Real-time Sign Language Recognition Using a Convolutional Neural Network

Mayur Ranchod
School of Computer Science and Applied Mathematics
University of the Witwatersrand
Johannesburg, South Africa
1601745@students.wits.ac.za

***Abstract*—There exists a considerable fraction of the global population which constitutes of people with speech and hearing impairments that encounter a communication barrier when attempting to converse with the hearing population. The people that experience this difficulty, also known as deaf-mute people, use sign language as a communication medium to express themselves in a fast and convenient manner which has proven to be particularly effective for face-to-face interactions. Since people of the general public are unfamiliar with sign language, using sign language comes with the limitation of only being effective in the presence of an interpreter to translate between sign language and a familiar spoken language. This limitation provides the motivation for this report i.e., we attempt to eliminate the dependence on an interpreter by harnessing the power of machine learning to enable deaf-mute people to communicate with the hearing population in real-time using a convolutional neural network. The solution that we present is trained to recognize American sign language since it is the most common form of sign language in addition to coinciding with the South African sign language. We acknowledge that numerous solutions have been proposed to address this problem, however, we attempt to develop a general-purpose solution since the applicability of these previous approaches is highly limited. An analysis of our results indicates that our solution is a vast improvement compared to previous approaches following from its ability to accurately detect the majority of the hand gestures whilst achieving a superior recognition time without the need for special equipment.**

***Index Terms*—sign language, convolutional neural network**

I. Introduction

Speech and hearing impairment are communication disorders which may arise as a result of a genetic disease or neurological complications and makes interaction within a society an intimidating task. More specifically, people with speech impairment experience difficulty when articulating sounds to form words while people with hearing impairment find it challenging or impossible to discern sounds within their environment. These impairments form a communication barrier for speech and hearing-impaired people (also known as deaf-mute people) to effectively communicate with the hearing population and vice versa. Over the past few decades, there has been a myriad of solutions introduced in an attempt to eliminate this communication barrier. At first, one may be convinced that communication via a written medium would be a feasible solution to interact in, however, this approach is not as effective as anticipated since deaf-mute people still experience difficulties expressing themselves in a spoken language. In addition, this approach is not ideal for face-to-face interaction or for deaf-mute people to express themselves in a fast and convenient manner.

The most common form of communication used by deaf-mute people today is through the expression of a sequence of hand gestures called *sign language*. More concretely, sign language is expressed through *finger-spelling* which is based on some form of sign language (American, Indian, Australian, etc) to depict the alphabets of the language using simple hand gestures. Depending on the form of sign language, the hand gestures may either be static or dynamic (which require motion to depict the alphabet such as j and z in the American Sign Language (ASL)). Fig. 1 illustrates the hand gestures used to describe the ASL.

Despite sign language being an effective method of communication, the effectiveness of this approach when communicating with the hearing population is still dependent on the presence of an interpreter to translate between sign language and a spoken language. Since most deaf-mute people do not have constant access to an interpreter, communication with the hearing population becomes a challenge in their absence since the majority of people are unfamiliar with sign language. Many solutions have been introduced which attempt to resolve this issue and to allow deaf-mute people to easily communicate with the hearing population by providing a means of translating the finger-spelling gestures to a more familiar language.

Despite the existence of numerous solutions which attempt to facilitate communication between deaf-mute people and the hearing population, these solutions are not suitable for general-purpose use as a result of the poor generality, robustness, and recognition time that these solutions achieve. These solutions range from simple statistical approaches to more sophisticated approaches which primarily rely on machine-learning techniques such as Support Vector Machine (SVM), K-Nearest Neighbours (KNN) and Convolutional Neural Networks (CNNs). We note that even though these solutions may have been designed to recognize a particular form of sign language, we are instead interested in the learning process followed as the language of choice can easily be substituted given sufficient data. Below, we provide a brief discussion on the limitations of these solutions which prevents them from

Fig. 1. Hand gestures corresponding to ASL. [1]

being general-purpose solutions.

One of the key aspects which makes the sign language recognition problem particularly challenging is the large number of possible gestures as well as the similarity between some of these gestures. Following from the complexity associated with detecting these hand gestures, many of the introduced solutions are unable to detect all hand gestures correctly and are, therefore, restricted to only recognizing a subset of the full set of hand gestures. In particular, some solutions [2] are restricted to detecting as little as 5 of the 24 static gestures while many solutions [3] [4] are restricted to the first half the alphabet (*a-k*). As a result of the limitations imposed on these solutions, it is evident that these approaches are not suitable for general-purpose use since a deaf-mute person would be unable to effectively communicate with less than half of the alphabet.

Another important consideration when reviewing the previous solutions is the associated recognition time. Since we attempt to address a communication problem, the recognition should be performed in real-time (with negligible latency) to allow for effective communication. Upon analysing the previous solutions, we see that the majority of solutions are unable to achieve an acceptable recognition time as most previous solutions experience a latency ranging between 1-2 seconds for a single recognition which is not conducive to effective communication.

Since the sign-language recognition problem can be viewed as constituting of 2 tasks, namely, locating the hand and recognizing the hand gesture, some approaches have relied on special apparatus to improve the effectiveness of the solution. In particular, some of the earliest solutions required a colour glove which made it easier to locate the hand. Most recent approaches [5] [6] [7] rely on using a depth camera such as a Microsoft Kinect. These solutions achieve a high degree of robustness since the OpenNI framework and the depth information provided by the camera assists with the localization and recognition of the hand gesture with the added benefit of being illumination invariant. Despite these solutions being able to achieve promising results, the specialized equipment required by these solutions do not make them suitable for general-purpose use as the required equipment is not easily accessible.

In response to the limitations experienced by these previous solutions, we attempted to develop a solution that can be used for general purpose use and easily accessible to anyone. We defined the goal for our solution to be able to accurately recognize the majority of hand gestures whilst achieving an unnoticeable degree of latency and not requiring any specialized equipment. To achieve this, we used a Convolutional Neural Network (CNN) that is trained on the MNIST sign-language dataset [1] which contains only static gestures (*a-y* excluding *j*) of the American Sign Language (ASL) which coincides with the South African sign language and therefore, we restricted our system to these static gestures. This did not significantly reduce the generality of the model since none of the approaches that we reviewed were able to detect all 26 hand gestures. The input to our CNN is an ordinary webcam stream which in turn predicts the alphabet depicted by the hand-gesture from the webcam.

To facilitate our solution, we implemented a background removal algorithm to reduce the effects of background elements in an attempt to improve the model's performance. This is algorithm consists of a series of image-processing techniques which we discuss in further detail in Section 4.

We summarize the contribution of the work presented in this report as follows:

- Constructing a CNN in an attempt to develop an effective general-purpose solution for real-time sign language recognition.
- Implementing a background removal algorithm in an attempt to improve the performance of our model.

The structure of this report is such that we review some of the previous solutions to the sign-language recognition problem as well as highlighting some of their key aspects in Section 2. This is then followed by an in-depth description of the dataset used which is presented in Section 3. Section 4 formally describes the method pertaining to our solution which is followed by an analysis of the results presented in Section 5. This report is concluded by Section 6 which provides a short summary that highlights the salient points presented in this report.

II. Related Work

Due to the importance of the sign-language recognition problem, there exists a myriad of solutions which have primarily been tackled using machine-learning techniques. The simplistic approach taken by [2] uses an SVM with Principal Component Analysis (PCA) that is only capable of detecting the gestures corresponding to *b, d, f, l* and *u* with a 99.4% accuracy. In addition to the model only being capable of detecting 5 of the 24 static hand gestures, this approach is not ideal as the number of image-processing operations performed is excessive. The benefit of this approach is that is

one of the few solutions that uses a webcam stream as input. A common approach taken to improve on the results of the SVM approach is to perform KNN classification with PCA. The study by [4] proposes an approach to recognize the first half of the alphabet (*a-k*) by representing each sample by a 48-dimensional vector by extracting 16 histogram bins from each colour channel. By using this full vector representation, an accuracy of 99.8% is achieved while their PCA-reduced representation achieves a mediocre accuracy of 28% which the authors attribute to the presence of highly correlated features in the ASL alphabet which PCA is unable to model/ account for. The study proposed by [8] introduces 2 solutions to the problem, namely, a neural network and a KNN approach. The authors experiment with numerous neural networks by varying the number of hidden nodes whilst using a single hidden layer. With regard to the PCA approach, KNN is performed in the *handspace* where the mean vector is subtracted from each sample to find *eigenhands* such that every sample is represented by a linear combination of the *eigenhands*. Both solutions were not effective as a 40-70% testing accuracy is achieved and further experimentation revealed that the recognition performance was highly dependent on the characteristics of the hand in the image (such hand size, skin tone, etc).

The solution presented by [6] takes a unique approach and uses a multi-class random forest for classification which heavily relies on the depth information provided by a Microsoft Kinect camera to achieve a 75% accuracy while being able to detect the majority of the alphabets. Most modern approaches use a CNN following from its efficacy at image classification tasks and its ability to automate the feature extraction process. The approach taken by [3] uses a pre-trained GoogleNet CNN pre-trained on the ILSVRC2012 dataset [9]. To adapt the network to perform sign-language recognition, the first 3 layers of the network were reweighted and the network was then trained on the Surrey University and Massey University ASL datasets [10] to achieve a 74% accuracy to detect the first half of the alphabets. Despite their solution being considered as 'real-time', the demonstration of their solution[1] illustrates the excessive latency associated with recognition. The approach introduced by [5] is considered as the state of the art by many which is designed to recognize the Italian sign language. Analogous to [6], their solution heavily relies on the depth information provided by a Microsoft Kinect sensor. Since this solution operates on full-body images, their solution consists of 2 CNNs and one Artificial Neural Network (ANN) in which the CNNs are used to extract hand gestures and upper-body gestures respectively while the ANN uses the concatenation of the result produced by the 2 CNNs to perform classification. This solution achieves an accuracy of 95.68% and has the added benefit of detecting dynamic gestures.

The above analysis emphasizes the discussion provided in the introduction section which considers the limitations of previous solutions for being used for general-purpose use. More specifically, we discovered that many approaches [2] [3] [4] detect less than half the alphabet while many solutions [5] [6] [7] rely on a Microsoft Kinect camera to extract depth information. This provided sufficient motivation for the need to develop a solution that achieves a good balance between and recognition-time without requiring special equipment.

[1]https://youtu.be/79Fmu0TZ9xc

III. Dataset and Features

The form of the dataset that we used is very similar to that of the popular MNIST handwritten digits dataset commonly used to benchmark the performance of machine-learning models. We used the MNIST sign-language dataset [1] which contains 27,455 training samples and 7172 testing samples presented in a CSV format where each sample is represented by a 28×28 vector and each feature represents the intensity of a pixel between $0-255$. Furthermore, we used 4119 samples (15% of the training samples) as validation data. The grayscale images formed are used to describe the static hand gestures of the ASL that are learned by our CNN. The gestures which describe *j* and *z* are excluded from this dataset as these are dynamic gestures that require motion. We note that this exclusion did not reduce the generalization of our solution as most previous approaches were unable to account for these gestures as well. We provide a handful of examples of the images described by the dataset in Fig. 2.

Upon experimentation, we deduced that normalization (scaling the image intensities to [0,1]) was not required as this did not have a significant effect on the convergence rate and did not lead to improved results. Since our solution is based on a CNN, no explicit consideration was required for feature extraction since the CNN automates the feature extraction process. The labels for the training and testing samples were converted to its one-hot encoded representation which is a common practice when performing multi-class classification. Upon analysing the labels of the training and testing data, we noticed that instead of the label depicting the alphabet described by its corresponding feature vector, the label instead depicted the index of the alphabet (e.g. 0 corresponding to a, 1 corresponding to b, etc). To convert the labels to a more convenient representation, we created a dictionary to easily translate the numerical representation to its alphabetical representation.

IV. Method

Since our goal was to develop a real-time sign language recognition solution for general-purpose use, we designed an interface that receives input from a standard computer webcam as opposed to relying on the specialized equipment required by previous approaches. The interface displays the input from the webcam as well as the region of interest (ROI) defined for the user to perform hand gestures. We emphasize that it is the user's responsibility to ensure that all hand gestures are centred and fully enclosed within the defined ROI. By

Fig. 2. Examples of the hand gestures described by the MNIST sign language dataset. [1]

defining a static ROI, this eliminated the task of locating the hand within the webcam input. For maximal performance, we strongly encourage that the background of the user to be clear/plain without any objects visible.

In addition to our standard approach to performing sign language recognition, we implemented a background removal algorithm using OpenCV with the expectation of achieving improved results by only focusing on the hand gesture. To achieve this, we implemented a skin detection algorithm adapted from [11] which tries to first detect the skin tone of the user under the current lighting conditions. To accomplish this, the user is first required to place their hand within the ROI whilst ensuring that the 9 green squares are fully covered by their hand. The green squares are used to detect the intensity/ skin tone at those regions. Upon completion, the intensities are converted to the HSV colourspace which is less sensitive to lighting changes than the RGB colourspace. We then compute the histogram of the intensities at those points which we then normalize to the range [0,255]. This produces the histogram which corresponds to the skin tone of the user which is used to remove the background of the user's hand. This is achieved by performing a backprojection operation which locates the regions within the ROI that corresponds with the computed skin-tone histogram. Lastly, the result is thresholded using a threshold value of 150 and we use the result to perform a bitwise 'and' operation with the original input to produce the final result. When performed correctly, the background of the hand gesture is removed as illustrated in Fig. 3. The interface that we developed allows the user to select whether background removal should be performed or not and to test the effectiveness of the above procedure.

In order to classify the hand gesture within the ROI, we constructed a Convolutional Neural Network (CNN) using the Keras framework (with a TensorFlow backend) since CNNs have been accepted as the de-facto standard for image-related tasks for numerous reasons. In particular, its unique ability to automatically extract features without human supervision leads

Fig. 3. An illustration of the result produced by the background removal algorithm.

to superior performance whilst still being a computationally efficient solution. Our network consists of 3 convolutional layers, each followed by a max-pooling layer and a dropout layer. The key aspect of a CNN are the convolutional layers that are responsible for feature extraction by filtering the input using a convolutional filter to produce a feature map. The convolution operation can be described by (1) where g is the result of the convolution operation, f represents the input and h represents the convolution filter.

$$g(i,j) = \sum_{k,l} f(i-k, j-l).h(k,l) \tag{1}$$

The max-pooling layers that follow the convolutional layers are used to reduce the complexity of the model by reducing the number of model parameters which also has a regularization effect. This is achieved by applying the max operator to a small subregion of the input. This layer is then followed by a dropout layer that is used to reduce overfitting. Once the feature extraction process is completed, the classification takes place by means of a fully connected layer using the softmax activation function.

Both implementations that we have introduced i.e. when background removal is performed and when background removal is not performed, uses the same aforementioned network to perform hand gesture recognition. A comprehensive description of our network detailing the specific parameters and design choices is provided in the next section along with an evaluation of the results achieved.

V. Experimental Results and discussion

We now present the specific parameters of our network as well as discuss some of the design choices that were considered. As mentioned previously, we constructed a CNN to classify the hand gestures within the ROI. This choice was motivated by the superiority of CNNs in terms of the quality of results and the computational efficiency achieved on image classification tasks. The role performed by the CNN is two-fold, namely, to perform feature extraction and to use these extracted features to classify the given hand gesture. The first part of our network is responsible for feature extraction which consists of 3 convolutional layers, each using the ReLU activation function and followed by a max-pooling layer and dropout layer. The number of hidden layers was determined by systematic experimentation and

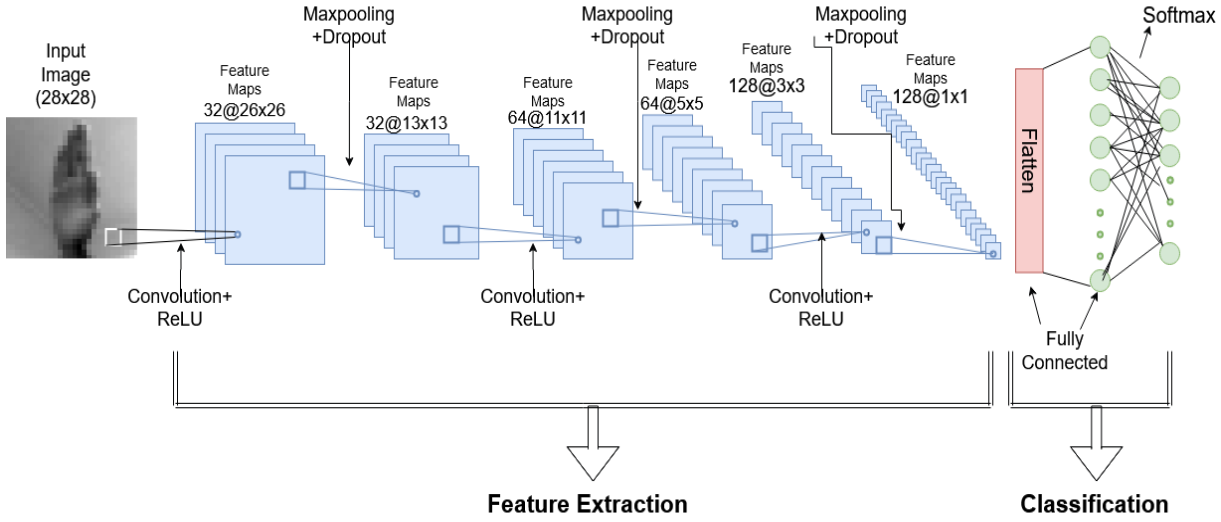

Fig. 4. The architecture of our CNN model.

by considering the complexity of the problem where too few layers may take an idealistic approach which would result in poor performance while using more layers would be unnecessary and could potentially lead to over-fitting. With regard to the number of filters used at each layer, as we go deeper into the network, the number of filters increases while the size decreases since the receptive field gets larger as we go deeper into the network which corresponds to modelling more complex features. More specifically, the number of filters used in each convolutional layer is 32, 64 and 128 respectively. Each convolutional layer is followed by a max-pooling layer with a pooling size of (2,2) which reduces the number of parameters and complexity of the model. This is then followed by a dropout layer used to prevent overfitting by randomly selecting nodes to be dropped-out with a specified probability (we use 0.25) in each weight update cycle. Once the features have been extracted, the extracted features are used to perform the classification. The network is first flattened, and we form a fully connected layer (using 2 Dense layers with 512 and 26 nodes respectively). The fully-connected layer uses the softmax activation function which results in a vector that represents the probability distributions of a list of potential outcomes. We illustrate the architecture in Fig. 4.

Since we were performing multiclass classification, the network used the categorical cross-entropy loss. To optimize the model, we contrasted the accuracies achieved by stochastic gradient descent (SGD) and Adam [12] when the learning rate and decay rates (in the case of Adam) were varied. Upon completion of the experiments, we deduced that using Adam with a learning rate of 0.001 and decay rate of (β_1=0.9, β_2= 0.999) produced the best results. We trained the model for 27 epochs with a batch size of 256 and determined these parameters by means of systematic experimentation. Once we trained the model, we began to evaluate the results achieved.

In Fig. 5, we illustrate how the model's accuracy increases on the training and validation data as the epochs increases and in Fig. 6, we show the decrease in the model's loss on the training and validation data as the epochs increases. Upon reviewing the accuracy of the model, we see that there is a considerable improvement from the model's initial state to the 5th epoch since an 80% accuracy is achieved within this short interval. We see that once the 10th epoch is surpassed, the rate of increase in accuracy begins to deteriorate. As we approach the 17-27th epochs, it may seem that no improvement is being made, however, closer investigation reveals that there is still a minuscule increase in the accuracy achieved. Upon reviewing the progression of the model's loss as the epochs increases, we see that the model's loss gradually decreases as the epochs increases. Fig 5 and Fig. 6 illustrates similar trends as those produced by [13]. For experimentation purposes, since it appears that no significant progress is made beyond the 10th epoch, we attempted to test the model when trained for 10 epochs while keeping the batch size fixed. This experiment revealed that even though it may appear that the model's loss and accuracy have reached convergence by the 10th epoch, our real-time user-testing revealed that this had a significant negative impact on the recognition accuracy of the model.

In order to evaluate the performance of our model, we compute a series of performance metrics such as accuracy, precision, and recall. Our model achieves a training accuracy of 98.7% and a 96.5% testing accuracy. These promising results provide an early indication of the efficacy of the model, subsequently, we determine whether our solution is competitive by comparing the results to those achieved by other CNN-based solutions trained on the MNIST sign language dataset. We note that not many solutions exist that are trained on this dataset since it is still a fairly new dataset. Furthermore, many researchers opt to create their own dataset

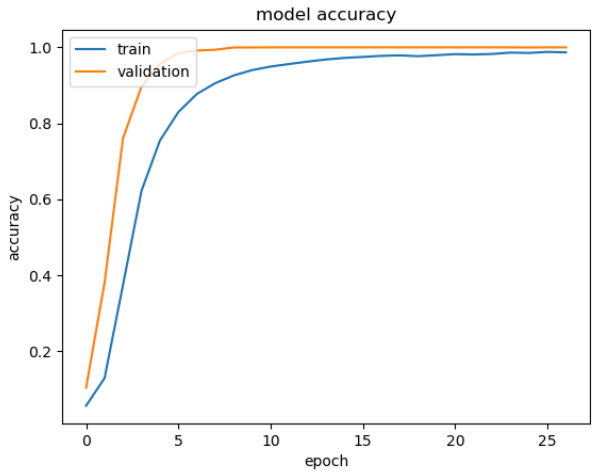

Fig. 5. The progression of the model's accuracy on the training and validation data as the epochs increases.

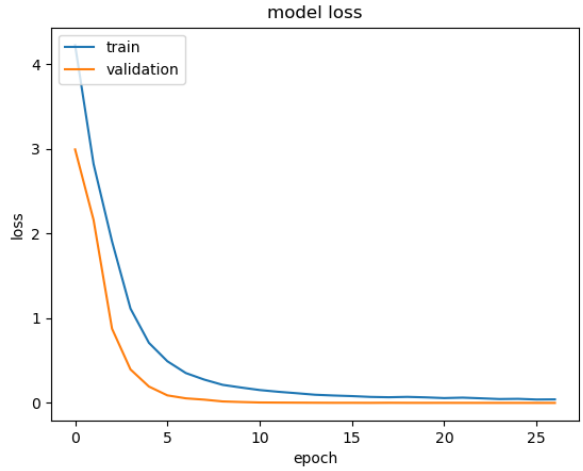

Fig. 6. The progression of the model's loss on training and validation data as the epochs increases.

as collecting data for this problem is a straightforward task. We present this comparison in Table I which indicates that the accuracy achieved by our model is indeed competitive as our model's accuracy exceeds the accuracy achieved by [13], however, the highest accuracy is achieved by [14] with a notable accuracy of 99.7%. Despite the approach by [14] achieving the greatest accuracy from all solutions that are compared, further analysis reveals that this approach is trained extensively on the data with a batch size of 1000 for 3301 epochs while our approach is able to achieve similar results while only training the model for 27 epochs using a batch size of 256. This observation highlights the efficiency of our solution since acceptable results are achieved whilst being a computationally efficient solution.

Table II presents the precision and recall that our model achieves for each alphabet. The *precision* of a given alphabet,

TABLE I
COMPARISON OF THE ACCURACY ACHIEVED BY CNN MODELS TRAINED ON THE MNIST SIGN LANGUAGE DATASET

Approach	Testing Accuracy Achieved
Ours	96.5%
Zhao [13]	89.32%
Ahuja [14]	99.7%

TABLE II
THE PRECISION AND RECALL ACHIEVED BY OUR MODEL FOR EACH ALPHABET.

Alphabet	Precision	Recall	Alphabet	Precision	Recall
a	1.00	1.00	n	1.00	1.00
b	1.00	1.00	o	1.00	0.93
c	0.95	1.00	p	1.00	1.00
d	1.00	1.00	q	0.92	1.00
e	1.00	1.00	r	0.83	0.72
f	1.00	1.00	s	0.91	1.00
g	0.93	0.78	t	0.80	0.89
h	0.94	0.92	u	0.92	0.99
i	0.93	1.00	v	1.00	0.99
k	1.00	0.94	w	1.00	1.00
l	1.00	1.00	x	0.90	0.92
m	1.00	1.00	y	1.00	0.94

x, is computed as the number of samples that were correctly classified as x over the total number of samples that were classified as x. Similarly, the *recall* for a given alphabet, x, is computed as the number of samples that were correctly classified as x over the total number of actual/ true samples of x. An analysis of Table II reveals that promising results are achieved since our model is very accurate in its classifications. This can be deduced from the observation that a precision and recall of 1.00 is achieved for many of the alphabets. To further understand the cause for which a lower precision or recall is achieved, we review the confusion matrix presented in Fig. 7 which provides a clear representation of the model's classification abilities. In particular, we are able to analyse the model's behaviour in terms of how many samples are classified correctly as well as determine the most common types of errors that are committed. Upon reviewing Fig. 7, we immediately see that the confusion matrix exhibits a prominent main diagonal which indicates that a vast majority of the samples in the testing dataset are classified correctly. Despite the model being able to correctly classify a vast majority of the data, we note that the model commits a handful of common errors. An example of this error is the model mistakenly classifying the gesture for i as y which is committed 35 times. By referring to Fig. 1, we see that the hand gesture for i and y are indeed quite similar since both gestures are composed of an upright fist with the 'pinky' finger raised with the main discriminating factor being that the thumb is protruded to form a y which is not the case for the gesture corresponding to i. We therefore acknowledge that these gestures are closely correlated. Another common confusion exists between the gesture corresponding to o being classified as c 35 times. A review of Fig. 1 reveals that the gestures corresponding to

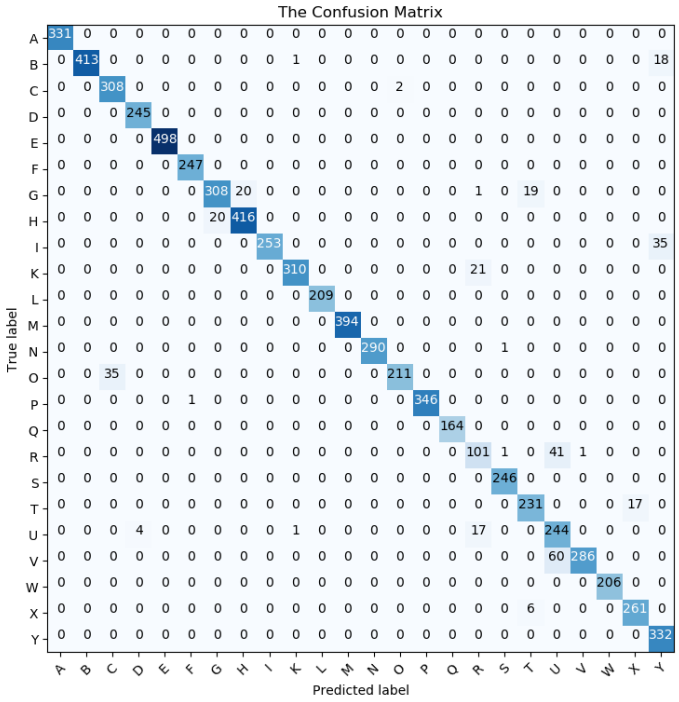

Fig. 7. The confusion matrix representing the results achieved by our model.

these alphabets are closely correlated since the overall hand shape is similar which is only distinguished by whether the index finger is joined with the thumb to form the gesture for o. The most common error committed is that the gesture for v is mistaken for u 60 times. By referring to Fig. 1, we see that the corresponding hand gestures are very similar with the only difference being that the raised fingers are joined to form a u while the fingers are spread to form the gesture for v. Taking the above analysis into account, we see that most of the common errors committed are justified to an extent following from the similarity of the hand gestures in the ASL. By considering the overall performance of our model, we are convinced that our model is a significant improvement compared to the previous models that were restricted to less than half of the alphabet since the overall performance achieved by our model is still promising.

A. Real-time User-testing

In addition to evaluating the model's theoretical performance, we experimented with our designed interface to determine the model's classification performance on real-world samples as well as to determine the associated recognition time. This analysis was used to infer whether our solution can be used as a general-purpose solution or not. To test our model, we first ensured that the background of the defined ROI in which we performed the hand gestures was clear/plain without any objects in sight.

We first considered the solution in which background removal is not performed. Upon experimentation, we deduced that this approach is effective as it is able to easily detect a vast majority of the gestures. Furthermore, the associated recognition time is near-instantaneous which is desirable when addressing a communication problem. This is in contrast to previous approaches which experience a recognition latency in excess of 2 seconds. Despite being able to detect the majority of hand gestures, we discovered that our solution experiences difficulty in detecting the hand gestures corresponding to a, e, m, n, and s. This was also an issue encountered by [6] [7] which both have the advantage of using the depth information provided by a Microsoft Kinect camera. To investigate the reason behind this limitation, we examined Fig. 1 and noticed that the gestures corresponding to these 5 alphabets are indeed very similar to each other. In particular, we see that all 5 gestures consist of an upright fist in which each gesture is primarily differentiated by the position of the thumb. These minor variations in the position of the thumb could not be reflected by our dataset. We attribute this limitation to the lack of variation of the samples in the MNIST sign language dataset. Consequently, we are convinced that using a dataset that contains minor variations in the rotation and viewpoints for the gestures could improve the performance of the model considerably. Despite the aforementioned limitation, we are convinced that our developed solution is a significant improvement to many previous approaches when considering the trade-off between the near-instantaneous recognition time and the accuracy achieved.

Upon analysing the performance of the solution in which we perform background removal, we noticed that this did not lead to an improved accuracy and in fact, led to a significant decline in the accuracy. We attribute this to the fact that the background is replaced with a black background while the images used to train the model have a grey background. We acknowledge that our background removal algorithm did not improve the accuracy of our model, however, its effectiveness at background removal makes it feasible to be used as part of other models which require background removal where the background removed images does not conflict with the training data.

VI. Conclusion

This report presented a solution to performing real-time sign language recognition to enable deaf-mute people to effectively communicate with the hearing population. The limitations disclosed by an analysis of the previous approaches provided the motivation to develop a solution that is suitable for general-purpose use. This was accomplished by developing a CNN to perform real-time sign language recognition without requiring specialized equipment. We saw that our approach is indeed competitive when comparing the results to those produced by other approaches. Furthermore, our real-time user-testing demonstrated the effectiveness of our solution which is able to detect a majority of the hand gestures. We discovered that our model experienced difficulty when detecting some of the hand gestures which we do not solely attribute to the design of our model but rather to the choice of the dataset.

Our background removal algorithm did not lead to improved performance as expected; however, its effectiveness makes it suitable to be used to facilitate other models. As future work, we are convinced that the robustness of the model can be significantly improved by using a different dataset containing more samples that capture a larger variation of the gestures under different lighting conditions and with minor rotations and translations of the hand gestures to find more distinctive features to discriminate between the gestures. Furthermore, it would be desirable to modify the model to account for dynamic gestures without the use of a depth camera.

REFERENCES

[1] tecperson, "Sign language mnist," 2017, https://www.kaggle.com/datamunge/sign-language-mnist, Accessed on: 14 June 2020.

[2] X. Jiang and W. Ahmad, "Hand gesture detection based real-time american sign language letters recognition using support vector machine," in *2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*. IEEE, 2019, pp. 380–385.

[3] B. Garcia and S. A. Viesca, "Real-time american sign language recognition with convolutional neural networks," *Convolutional Neural Networks for Visual Recognition*, vol. 2, pp. 225–232, 2016.

[4] D. Aryanie and Y. Heryadi, "American sign language-based finger-spelling recognition using k-nearest neighbors classifier," in *2015 3rd International Conference on Information and Communication Technology (ICoICT)*. IEEE, 2015, pp. 533–536.

[5] L. Pigou, S. Dieleman, P.-J. Kindermans, and B. Schrauwen, "Sign language recognition using convolutional neural networks," in *European Conference on Computer Vision*. Springer, 2014, pp. 572–578.

[6] N. Pugeault and R. Bowden, "Spelling it out: Real-time asl fingerspelling recognition," in *2011 IEEE International conference on computer vision workshops (ICCV workshops)*. IEEE, 2011, pp. 1114–1119.

[7] A. Kuznetsova, L. Leal-Taixé, and B. Rosenhahn, "Real-time sign language recognition using a consumer depth camera," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2013, pp. 83–90.

[8] J. Atwood, M. Eicholtz, and J. Farrell, "American sign language recognition system," *Artificial Intelligence and Machine Learning for Engineering Design. Dept. of Mechanical Engineering, Carnegie Mellon University*, 2012.

[9] ImageNet, "Large scale visual recognition challenge 2012 (ilsvrc2012)," 2012, http://image-net.org/challenges/LSVRC/2012/index, Accessed on: 14 June 2020.

[10] M. University, "Gesture dataset 2012," 2012, https://www.massey.ac.nz/ albarcza/gesture$_d ataset 2012.html, Accessed on : 14.June 2020.

[11] K. Nikolskaia, N. Ezhova, A. Sinkov, and M. Medvedev, "Skin detection technique based on hsv color model and slic segmentation method," in *Proceedings of the 4th Ural Workshop on Parallel, Distributed, and Cloud Computing for Young Scientists, Ural-PDC*, 2018, pp. 123–135.

[12] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[13] Y. Zhao and L. Wang, "The application of convolution neural networks in sign language recognition," in *2018 Ninth International Conference on Intelligent Control and Information Processing (ICICIP)*. IEEE, 2018, pp. 269–272.

[14] R. Ahuja, D. Jain, D. Sachdeva, A. Garg, and C. Rajput, "Convolutional neural network based american sign language static hand gesture recognition," *International Journal of Ambient Computing and Intelligence (IJACI)*, vol. 10, no. 3, pp. 60–73, 2019.