

Alphabet Sign Language Image Classification Using Deep Learning

Rangel Daroya^{1,3}, Daryl Peralta^{1,4}, and Prospero Naval, Jr.²

¹Electrical and Electronics Engineering Institute, University of the Philippines

²Department of Computer Science, University of the Philippines

³rangel.daroya@eee.upd.edu.ph

⁴daryl.peralta@eee.upd.edu.ph

Abstract—Sign language is very important for people who have impaired hearing and speaking inabilities. In this work, we present a method to classify RGB images of static letter hand poses in Sign Language using a Convolutional Neural Network (CNN) inspired by Densely Connected Convolutional Neural Networks (DenseNet). It was further implemented to classify sign languages in real time using a web camera. DenseNet has been widely used for classification tasks due to the advantages it introduces such as alleviating the vanishing gradient - a common problem encountered with deep networks. Since a deep network is proposed to be used for our sign language classification task, this characteristic is useful. Our proposed network was able to achieve an accuracy of 90.3 % which is comparable to other works including those that used depth images in addition to RGB images. Our network was also able to achieve prediction rates of 50 to 100 Hz which makes it capable of real-time prediction.

Index Terms—deep learning, sign language recognition, neural networks, DenseNet, CNN

I. INTRODUCTION

Sign languages provide a way to communicate without the use of voice. This is very important for individuals who have impaired hearing and inability speaking because it provides them an alternative means of communicating with other people. Furthermore, sign language is a universal language that can allow people of different speaking language to communicate which is very important in multicultural settings and times of disaster.

Understanding sign language is challenging since it requires memorizing a lot of hand poses and gestures. Some signs even have very similar appearance. This suggests a need for an automatic sign language recognition system allowing anyone to understand sign languages.

In this work, we aim to address the need for a robust and real-time alphabet sign language recognition system by using deep learning which has shown excellent performance in image classification tasks. Since cameras are almost everywhere nowadays in smartphones and laptops, we designed our network to use only images without depth. This also allows for easy expansion in other applications in the future, since there would be no need for specialized sensors or other hardware. We limit our work on static hand gestures excluding j and z from the classification. We used the American Sign Language (ASL) [1] as reference for the hand poses.

II. RELATED WORK

Different approaches have been proposed in trying to solve the problem of recognizing sign language hand gestures. Some of the early works include [2] which used SVM to classify South African Sign Language (SASL) and [3] which used parallel hidden markov models.

In [4], Sole et al. used Extreme Learning Machine (ELM) to learn to classify static hand gestures on the letters of the Auslan dictionary. Although the research showed good preliminary results, the data used to test the network was restricted to images collected from the same day, thus lacking generalization.

In [5], Kim et al. demonstrated a way to utilize a deep neural network (DNN) to classify frames in image sequences of finger spelled letters. Their work was signer-independent meaning capable of recognizing hand poses of any user. They used Histogram of Gradients (HoG) image features [6] as input to the deep neural networks. Similar to [5], Kim et al. [7] also used DNN with HoG features. Both [5] and [7] got their best result using segmented CRF with DNN. Our work was also able to create a signer-independent classification of images but in contrast to these works our classifier is completely deep neural network including the feature extraction.

The work of Pugeault et al. [8] presented the use of depth images from a Microsoft Kinect device. They used multi-class random forest classification and tested their method by varying the input from image only, depth only and combined image with depth. They achieved their best result when depth is combined with image. Their system is also fast enough for real-time classification. In [9], Kang et al. also used depth images as its input without the color image. They also used a DNN for their classifier and was also able to achieve real-time rates. Similar to them, our goal also is to achieve real-time rates but using only color image since only few people have access to depth sensors.

In recent years convolutional neural networks had great success in computer vision tasks including image recognition. Starting with AlexNet [10] which popularized deep convolutional neural networks by winning the ImageNet Challenge: ILSVRC 2012 [11]. Following AlexNet, new network architectures such as [12], [13] and [14] also made significant contributions and improved performance. This work aims to utilize the advances in deep learning techniques

and apply them to create a robust and real-time finger spelled sign language classifier.

In a more recent work of [15], a new model architecture Dense Convolutional Network (DenseNet) was proposed. DenseNet presented a way to solve the vanishing gradient problem of deep networks by connecting each layer to every other layer in a feed-forward fashion. DenseNet exploits the potential of the network through feature reuse which results to less number of parameters. With these advantages, our network model was based on DenseNet.

III. METHODOLOGY

A. Dataset

The dataset used was based on the American Finger Spelling format. Figure 1¹ shows the followed format.

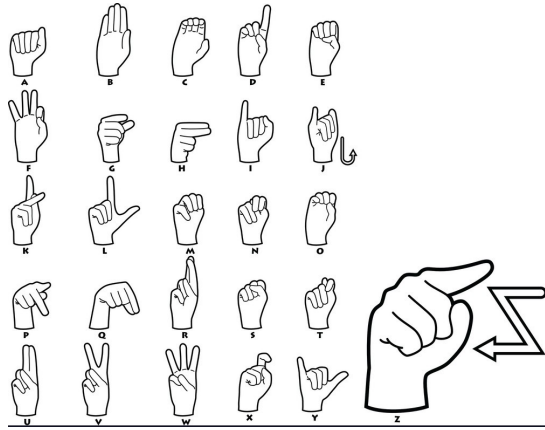


Fig. 1. Hand poses used as a basis in constructing the dataset for each letter.

The dataset used in training the network was composed of static hand poses of letters form the alphabet. This meant excluding the letters j and z due to the motion required to represent them. The work of [8] provided a set of images containing more than 50,000 images of letters formed by the hands. Five sessions were conducted to collect five different sets of letters from different people. They conducted their data collection under similar lighting and background conditions. Figure 2 shows some samples.

Although the dataset discussed contained many images for training, it was still insufficient to achieve the accuracy and robustness desired for this work. Its data collection was restricted to specific lighting conditions and backgrounds. As a result, we acquired a new set of images in more varied lighting conditions and using more variations in the hand letters. Some images were taken such that the hand does not necessarily cover the whole image, and others show the hand being tilted in different ways to take into account these variations. The background chosen for the images were also changed for the duration of the data collection. Figure 3 shows some samples.

¹<https://www.vectorstock.com/royalty-free-vector/sign-language-alphabet-vector-57292>

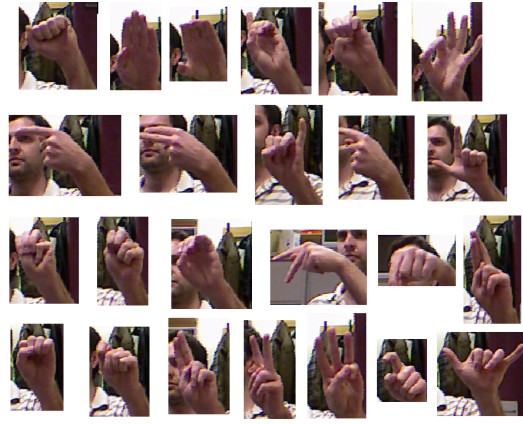


Fig. 2. Samples from the dataset of [8].

B. Data Augmentation

We used data augmentation to further improve the variety in our dataset. Our data augmentation includes random rotations ranging 5 degrees, random horizontal and vertical shifting, random shearing and random zooming. This was implemented through the use of Keras Image preprocessing module.



Fig. 3. Samples from our collected dataset. The coverage of the hand per image was varied, including the hand tilt.

Python was used to implement the data collection through a laptop camera with an image size of 120 pixels by 160 pixels. For every finger spelled letter, we collected around 2,000 images, thereby producing almost an additional 50,000 images for training. For consistency of the data being used for training and testing, the orientation of the hands with respect to the camera for every corresponding letter was based on the dataset of [8]. Some images also included faces of the people implementing the hand poses to improve the

robustness of the network by introducing some noise to the data.

C. Network Architecture

Given an image of a letter hand pose at test time, our goal is to construct a network that can properly classify the image to its corresponding letter using a deep neural network. In order to achieve this, we used the DenseNet architecture from [15]. Figure 4 shows an implementation of a DenseNet block.

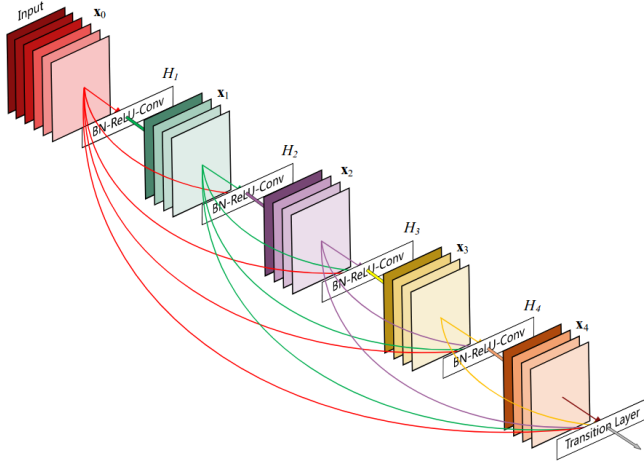


Fig. 4. Structure of a dense block with 5 layers. [15]

This was used to alleviate the problem of vanishing gradients despite using a deep network architecture. Furthermore, it encourages feature reuse across layers, while minimizing the number of parameters. The latter characteristic helps in applying the network for real-time prediction using a web camera. Fewer parameters would result to less training and prediction time, a desirable quality for the purposes of this work.

In the implementation of the network, four such dense blocks in Figure 4 were used, in addition to three transition layers. Each dense block consisted of applying ten layers of batch normalization, relu activation, and a 3x3 convolutional layer. Transition layers on the other hand, use batch normalization, relu, a 1x1 convolutional layer, and a 2x2 average pooling layer. The transition layers are used to effectively reduce the feature map size after every dense block. Figure 5 shows the summary of the network used.

The output of the network is a 24-dimension vector, corresponding to each letter, with each probability between 0 and 1. To properly train the network, categorical cross entropy loss was utilized to make sure that the letter predicted with the highest probability corresponds to the ground truth. This loss was used to measure the performance of the classification model. This is given by Equation 1 where \hat{y}_i is the ground truth and y_i is the predicted probability.

$$L(\hat{y}, y) = - \sum_i \hat{y}_i \log(y_i) \quad (1)$$

TABLE I

ACCURACY AND THE LOSS OBTAINED FROM THE IMAGES OF THE TEST SET.

	Accuracy (%)	Loss
Ours	90.3	0.909

This loss has a minimum value of 0, which is the ideal training and test loss.

D. Implementation Details

The dataset was split to allocate 90% of the data to training and 10% to testing. The data used for testing came from [8], specifically the set of images coming from a person that was not seen throughout training. This was done to measure how well the network was generalizing. We used Keras to implement and train the network with GeForce GTX 1080 Ti. Adam was used as the optimizer with learning rate of 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e-08$, and decay of 0.0. The network was trained for 300 epochs with a mini-batch size of 128. We resize the images to 32 x 32 x 3 for training and testing.

IV. RESULTS AND DISCUSSION

The network was tested on 12,792 images that it did not see during training. The accuracy of the classification was determined as well as the test loss calculated. Table I shows these results.

Aside from the accuracy and loss, a confusion matrix was also obtained to properly contrast the predicted letter and the ground truth. Each entry represents the frequency of the predicted letter in comparison to its ground truth. This is illustrated in Figure 6. The y-axis represents the ground truth, while the x-axis represents the predicted letter.

From the confusion matrix, although there were some lapses in certain letters, it was determined that majority of the predictions are correct. Some letters had difficulty in prediction, particularly due to the similarity in the appearances of the letters. For example, the letter E was confused with letters O and S since the fingers are positioned in the same way with only a slight difference in the way the fingers are clenched, and the position of the thumb. The letter I was mostly predicted correctly, but it is sometimes confused with Y since the only difference between the two is whether or not the thumb is protruding. The same can be said about predicting the letter X, which is sometimes confused with D and S. D and X are similar since the representation for both require the pointer finger to be raised. S and X, on the other hand, both have a mostly clenched hand. However, the network was still mostly able to predict X correctly.

Another letter the network had a difficult time predicting is letter Q. This letter is confused with letters A, N, O, and Y. For letters A and N being confused with Q might have to do with how the fingers from the index to the tiniest finger are tucked in, similar to Q. Both Q and O, on the other hand have a loop formed by the fingers.

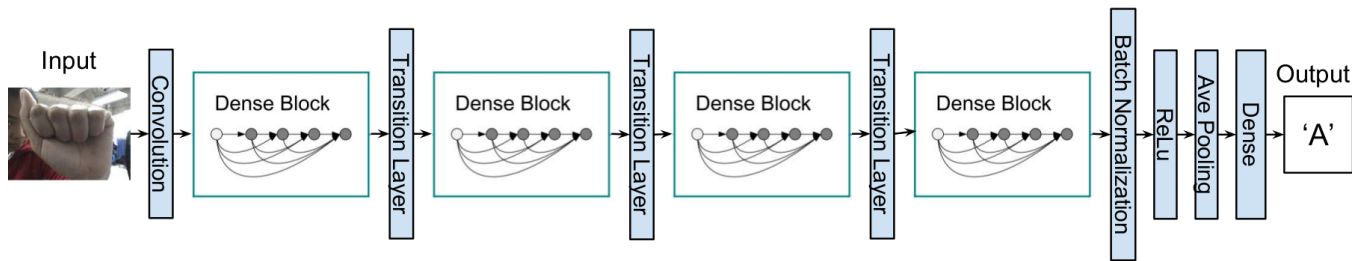


Fig. 5. Network architecture used for classifying the images as letters. Four Dense Blocks [15] were used together with three Transition Layers. A Dense layer was used to get the output with a softmax activation to predict the letter shown in the image input.

	A	B	C	D	E	F	G	H	I	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
A	472	0	0	0	14	0	0	0	0	0	0	0	6	0	0	0	0	0	49	0	0	0	4	0
B	0	534	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C	0	0	540	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	0	544	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	37	0	0	0	11	0	0	0	0	106	0	0	0	385	0	0	0	0	0	0
F	0	7	0	0	0	504	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0
G	0	0	0	0	0	0	506	10	0	13	0	0	0	0	1	0	7	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	529	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
I	2	0	0	0	0	0	0	0	473	0	0	0	0	0	0	0	0	0	1	0	0	0	0	49
K	0	0	0	0	0	5	0	0	0	536	0	0	0	0	0	0	6	0	0	0	22	0	0	0
L	2	0	0	0	0	0	0	0	0	0	512	0	0	0	0	0	0	0	0	0	0	0	0	0
M	0	0	0	0	0	0	0	0	0	0	0	503	19	3	0	0	0	0	0	0	0	0	0	0
N	2	0	0	0	0	0	0	0	0	0	0	2	516	0	0	0	0	10	0	0	0	0	0	0
O	0	0	1	0	0	0	0	0	1	0	0	0	0	529	0	0	0	0	0	1	0	0	0	0
P	0	0	0	0	1	0	10	0	0	0	0	0	1	5	507	3	0	0	0	0	0	0	10	0
Q	60	2	9	0	1	0	1	0	0	0	0	0	85	46	3	242	0	0	6	0	0	0	11	63
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	561	0	0	0	1	0	0	0
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	526	0	0	0	0	0	0
T	2	0	0	0	66	0	0	0	0	0	0	0	26	0	0	0	0	0	437	0	0	0	0	0
U	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	529	0	0	0	0
V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	527	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	513	0	0
X	0	0	0	21	0	0	0	0	1	0	1	0	3	0	0	1	2	35	1	0	2	0	455	0
Y	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	518

Fig. 6. Confusion matrix showing the frequency of the predicted letters in contrast to the ground truth, where highlighted numbers indicate the higher frequencies. The x-axis represents the predicted letters while the y-axis represents the ground truth.

Despite having these errors, the network performed at an accuracy that is almost at par with existing systems. Table II shows the performance of our network in comparison to others.

Methods using depth images had better accuracies compared to methods using only color images. Our network achieved comparable results despite using only color images. Furthermore, the difference in the testing accuracy can also come from the variation in the test and training dataset, since the networks used different sources of data.



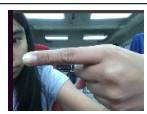
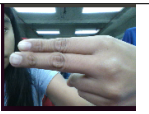


TABLE II
PERFORMANCE COMPARISON OF OUR NETWORK IN HAND POSE LETTER CLASSIFICATION TO OTHERS.

Author	Method	Input Data	Accuracy (%)
Sole et al. [4]	Neural Network	RGB	95
Kim et al. [16]	Semi-Markov	RGB	88.4
Kim et al. [5]	HoG + DNN	RGB	82.7
Kim et al. [7]	HoG + DNN	RGB	92.4
Pugeault et al. [8]	Random Forest	RGBD	75
Kang et al. [9]	DNN	Depth	99.9
Ours	DNN	RGB	90.3

V. REAL-TIME EVALUATION

The network was also tested on its real-time predictions on more varied environment. This proved the ability of the network to generalize on data it has not seen before using hands of people it has not yet encountered. It was easier to test the accuracy and reliability of the network with this method due to the ease at which the environmental factors can be changed. Table III shows some results using a laptop camera.

TABLE III
PREDICTIONS OF THE NETWORK FROM REAL-TIME DATA ACQUIRED
WITH A WEB CAMERA.

Image	Prediction	Image	Prediction
	A		B
	G		H
	D		Y

The network was able to produce an output in the range of 20 to 10 ms or 50 to 100 Hz in a laptop with NVIDIA GTX 1060. This speed is enough for real-time prediction. This was achieved by trying to target a network architecture with a small number of parameters. We were able to achieve a network with 837,444 trainable parameters.

VI. CONCLUSIONS

We presented an approach to real-time alphabet sign language image classification using deep learning. Our proposed deep learning model based on DenseNet was able to perform with 90.3 % accuracy on our test dataset. Our work also included data gathering using a web camera to increase the dataset size to more than 100,000 RGB images making our prediction more robust. Our network was also capable of real-time prediction using image frames from a web camera with rates of 50 to 100 Hz.

Future study may extend our work to accept video frames to include letters j and z in the classification so that more varied inputs can be processed and understood by the network. Furthermore, there is a need for a large public dataset for automatic sign language recognition to utilize new deep learning techniques and a better way to benchmark performance. There are currently very few common sets of data being used, and most of the time researches being conducted use a subset of this. This makes it more difficult to compare different works - a dataset might have a different effect on a network compared to another.

REFERENCES

- [1] C. Padden and D. C. Gunsauls, "How the alphabet came to be used in a sign language," *Sign Language Studies*, vol. 4, no. 1, pp. 10–33, 2003.
- [2] S. Naidoo, C. Omlin, and M. Glaser, "Vision-based static hand gesture recognition using support vector machines," *University of Western Cape, Bellville*, 1998.
- [3] C. Vogler and D. Metaxas, "Parallel hidden markov models for american sign language recognition," in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 1. IEEE, 1999, pp. 116–122.
- [4] M. M. Sole and M. Tsoeu, "Sign language recognition using the extreme learning machine," in *AFRICON, 2011*. IEEE, 2011, pp. 1–6.
- [5] T. Kim, W. Wang, H. Tang, and K. Livescu, "Signer-independent fingerspelling recognition with deep neural network adaptation," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 6160–6164.
- [6] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.
- [7] T. Kim, J. Keane, W. Wang, H. Tang, J. Riggall, G. Shakhnarovich, D. Brentari, and K. Livescu, "Lexicon-free fingerspelling recognition from video: Data, models, and signer adaptation," *Computer Speech & Language*, vol. 46, pp. 209–232, 2017.
- [8] N. Pugeault and R. Bowden, "Spelling it out: Real-time asl fingerspelling recognition," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1114–1119.
- [9] B. Kang, S. Tripathi, and T. Q. Nguyen, "Real-time sign language fingerspelling recognition using convolutional neural networks from depth map," in *Pattern Recognition (ACPR), 2015 3rd IAPR Asian Conference on*. IEEE, 2015, pp. 136–140.
- [10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
- [11] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.
- [12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
- [13] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.
- [14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [15] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, vol. 1, no. 2, 2017, p. 3.
- [16] T. Kim, G. Shakhnarovich, and K. Livescu, "Fingerspelling recognition with semi-markov conditional random fields," in *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1521–1528.