base model whose performance can be improved or fit to the current problem by fine tuning the hyper parameters by adding some layers, changing learning rate, etc. Figure 7 shows general process Transfer Learning follows.

The Transfer Learning Model is developed for problems and can be applied to similar kinds of application by modifying the network as per requirement like the size of input image, Number of output classes, adding more layers etc. Transfer learning leads to lower the training time for a learning model and can end result in lower generalization error.

Therefore, as mentioned in CNN we have chosen MobileNet V1 pretrained model. The overall architecture of the MobileNet V1 is as follows, having 30 layers with

a. Convolutional layer with stride 2
b. Depthwise layer
c. Pointwise layer that doubles the number of channels
d. Depthwise layer with stride 2
e. Pointwise layer that doubles the number of channels ().

Figure 7a. *MobileNet V1 Body Architecture*

Type / Stride	Filter Shape	Input Size
Conv / s2	3 × 3 × 3 × 32	224 × 224 × 3
Conv dw / s1	3 × 3 × 32 dw	112 × 112 × 32
Conv / s1	1 × 1 × 32 × 64	112 × 112 × 32
Conv dw / s2	3 × 3 × 64 dw	112 × 112 × 64
Conv / s1	1 × 1 × 64 × 128	56 × 56 × 64
Conv dw / s1	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 128	56 × 56 × 128
Conv dw / s2	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 256	28 × 28 × 128
Conv dw / s1	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 256	28 × 28 × 256
Conv dw / s2	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 512	14 × 14 × 256
5× Conv dw / s1	3 × 3 × 512 dw	14 × 14 × 512
5× Conv / s1	1 × 1 × 512 × 512	14 × 14 × 512
Conv dw / s2	3 × 3 × 512 dw	14 × 14 × 512
Conv / s1	1 × 1 × 512 × 1024	7 × 7 × 512
Conv dw / s2	3 × 3 × 1024 dw	7 × 7 × 1024
Conv / s1	1 × 1 × 1024 × 1024	7 × 7 × 1024
Avg Pool / s1	Pool 7 × 7	7 × 7 × 1024
FC / s1	1024 × 1000	1 × 1 × 1024
Softmax / s1	Classifier	1 × 1 × 1000

In MobileNet V1, computationally expensive convolution layers are replaced with a simpler depthwise separable convolution, which is a 3×3 depthwise convolution layer followed by a 1×1 convolution layer

This takes fewer learned parameters than a standard convolution layer.

Modified architecture of MobileNet V1 used for research work as shown in Figure 7b where 2 fully connected layers added, output class value is modified as 311, the input image size is 64x64x3 rather than 224 x224x3.

Figure 7b. *Modified MobileNet Body Architecture*

Type/Stride	Filter Type	Input Size
Conv/S2	3x3x3x32	64x64x3
Conv dw /s1	3x3x3x32 dw	32x32x32
Conv/S1	1x1x32x64	32x32x32
Conv dw /s2	3x3x3x64 dw	32x32x64
Conv/S1	1x1x64x128	16x16x64
Conv dw /s1	3x3x3x128 dw	16x16x128
Conv/S1	1x1x128x128	16x16x128
Conv dw /s1	3x3x128 dw	16x16x128
Conv/S1	1x1x128x128	8x8x128
Conv dw /s1	3x3x128 dw	8x8x256
Conv/S1	1x1x128x256	8x8x256
Conv dw /s1	3x3x256 dw	8x8x256
Conv/S1	1x1x256x512	4x4x256
5x Conv dw /s1	3x3x512 dw	4x4x512
5x Conv/S1	1x1x512x512	4x4x512
Conv dw /s2	3x3x512 dw	4x4x512
Conv/S1	3x3x512 x1024	2x2x512
Conv dw /s2	3x3x1024 dw	2x2x1024
Conv/S1	1x1x1024x1024	2x2x1024
Avg. pool/S1	Pool 7x7	2x2x1024
FC/S1	1024x512	1x1x1024
FC/S1	512x341	1x1x512
Softmax/S1	classifier	1x1x341

Optimizer

After building the model, we need to train the model for the task and compute the loss, which is the difference between expected output and actual output (). This is accomplished by updating the network's parameters in response to the model's errors on the training dataset. The aim of an optimizer is to update the parameters of the neurons in such a way that the loss function is reduced. The Loss function serves as a reference for the optimizer, guiding it in the right direction so that the error is constantly decreased until either a good enough model is found or the learning process gets stuck and stops (). Optimizers are algorithms that change the training parameters of neural networks such as weights and biases in order to reduce the losses ().

Gradient Descent

This is the basic optimizer, which minimizes loss by taking first order derivative of the loss function and learning rate to scale back the loss and achieve the minima. The parameters are updated after calculating gradient for the whole dataset, which slows down the process. It also requires a large amount of memory to store this temporary data, making it computationally expensive. The algorithm works as follows

Initialize the weight w and bias b to some small value.

Set the learning rate α to small value like 0.001, 0.003 etc, the learning rate determines how big the step should be in each iteration.

If α is very small, it would take a long time to converge and become computationally costly.

If α is large, it may be unsuccessful to converge and overshoot the minimum.

On each iteration, take the partial derivative of the cost function J(θ) with respect to each parameter (gradient)

Update the parameter

$$\theta = \theta - \alpha \nabla_{\theta} \mathrm{J}\left(\theta\right) \tag{1}$$

Stochastic Gradient Descent (SGD):

In Gradient Descent the weight is updated just one occasion after considering the whole dataset, so gradient is usually overlarge theta (θ), can make bigger jumps so cannot reach to optimal solution .To overcome this problem Stochastic Gradient Descent (SGD), an extension of GD was introduced during which parameters are updated on every iteration. It means after every training sample, the loss function is tested and therefore the model is updated but the convergence are going to be very slow.

$$\theta = \theta - \alpha \nabla_{\theta} \mathrm{J}\left(\theta; \mathrm{sample}\right) \tag{2}$$

Mini Batch Gradient Descent

To reduce overhead computation for every iteration, mini batch gradient descent, a variation of GD is evolved which updates the parameters after a subset of minibatch of dataset. Typically the size of mini batch are 32, 64,128 etc.

$$\theta = \theta - \alpha \nabla_{\theta} \mathrm{J}\left(\theta; \mathrm{Batch}\right) \tag{3}$$

Mini batch Stochastic Gradient Descent doesn't compute the exact derivative of loss function. Instead, it estimates loss on a small batch which means it is not always going in the optimal direction, because derivatives are 'noisy'? Exponentially weighed averages can provide us a better estimate which is closer to the actual derivate than noisy calculations.

SGD with Momentum

In order to update the parameters of the network, a new parameter which is called momentum considered. SGD with momentum consider current gradient as well as the previous gradients which are accumulated. The equations of gradient descent with momentum are revised as follows.

$$\mathrm{v}_{\mathrm{t}=} {}^{3} \mathrm{v}_{\mathrm{t}-1} + \pm \nabla_{\theta} \mathrm{J}\left(\theta\right) \tag{4}$$

$$\theta = \theta - \mathrm{v}_{\mathrm{t}} \tag{5}$$

NAG(Nesterov accelerated gradient)

SGD with momentum considers previous momentums due which there are chances of overshoot i.e. will not find optimum solution. NAG resolves this concern by computing look ahead term called projected gradient. This value is often obtained by going 'one step ahead' using the previous velocity. This looking ahead improves the performance of Gradient Descent algorithm.

$$v_t = {}^{3} v_{t-1} + \pm \nabla_{\theta} J\left(\theta - {}^{3} v_{t-1}\right) \tag{6}$$

$$\theta = \theta - v_t \tag{7}$$

$\theta - \gamma v_{t-1}$ is the gradient of looked ahead

Adagrad (Adaptive Gradient Algorithm)

All gradient-based optimizers are incapable to handle variety in input data due to constant learning rate.

Adagrad adapts the learning rate that can adjust according to the input provided, decrease the learning rate more rapidly for parameters that have large gradient components and slower for ones that have a smaller gradients.It changes the learning rate 'α' for each parameter and at every time step 't'. It is based on a second order derivative which is computationally costly.

$$g_{t,i} = \nabla_{\theta} J\left(\theta_{t,i}\right) \tag{8}$$

$g_{t,i}$ is then the partial derivative of the loss function with respect to the parameter θ_i at time step t:

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\alpha}{\sqrt{G_{t,ii} + \varepsilon}} . g_{t,i} \text{ (9) ()}$$

RMS Prop(Root Mean Square Propagation)

The learning rate is divided by an exponentially decaying average of squared gradients by RMSprop (). RMSprop discards history.

$$E\left[g^2\right]_t = \gamma E[g^2]_{t-1} + \left(1-\gamma\right) g_t^2 \tag{10}$$

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{E[g^2]_t + \varepsilon}} \; g_t \tag{11}$$

Where γ=0.9

Instead of computing exponentially decaying average of squared gradient Adadelta limits the window of accumulated previous gradients to fixed size w.

$$\nabla\theta_t = -\frac{RMS\left[\Delta\theta\right]_{t-1}}{RMS\left[g\right]_t} g_t \tag{12}$$

$$\theta_{t+1} = \theta_t + \Delta\theta_t \tag{13}$$

Where

$$RMS[\Delta\theta]_t = \sqrt{E[\Delta\theta^2]_{t\ +}\varepsilon} \tag{14}$$

Where $E\left[\theta^2\right]_t$ is running average of squared parameter updates.

$$E\left[\theta^2\right]_t = \gamma E[\theta^2]_{t-1} + \left(1-\gamma\right)\Delta\theta_t^2 \tag{15}$$

Adam

Adam is another adaptive strategy that incorporates the best aspects of both RMSProp and momentum optimizers.

Adam holds an exponentially decaying average of previous squared gradients v_t like RMSProp and therefore the average of previous gradients decays exponentially $\mathrm{m_t}$ almost like momentum optimizers. The parameters β1 and β2 control the decay rates of these moving averages ()()().

$$m_{t=}\beta 1 m_{t-1} + \left(1-\beta 1\right) g_t \tag{16}$$

$$v_t = \beta_2\, v_{t-1} + \left(1-\beta_2\right) g_t^2 \tag{17}$$

Where m_t first moment of gradients, v_t is second moment of gradients, $\beta 1, \beta 2$ are decay rates. $\mathrm{m_t}$, v_t are initialized to 1.0, so bias of moments inclined towards 0. This bias is overcome by first calculating the biased estimates before then calculating bias-corrected estimates

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{18}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{19}$$

Adam update rule

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{\hat{v}_t + \epsilon}}\ \hat{m}_t \tag{20}$$

Adam adapts learning rate based on average of second moments of gradients (variance).

Adamax (Adaptive Moment Estimation):

The v_t factor in the Adam update rule scales the gradient inversely proportionally to the $\ell 2$ norm of the previous gradients (v_{t-1} term) and current gradient $\left[g_t\right]^2$.

$$v_t = \beta_2 v_{t-1} + \left(1-\beta_2\right) | g_t |^2 \tag{21}$$

Instead of this we can generalize the update to the ℓ_p norm.

$$v_t = \beta_2^p v_{t-1} + \left(1-\beta_2^p\right) | g_t |^p \tag{22}$$

Put p->∞

Adamax use u_t to denote the infinity norm-constrained v_t, to avoid any misunderstanding with Adam optimizer

$$u_t = \beta_2^\infty v_{t-1} + \left(1-\beta_2^\infty\right) | g_t |^\infty \tag{23}$$

$$= \max(\beta_2.v_{t-1}, | g_t | \tag{24}$$

Now substitute u_t into the Adam update equation for replacing $\sqrt{v_t} + \varepsilon$ .We obtain the Adamax update rule:

$$\theta_{t+1=}\theta_t - \frac{\alpha}{u_t}\ \hat{m}_t \tag{25}$$

u_t relies on the max operation. Default values are α =0.002, β_1=0.9 and β_2=0.999 gives optimum results.

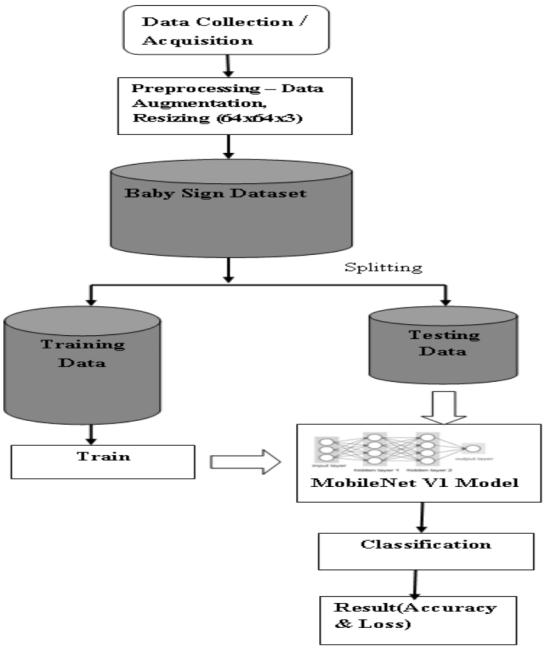
Nadam (Nesterov-accelerated Adaptive Moment Estimation):

Nadam fuses Adam and NAG optimizers. In order to contain NAG into Adam, it replaces $\hat{m}_t$ with momentum term m_t.

By integrating the exponential decay of the moving averages for the past and current gradients, the learning process can be accelerated.

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{\hat{v}_t + \epsilon}} \; m_t \tag{26}$$

Figure 8. Block Diagram of Proposed System



Proposed Architecture

Following Steps are used to accomplish the result (Figure 8)

1. Data collection: The dataset used for research work is our own data in which 311 different categories of Baby Sign Language sign have been captured using Nikon Digital camera with 20MP resolution. As we are aware that some sign require a single gesture and some require more than one gesture, the images are merged in order to present as a single sign. Dataset contains signs performed by 2 signers. The sample images are presented in Figure 3.

2. Data augmentation (Pre-processing): Transform the size of all sign images from (4608 X 3456x 3) to (64 x 64 x 3) to reduce computational resources. To provide a diversified dataset to CNN model we augmented the original dataset.
3. Split dataset (80:20) into a training set of 17418 images and a test set of 4319 images belonging to 311 classes. The training set is also divided the data into batches, and performed shuffling for the rearrangement of the images randomly.
4. Define proposed MobileNet V1 model architecture with modification
   a. 2 Fully Connected layers with dropout at the top of the base model.
   b. The last fully connected layer's shape is the number of classes
   c. The activation function is Softmax.
   d. Retrains all the layer above layer 3 of the model
   e. L2 Regularizer is connected to the first fully connected layer.
5. After experimenting number of times with different optimizers like Adam, Adamax, SGD, Adagrad, RMSProp, Nadam, etc. we got optimum result when we train all layer above layer 3 with training options as all different optimizer, categorical-cross entropy loss function, L2 Regularizer to reduce overfitting and initial learning rate = 0.00001 but it decrease during training when Validation_loss becomes stagnant by factor of 0.02 .
6. Fit and train the data for mini batch_size = 64 and epochs = 100
7. Verify the system on validation data, calculate the training and test accuracy, training, and test loss.

Result and Discussion

The experiments carried out on our original dataset for all optimizers. For MobileNet V1 architectures, Nadam and Adamax give the best accuracy on training data. On the other side, Adadelta gives the worst performance during training as shown in Figure 9 and Figure 10. Accuracy of Adadelta optimizer did not improve throughout training. RMSProp begins training with the highest loss while Adamax begins with the lowest. Therefore, it can be concluded that Adamax makes a better start the training by using the same parameter initialization. Adam and its variants Nadam, Adamax almost gives better results as compared to other optimizers. Gradient based algorithm SGD with Nesterov and momentum=0.9 gives better results than other variants. While training the model the learning rate is not constant, validation loss is monitored if there is no change in it then learning rate is reduced. As shown in Figure 11 for the Adagrad algorithm the learning rate is not reduced as frequently as the rest of the optimizers.

Figure 9. The behaviour of optimizer on user dataset during training for MobileNet

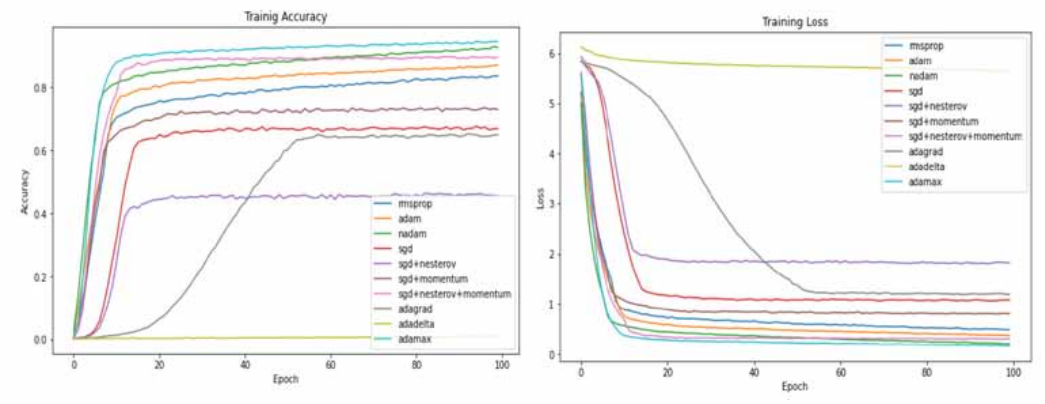


Figure 10. The behaviour of optimizer on user dataset during Validation for MobileNet

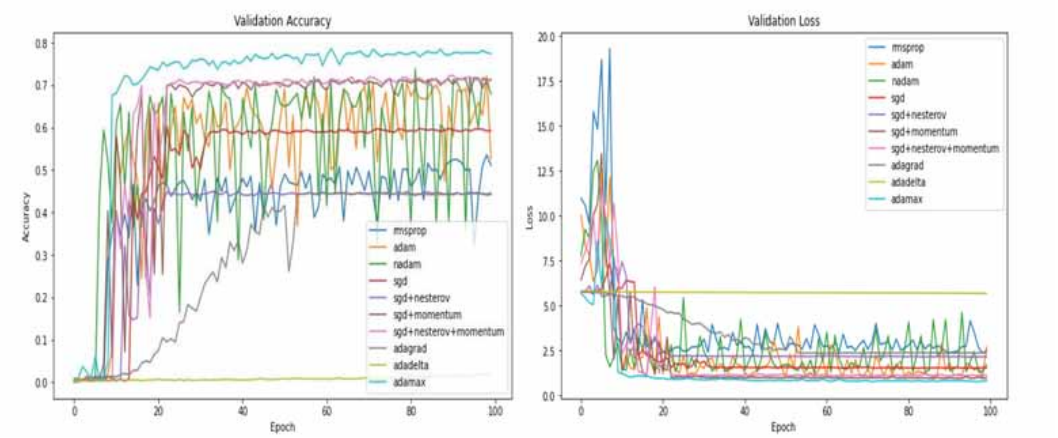


Figure 11. Learning Rate of optimizer

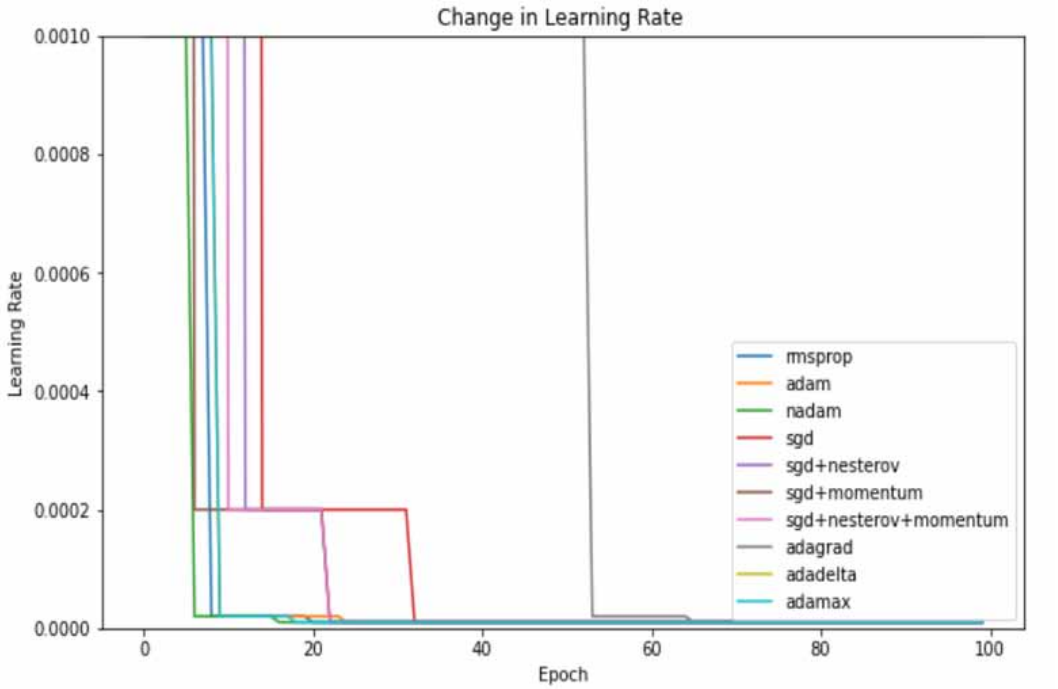
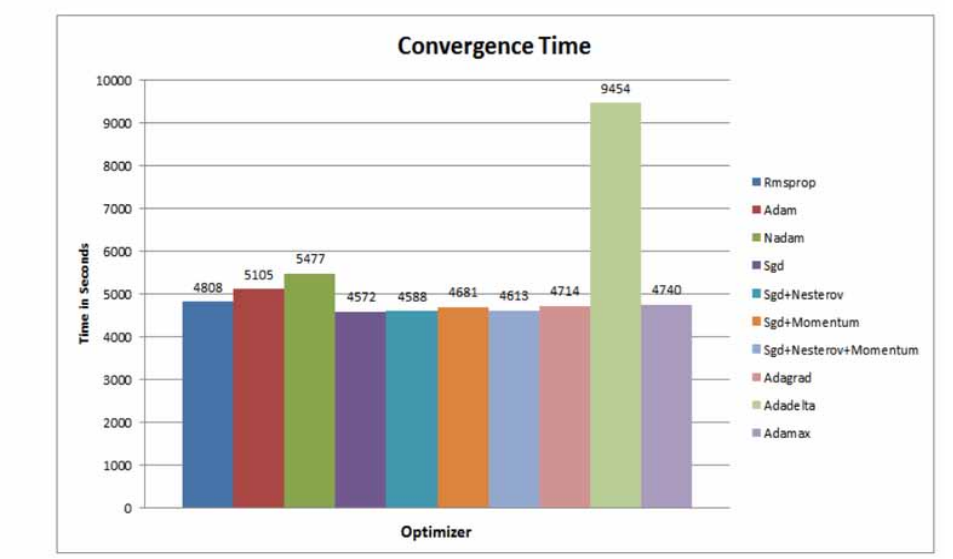


Figure 12. Comparison of the training speed



In terms of the performance of the classification of Baby Sign Language, Adamax, which is a variant of ADAM, is the best choice to optimize the models. However, performance, the only parameter 3+602. to consider determine whether a model is good cannot be certain, so the training speeds of the models are compared. It is clear that the model with SGD optimizer converges faster at the same number of iterations. This indicates that the model with Adamax optimizer can achieve a suitable solution with optimum time. Comparison of the performance of the models SGD optimizations has the fastest training speed, while Adadelta has the slowest training speed. The specific training speed values of the models are shown in Figure 12. Thus, the experimental results indicate that Adadelta is the best choice for optimization. It outperformed with training and validation accuracy of 98.18% and 78.60%, respectively, on the RGB image dataset. The results have also been evaluated based on accuracy and loss.

CONCLUSION

In this research, we presented an effective method to recognize almost 311 Baby Signs, which consists only words mostly used in daily routine and which helps in developing a bond among child and parents. The proposed pretrained CNN model MobileNet V1with 2 fully connected layers on top it with drop out of 0.5 which increase speed and accuracy of the model.

The system results in the highest training and validation accuracy of 98.18% and 78.60%, respectively, with respect to change in hyper parameters such as the number of epochs, no. of layers to retrain in order to detect the feature, etc.

It is possible to boost the dataset in the future by obtaining more samples which consider the lighting conditions, more number of signer to get diversity in the dataset and distance of the signer from the camera. In addition, the system will be extended to identify dynamic Baby Signs that involve capturing and creating a video-based dataset which contains both temporal and spatial aspects, are included in a variety of images, and the system will be tested by slicing the videos into frames and classifying using the CNN model.

REFERENCES

Ameen, S., & Vadera, S. (2017). A convolutional neural network to classify American Sign Language fingerspelling from depth and colour images. *Expert Systems: International Journal of Knowledge Engineering and Neural Networks*.

Bhagat, N. K., Vishnusai, Y., & Rathna, G. N. (2019). Indian Sign Language Gesture Recognition using Image Processing and Deep Learning. In Digital Image Computing: Techniques and Applications (DICTA) (pp. 1-8). Academic Press.

Bindal. (2021). *Some State of the Art Optimizers in Neural Networks*. https://medium.com/techspace-usict/some-state-of-the-art-optimizers-in-neural-networks-a3c2ba5a5643

Das, , Gawde, , Suratwala, , & Kalbande, . (2018). Sign Language Recognition Using Deep Learning on Custom Processed Static Gesture Images. *International Conference on Smart City and Emerging Technology (ICSCET)*.

Donoghue, E. C. (2014). *Sign Language and Early Childhood Development, Rehabilitation, Human Resources and Communication Disorders.* Undergraduate Honors Theses. 20.

Halgamuge, M.N., Daminda, E., & Nirmalathas, . (2020). A. Best optimizer selection for predicting bushfire occurrences using deep learning. *Natural Hazards*, *103*, 845–860.

Howard, Zhu, Chen, Kalenichenko, Wang, Weyand, Andreetto, & Adam. (2017). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. Academic Press.

Khan, A., Sohail, A., & Zahoora, U. (2020). A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, *53*, 5455–5516.

Morocho-Cayamcela, M. E., & Lim, W. (2019). Fine-tuning a pre-trained Convolutional Neural Network Model to translate American Sign Language in Real-time. In *2019 International Conference on Computing, Networking and Communications (ICNC)* (pp. 100-104). Academic Press.

Nadgeri, S., & Kumar, A. (2019). Image Texture based approach in understanding and classifying Baby Sign Language. *2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, 854-858.

Pigou, L., Dieleman, S., Kindermans, P. J., & Schrauwen, B. (2014). Sign Language Recognition Using Convolutional Neural Networks. In L. Agapito, M. Bronstein, & C. Rother (Eds.), Lecture Notes in Computer Science: Vol. 8925. Computer Vision - ECCV 2014 Workshops. ECCV 2014 (pp. 572–578). Academic Press.

Pigou, L., Dieleman, S., Kindermans, P.-J., & Schrauwen, B. (2015). *Sign Language Recognition Using Convolutional Neural Networks. Academic Press.*

Pizer, G., Walters, K., & Meier, R. P. (2007). Bringing Up Baby with Baby Signs: Language Ideologies and Socialization in Hearing Families. *Sign Language Studies*, *7*(4), 387–430.

Poojary, R., & Pai, A. (2019). Comparative Study of Model Optimization Techniques in Fine-Tuned CNN Models. *2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA),* 1-4.

Rathi, D. (2018). Optimization of Transfer Learning for Sign Language Recognition Targeting Mobile Platform. *International Journal on Recent and Innovation Trends in Computing and Communication*, *6*(4), 198–03.

Ruder, S. (2012). An overview of gradient descent optimization algorithms. https://ruder.io/optimizing-gradient-descent/

Ruder, S. (2016). *An overview of gradient descent optimization algorithms*. Academic Press.

Sah. (n.d.). *Convolution-In Deep Learning*. https://medium.com/@eshant.sah/convolution-in-deep-learning-62eb08b42070

Sarkar, A., Talukdar, A. K., & Sarma, K. K. (2019). CNN-Based Real-Time Indian Sign Language Recognition System. In R. Chillarige, S. Distefano, & S. Rawat (Eds.), *Advances in Computational Intelligence and Informatics. ICACII 2019* (Vol. 119, pp. 71–79). Lecture Notes in Networks and Systems.

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, *6*(60).

Smolyakov. (n.d.). Neural Network Optimization Algorithms. https://towardsdatascience.com/neural-network-optimization-algorithms-1a44c282f61d

Suharjito, Gunawan, Thiracitta & Nugroho. (2018). Sign Language Recognition Using Modified Convolutional Neural Network Model. In *2018 Indonesian Association for Pattern Recognition International Conference (INAPR)* (pp. 1-5). Academic Press.

Sun, Y. (2019). Convolutional Neural Network Based Models for Improving Super-Resolution Imaging. *IEEE Access : Practical Innovations, Open Solutions*, *7*, 43042–43051.

Tao, W., Leu, M., & Yin, Z. (2018). American Sign Language Alphabet Recognition Using Convolutional Neural Networks with Multiview Augmentation and Inference Fusion. *Engineering Applications of Artificial Intelligence*, *76*, 202–213.

Thompson, R. H., Cotnoir-Bichelman, N. M., McKerchar, P. M., Tate, T. L., & Dancho, K. A. (2007). Enhancing early communication through infant sign training. *Journal of Applied Behavior Analysis*, *40*(1), 15–23. PMID:17471791

Wangchuk, K., Riyamongkol, P., & Waranusast, R. (2020). *Real-time Bhutanese Sign Language digits recognition system using Convolutional Neural Network*. ICT Express.

Yale, M. E., Messinger, D. S., Cobo-Lewis, A. B., & Delgado, C. F. (2003). The temporal coordination of early infant communication. *Developmental Psychology*, *39*(5), 815–824. PMID:12952396

Yeh, W., Tseng, T., Hsieh, J., & Tsai, C. (2016). Sign language recognition system via Kinect: Number and english alphabet. *International Conference on Machine Learning and Cybernetics (ICMLC)*, 660-665.

Yi, D., Ahn, J., & Ji, S. (2020). An Effective Optimization Method for Machine Learning Based on ADAM. *Applied Sciences*, *10*(1073).