Sign Language Interpretation using Ensembled Deep Learning Models

Samarth Khanna[1] and *Kabir* Nagpal[1*]

[1]Manipal University Jaipur, Jaipur, Rajasthan, India

Abstract. Communication is an integral part of our day-to-day lives. People experiencing difficulty in speaking or hearing often feel neglected in our society. While Automatic Speech Recognition Systems have now progressed to the purpose of being commercially viable, Signed Language Recognition Systems are still in the early stages. Currently, all such interpretations are administered by humans. Here, we present an approach using ensembled architecture for the classification of Sign Language characters. The novel ensemble of InceptionV3 and ResNet101 achieved an accuracy of 97.24% on the ASL dataset.

1 Introduction

One can only imagine how hard it is for people who are facing auditory issues or have a hearing impairment. Be it at a social gathering or the worst case is when they are facing a medical emergency. The deaf and dumb community relies heavily on professional sign language interpreters to be able to even have a basic conversation. We could not help but notice that these people while being no different than us are still excluded just because of a lack of support.

The identification of sign languages is a difficult problem since many sign languages, each of which has its own set of grammatical rules, reflect various spoken languages. Another reason to be concerned is the possibility that different sign languages may interpret the same gesture differently. [1]. While we have state-of-the-art translation methodologies, sign language recognition remains largely unexplored.

But every big problem needs to be tackled step-by-step. We aim to translate characters and numbers of a language with higher accuracy to establish the robustness of the model. As characters are the building blocks of any language and are discrete, so it makes sense to test the model upon a character dataset first and then move on to word and sentence level.

While tackling this problem we found that a problem like this needs a dataset that not only contains all the relevant characters but also depicts real-life scenarios. For example, the test images might be noisy, the background will never be constant, and the images will be captured under different lighting conditions. The gestures could be slightly rotated and even the size and type of hand might be different.

*Corresponding author: 21kabirnagpal@gmail.com

A dataset with no noise would give a better accuracy but the model therefore trained would not suffice as a robust model. All such variations must be present in the training dataset to maintain consistency in the results while testing in real-time.

We have used the dataset mentioned in [8] because it managed to fit the criteria, we set for the dataset to be appropriate as the images contained a fair amount of noise and the same characters were captured in different backgrounds, lighting conditions, and different alignments as in Figure 1. We preferred to use an image-based dataset which can be later extended to a video dataset. Videos are frames and hence a model which would give a better accuracy in the mentioned dataset would certainly be eligible for testing on a wide variety of datasets of not only images but videos as well.

Fig. 1. A subset of Images of Bengali Dataset

2 Literature Review

2.1 Related Work

Several pieces of research have been conducted for automatic speech recognition however very few unique research have been performed for sign language recognition. The conducted research can be distinguished based on their usage. While the primary research includes a static corpus of individual characters or words that forms the base of all research, advanced models have been built with advances in technology and powerful GPUs.he title is set in bold 16-point Arial, justified. The first letter of the title should be capitalised with the rest in lower case. You should leave 22 mm of space above the title and 6 mm after the title.

The authors in [2] used depth cameras and random forests to estimate American sign language with 92% accuracy using Microsoft Kinect. The generated images were segmented to depict various regions of hands such as palm fingers etc. using per-pixel classification. Various join angle features were then estimated and fed into the random forest.

A similar estimation technique is also used by [3] on the Korean sign language dataset consisting of videos as input. The estimation technique was however applied to complete the upper body and normalized features were used to train a sequence-to-sequence network. This achieved an overall accuracy of 93.28%.

As convolutions are excellent in learning and extracting features, [6] and [7] used 3D residual convolution and 2D LSTM and HMM. Fine-tuned VGG network was used by [8,9] to recognize sign language on the Bengali dataset consisting of images. [10] also proposed a CNN for Bengali data with architecture similar to VGG.

2.2 Deep Learning Architectures Used

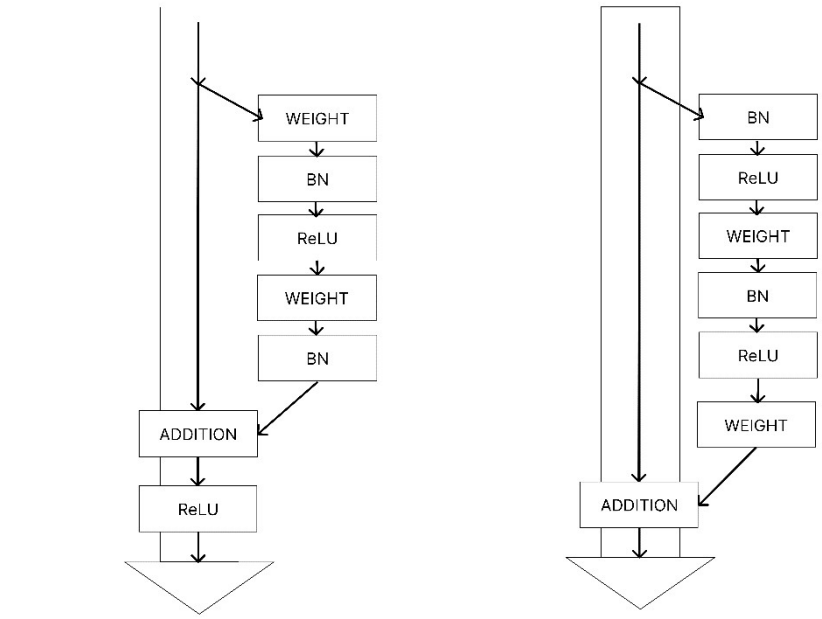
The VGG network, which earned First Place in the competition in 2014, was one of the first open-source contributions of ImageNet weights. K. Simonyan et al. [12] developed a

heuristic examining the association between accuracy and the degree of Convolutional layers. Several convolutional layers with a 3x3 kernel size were arranged with the pooling layer. According to the authors, the best models are 16-layer and 19-layer variants. Although the the proposed work established a direct relationship between the number of layers to performance, further variants suffered a vanishing gradient problem.

H. Kaiming Et Al. [13] introduced the Residual network in 2015, and it won various competitions. The Skip Connections concept was proposed, which aided in constructing deeper networks. Working on the VGG network's shortcoming, the authors discovered insufficient features for lower layers to work on, resulting in vanishing gradients. On the other hand, Skip Connections avoids this flaw by forwarding the weights of earlier layers to newer layers. There is also a reduction in the network's overall complexity compared to an equal number of layers in the VGG network. The researchers finalized three variants with 50, 101, and 152 layers.

The work was even taken forward in [15] to change a few layers, as shown in the image. The author concluded that activation after identity is necessary for greater generalization and smooth weight propagation. The methods outperform equivalent earlier residual networks by 1-1.5 percent on the ImageNet dataset.

The authors of [14] investigated the height and width of the network and proposed that a balance is essential for optimal network performance. They also discovered that a 5X5 kernel is 2.78 times more computationally expensive than a 3X3 kernel but extracts comparably more features. As a result, they divide the filter into several smaller filters, as shown in Fig 3. [18] reproduced a similar design with skip connections, which improved top-five accuracy by 2.5-3 percent in numerous contests, including ImageNet.



(a) (b)

Fig. 2. Representation of Residual Block in originally proposed (a) ResNet and Version 2 (b).

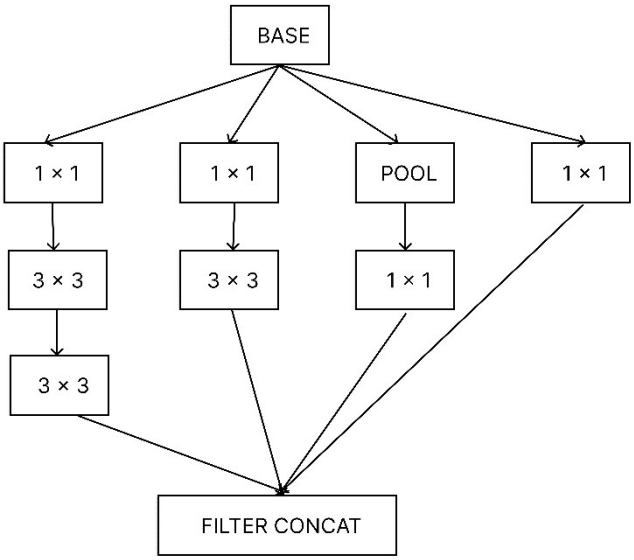


Fig. 3. Representation of a 5X5 filter convolution module using 3X3 and 1X1 filter convolutions

F. Chollet published Xception net in his work [16], which replaces convolution with Depth Based Separable Convolutions, continuing the research of Inception Net. He theorized that typical filters with two dimensions (height and width) map cross-channel and spatial correlations simultaneously, but the depth-wise separable works on each channel individually and then combines the results. The method is beneficial because data from different channels are multiplied by a separate kernel, extracting the degree of characteristics depending on the layer. On ImageNet, the model outperforms Inception V3 achieving 94.5% accuracy.

In 2018, [17] introduced Dense ConvNet. It included skip connections, just like ResNet, but each layer receives characteristics from all previous layers. As a result, the total number of skip connections in this architecture is of the order of n squared, where n represents number of layers. This resulted in a significantly deeper network, feature reuse, and resistance to vanishing gradients. There are three versions of the model: 121, 169, and 201 layers.

Z. Barret Et Al. introduced NASNet [19], which they designed using the NAS [20] reinforcement learning algorithm. To create a block, NAS calculated each parameter, such as the number and type of layers. Filter decomposition as in Inception Net, and separable layers as in Xception Net, are also observed in this cell. The NAS-generated block was replicated four to seven times to create a complete network. On ImageNet, the network obtained a performance of 96 percent.

3 Methodology

Transfer learning is a prominent method for solving several modern machine learning problems. An algorithm or a deep learning model is trained on a similar task but contains a large dataset. The model's weights are extracted and retrained on the given problem statement. Because the initial coefficients (model weights) for the new dataset are not random and contain the context from previous training, they converge faster, resulting in better results with fewer data points.

This methodology has many applications, particularly in computer vision, where new algorithms/ architectures are firstly benchmarked on the 1.2 million images in the ImageNet [23] dataset and then saved for future use. We use the weights open-sourced on Keras [11] to quantify the performance of several models in this proposed work for sign languages. These models are as below:

3.1 Generalized Architecture

For each model briefed in section 4.1, we employed the following architecture (Table 1) to standardize the results. The input layer alters the input images according to model configurations, including changes in size and other augmentation. After joining the pre-trained model, two dense layers with L2 regularizers and a final SoftMax activation layer are added. Regularization is employed to control overfitting by introducing bias to the error. Reduce Plateau and Early-Stopping with a patience of 5 epochs, optimally puts a stop to the model training.

Table 1. General Architecture of Network Used

Layer No.	Name	Specifications
1	Input Layer	Resize according to requirements of architecture
2	Existing Architecture	-
3	Flatten	-
4	Dense	Activation = Relu, Shape = 512
5	Dense	Activation = Relu, Shape = 64
6	Activation	SoftMax, Shape = 38

Different pooling layers and optimizers are utilized in each experiment. Each experiment has a preset learning rate of 0.001. The experiments conducted are as in table 2.

Table 2. General Architecture of Network Used

Experiment No.	Name	Pre-Trained Weights	Pooling
1	Adam	No	Maximum And Average
2	Adam	Yes	Maximum And Average
3	SGD	No	Maximum And Average
4	SGD	Yes	Maximum And Average

The Convolution neural network algorithm is created in Python [21] using the Keras library. Because of the free availability of GPU to enhance model performance, Google Colab [22] is employed as an environment. Figure 4 depicts how the findings are saved in the database.

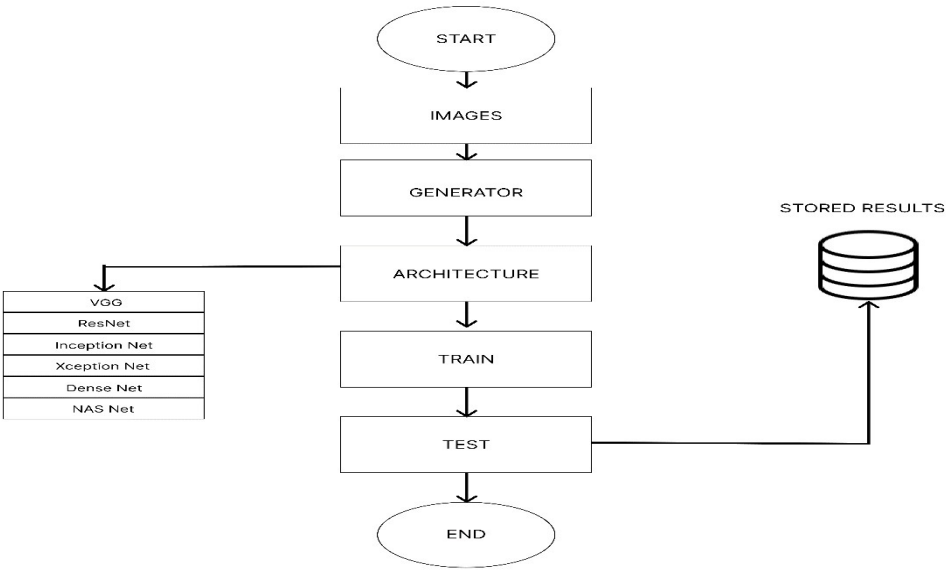


Fig. 4. Method Architecture for Training and Testing Each CNN Architecture

Images are passed to a generator where they are loaded in batches of 32 to conserve RAM. These images are passed to one of the architectures briefed in section 3. Images are trained with parameters as per Table 1. The test results are then stored in a CSV file.

The initial round of experiment was carried out without the use of weights in the Adam optimizer. The second round was carried out with the Adam optimizer and pre-trained weights. The third round was carried out with the SGD optimizer and no weights. The final round was carried out with pre-trained weights and the SGD optimizer. InceptionV3 and ResNet101.

Pre-trained models and configurations that produce the best results are pooled to form one ensemble model and trained on the same general architecture, to obtain the final result. The ensemble models combine the final layer outputs of the two best models to generate a single layer. This layer is subsequently transmitted to layer 4 as in table 2.

4 Results

During the first round of testing, Xception Net with Avg pooling achieved an accuracy of 89 percent. The VGG architecture, as well as ResNet 152, ResNet50, MobileNetV2, and NasNet Mobile, did not converge.

The second round of testing showed that Xception Net on Average pooling retained their lead, along with Inception V3 with a resultant accuracy of more than 96 percent, but VGG failed to converge. In the third round of testing, none of the network architectures were able to converge. This might be due to the optimizer's default low learning rate.

InceptionV3 and ResNet101 both achieved high accuracies of 96.78% and 96.64% respectively in the final round of testing.

The top models from the previous experiment were utilized to construct an ensemble model that employed InceptionV3 and ResNet101 in parallel. Figure 5 shows a representation of the architecture. Pre-trained weights and the SGD optimizer were able to achieve

maximum accuracy of 97.24 percent in fewer than 20 epochs (Figures 6a and 6b). All the remaining hyperparameters are set to the values shown in Table 1.

Table 3. Results of all the performed experiments except experiment 3 because it failed to converge for all models

Model Names/	**Experiment 1**		**Experiment 2**		**Experiment 4**	
Pooling Layers	**AVG**	**MAX**	**AVG**	**MAX**	**AVG**	**MAX**
DenseNet121	0.85	0.8066	0.9342	0.9474	0.9638	0.9625
DenseNet169	0.8355	0.8118	0.9303	0.9336	0.9651	0.9638
DenseNet201	0.0263	0.8217	0.9276	**0.9533**	0.9592	0.9645
InceptionResNetV2	0.8882	0.8046	0.9434	0.9447	0.9579	0.9618
InceptionV3	0.8711	0.8664	**0.9618**	0.9283	**0.9678**	**0.9651**
Mobile Net	0.8342	**0.8803**	0.9559	0.9395	0.9579	0.9533
MobileNetV2	0.0263	0.0263	0.925	0.9388	0.9625	0.9566
NASNetMobile	0.0309	0.0263	0.7849	0.5046	0.9507	0.9408
ResNet101	0.0276	0.7836	0.9428	0.9026	0.9664	0.9474
ResNet101V2	0.7401	0.7875	0.9276	0.9092	0.9586	0.9605
ResNet152	0.0263	0.0263	0.9302	0.9092	0.9612	0.9546
ResNet152V2	0.8151	0.8151	0.9322	0.9112	0.9638	0.9586
ResNet50	0.0263	0.0289	0.9349	0.9158	0.9592	0.9586
ResNet50V2	0.0263	0.8513	0.9276	0.9283	0.9566	0.9592
VGG16	0.0263	0.0263	0.0263	0.027	0.9625	0.9612
VGG19	0.0263	0.0263	0.0263	0.0263	0.9605	0.9605
Xception	**0.8967**	0.8711	0.9579	0.0263	0.9533	0.9467

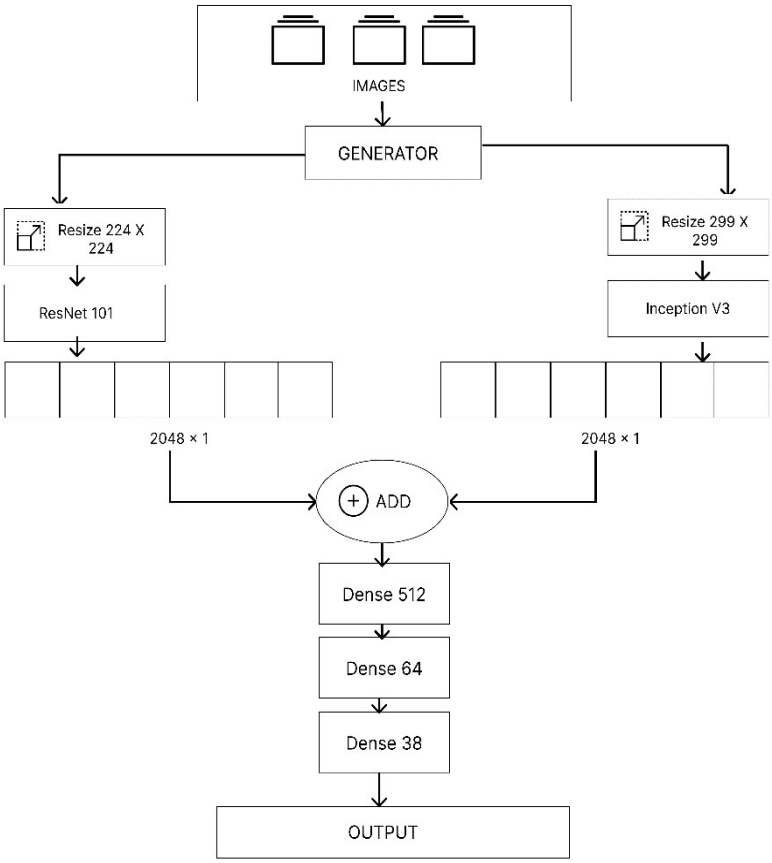
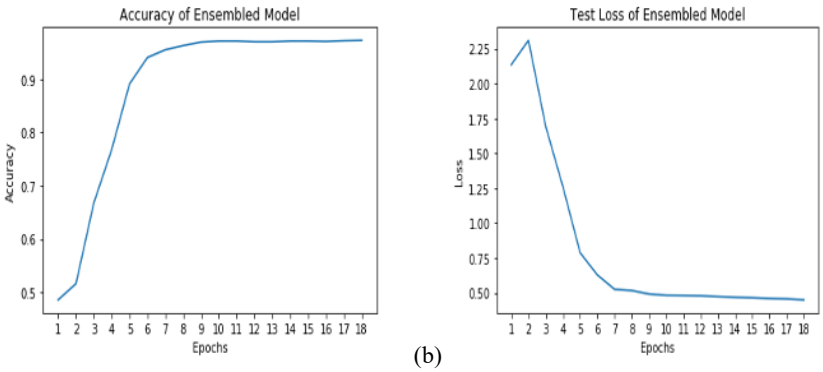


Fig. 5. Visualization of Ensembled Model with InceptionV3 and ResNet101



(a) (b)

Fig. 6. Graph representing change in accuracy (a) and loss (b) with each epoch

5 Conclusion

We conclude that the Ensembled model of Inception V3 and ResNet101 produces an accuracy of 97.24 percent. As observed, the novel technique allows superior output compared to their respective baselines of 96.78% and 96.64%. Because CNN serves as a foundation for feeding characteristics into LSTM, this increase in CNN accuracy will also aid other researchers in improving detection and prediction accuracy in real-world scenarios. This may be extended to the word-level classification of sign language. More data, however, is necessary for testing. In the future, testing data will be generated to compare it to real-world circumstances and noise. Additionally, new datasets from different languages will be evaluated to generalize the design.

References

1. Li, Dongxu, et al. "Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison." Proceedings of the IEEE/CVF winter conference on applications of computer vision. 2020.
2. Dong, Cao, Ming C. Leu, and Zhaozheng Yin. "American sign language alphabet recognition using microsoft kinect." Proceedings of the IEEE conference on computer vision and pattern recognition workshops. 2015.
3. Ko, Sang-Ki, et al. "Neural sign language translation based on human keypoint estimation." Applied Sciences 9.13 (2019): 2683.
4. De Coster, Mathieu, Mieke Van Herreweghe, and Joni Dambre. "Sign language recognition with transformer networks." 12th International Conference on Language Resources and Evaluation. European Language Resources Association (ELRA), 2020.
5. Cao, Zhe, et al. "Realtime multi-person 2d pose estimation using part affinity fields." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
6. Pigou, Lionel, Mieke Van Herreweghe, and Joni Dambre. "Sign classification in sign language corpora with deep neural networks." International Conference on Language Resources and Evaluation (LREC), Workshop, Proceedings. 2016.
7. Koller, Oscar, Sepehr Zargaran, and Hermann Ney. "Re-sign: Re-aligned end-to-end sequence modelling with deep recurrent CNN-HMMs." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.
8. Hossen, M. A., et al. "Bengali sign language recognition using deep convolutional neural networks." 2018 Joint 7th International Conference on Informatics, Electronics & Vision (ICIEV) and 2018 2nd International Conference on Imaging, Vision & Pattern Recognition (icIVPR). IEEE, 2018.
9. Rafi, Abdul Muntakim, et al. "Image-based Bengali Sign Language Alphabet Recognition for Deaf and Dumb Community." 2019 IEEE Global Humanitarian Technology Conference (GHTC). IEEE, 2019.
10. Hasan, Md Mehedi, Azmain Yakin Srizon, and Md Al Mehedi Hasan. "Classification of Bengali Sign Language Characters by Applying a Novel Deep Convolutional Neural Network." 2020 IEEE Region 10 Symposium (TENSYMP). IEEE, 2020.
11. Chollet, F., & others. (2015). Keras. GitHub. Retrieved from https://github.com/fchollet/keras
12. Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

13. He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
14. Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016
15. He, Kaiming, et al. "Identity mappings in deep residual networks." European conference on computer vision. Springer, Cham, 2016.
16. Chollet, François. "Xception: Deep learning with depthwise separable convolutions." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
17. Huang, Gao, et al. "Densely connected convolutional networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
18. Szegedy, Christian, et al. "Inception-v4, inception-resnet and the impact of residual connections on learning." Thirty-first AAAI conference on artificial intelligence. 2017.
19. Zoph, Barret, et al. "Learning transferable architectures for scalable image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.
20. B. Zoph and Q. V. Le. Neural architecture search with reinforcement learning. In International Conference on Learning Representations, 2017.
21. Rossum, G and F L. Drake. "Python 3 Reference Manual." (2009).
22. Bisong, Ekaba. Building machine learning and deep learning models on Google cloud platform. Berkeley, CA, USA: Apress, 2019.
23. Deng, J, W Dong, R Socher, K Li and L Fei-Fei. "Imagenet: A large-scale hierarchical image database." (2009).