Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Ensemble Transfer Learning for Hand-sign Digit Image Classification

Andi Muhammad Amil Siddik[1,*], Ainun Mawaddah Abdal[2], Armin Lawi[1], Edy Saputra Rusdi[1]

[1] Information Systems Study Program, Department of Mathematics, Faculty of Mathematics and Natural Sciences, Universitas Hasanuddin, Makassar, Indonesia
[2] Actuarial Study Program, Department of Mathematics, Faculty of Mathematics and Natural Sciences, Universitas Hasanuddin, Makassar, Indonesia

ARTICLE INFO

Article history:
Received 18 September 2023
Received in revised form 8 November 2023
Accepted 6 March 2024
Available online 8 April 2024

Keywords:
Ensemble transfer learning; ResNet-50; VGG-19; image classification

ABSTRACT

Hand sign classification is a challenging task in image processing and machine learning. Robust learning algorithms are essential to achieve optimal performance. Ensemble transfer learning, a technique that combines ensemble learning and transfer learning, is a promising approach to improve classification model performance. This study investigates the use of ensemble transfer learning for hand sign classification. The Sign Language Digits Dataset, which contains ten distinct handwritten image types, was used to evaluate the performance of three architectures: ResNet-50, VGG-19, and Ensemble Transfer Learning (a fusion of ResNet-50 and VGG-19). The results showed that all architectures performed well, but Ensemble Transfer Learning with 2 ResNet-50 and 1 VGG-19 achieved the best performance with an accuracy of 99.03%. This suggests that ensemble transfer learning can effectively enhance model performance in image hand sign classification. The study also found that combining ResNet-50 and VGG-19 in Ensemble Transfer Learning yielded superior results compared to individual models. This is because ensemble transfer learning can leverage the strengths of both models to improve the overall performance. The findings of this study highlight the significance of employing ensemble transfer learning techniques to enhance accuracy and reliability in hand sign image classification.

1. Introduction

Recognizing hand-sign digits holds significant importance for a wide range of applications, including sign language communication and assistive technologies [1]. Traditional methods for hand sign recognition have encountered challenges, primarily due to the intricate nature and diversity of hand signs. These methods heavily rely on manually engineered features and shallow machine learning algorithms [2,3].

As hand signs continue to play an increasingly prominent role in our daily lives, researchers have developed various techniques to enhance their identification. Among these, Convolutional Neural Networks (CNNs) have emerged as one of the most popular and effective approaches for extracting information from images of hands [4-8].

---

[*] *Corresponding author.*
E-mail address: amilsiddik@unhas.ac.id

https://doi.org/10.37934/araset.43.1.95111

While it is a general principle that larger datasets tend to yield better performance [9-11], it is essential to acknowledge that training a model with such datasets demands considerable time and computational resources [12]. In our research, we work with a dataset consisting of 2062 images, which is categorized as relatively small. However, this dataset, drawn from 218 different individuals, offers a crucial advantage by providing the essential diversity needed in the representation of hand signs and sign languages. This diversity encompasses a wide array of styles, expressions, and contextual variations, closely resembling real-world situations. This diversity is instrumental in enabling our classification model to effectively recognize hand signs or sign language that may be encountered in various scenarios, while also mitigating potential biases that can arise when a dataset is excessively focused on one individual or group.

Transfer learning, a technique that leverages pre-trained models on extensive datasets, offers a potent solution for improving the performance of models on smaller datasets. By employing a pre-trained model, our approach allows the model to assimilate features extracted from larger datasets and apply that knowledge to our smaller dataset. This strategy addresses the inherent challenges posed by smaller datasets and bolsters the model's performance. Transfer learning has demonstrated its effectiveness across various domains, including image classification, natural language processing, and speech recognition. Notably, it stands out as a valuable tool for tackling classification problems associated with limited data. In our research, we focus on two renowned pre-trained models, VGG19 and ResNet50, which have been extensively used for transfer learning in image classification tasks [13-18]. These models have exhibited remarkable success on extensive image datasets and are well-suited for extracting vital high-level information from hand-sign digit images [18].

Beyond transfer learning, we also explore the potential of ensemble methods, a popular machine learning approach that combines multiple prediction models to achieve superior predictive solutions compared to individual models. Ensemble methods have proven valuable in a range of machine learning tasks, including image classification [19,20].

In this paper, we aim to create various ensemble method combinations using VGG19 and ResNet50 for the classification of hand-sign digit images, with a particular focus on comparing their performance. We will investigate different ensemble methods, such as stacking and bagging, and explore various hyperparameters, including the number of layers. Our research will also involve a comparative analysis, evaluating the performance of ensemble methods against that of individual models, such as VGG19 and ResNet50. The findings of our study promise to offer valuable insights into determining the optimal combination of ensemble methods and hyperparameters for achieving superior transfer learning in hand-sign digit image classification using VGG19 and ResNet50.

2. Methodology

2.1 Data Collection and Preprocessing

2.1.1 Data collection

We using the American Sign Language (ASL) Digits Dataset, comprising images of individuals' hands forming numerals from 0 to 9. This hand-sign image dataset was collected from 218 different people. The dataset contains a total of 2062 photos, each of which is sized at 100x100 pixels. We can access the dataset at https://www.kaggle.com/ardamavi/sign-language-digits-dataset. Figure 1 provides a preview of the American Sign Language Digits, while Table 1 displays the dataset's class distribution.

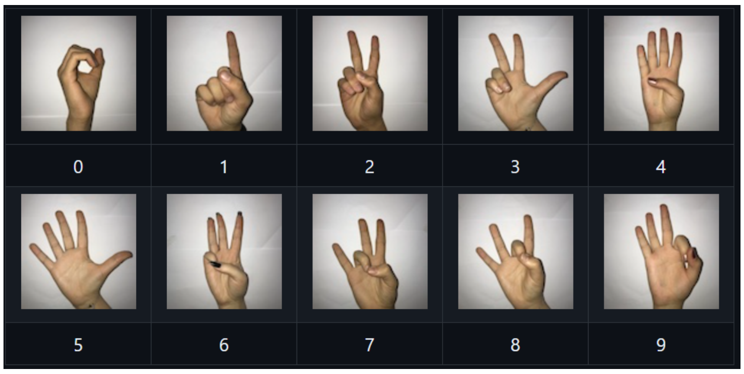


Fig. 1. Dataset Preview

Table 1
Dataset class distribution

Label	Number of Images
0	205
1	206
2	206
3	206
4	207
5	207
6	207
7	206
8	208
9	204

This distribution showcases the dataset's balanced nature, as the counts for each class are relatively uniform, with only slight variations. The balanced distribution is a crucial aspect of our dataset preparation, as it helps prevent potential biases during model training and evaluation. By ensuring a similar number of samples for each class, we enhance the model's ability to generalize across all classes effectively.

2.1.2 Dataset preprocessing

To prepare for our experiments, we adjusted the image size to 224x224 pixels, a precise prerequisite for our transfer learning model to ensure optimal processing. Following this adjustment, we partitioned the dataset into training, validation, and testing subsets, apportioning them in proportions of 80%, 15%, and 5% (1649 images were used for training, 310 images for validation, and 103 images for testing), respectively. A consistent split was employed. We set random_state to a specific value (in this case, 1), which ensures that the randomization for data splitting is reproducible. That is, if we were to run the code multiple times with the same random_state value, we would get the same split every time. This is useful for ensuring consistency in our experiments and results. Here is a brief explanation of the usefulness of each subset:

1. **Training data** (80%): This subset of data is used to train the model. The model learns from the training data to recognize patterns and make predictions. The training data must represent the problem being solved and must be large enough to allow the model to learn well.
2. **Validation data** (15%): This subset of data is used to optimize the model during the training process. The model is trained using the training data, then its performance is tested using the validation data. This is done to see the ability of the model during training to recognize patterns in general. Validation data can also be used to see the accuracy of the model created, if you are not satisfied with the results, you can change the parameters to improve the model's ability.

3. **Testing data** (5%): This subset of data is used to test the model after the training process is complete. The testing data must be separate from the training and validation data and must represent the problem being solved. The testing data should never be seen by the model before. Testing data is used to objectively evaluate the performance of the model and ensure that the model can be used to solve the problem at hand.

After rescaling the image size, data augmentation techniques were applied to the training set. These include rotation, allowing for a maximum angle of 20 degrees; horizontal shifts with a maximum displacement of 20% of the image width; and vertical shifts with a maximum movement of 20% of the image height. Furthermore, shear is introduced with a maximum angle of 20 degrees, while zooming is permitted with a maximum factor of 20%. Horizontal flipping is employed with a probability of 1, ensuring it is consistently applied. To address gaps in the augmented images, the nearest neighbor method is used for filling, enhancing the dataset, and facilitating robust training for machine learning models.

2.2 Experimental Setup

In this section, we provide a detailed account of the experimental setup used in our study. A well-documented experimental setup is crucial to ensure the credibility and reproducibility of our research.

Computational Resources

To conduct our experiments, we utilized a high-performance computing cluster equipped with multiple GPU nodes. The cluster consisted of NVIDIA Tesla V100 GPUs, which are well-suited for deep learning tasks due to their high processing capabilities. We employed this computational infrastructure to significantly expedite the training of our deep neural networks.

Libraries and Frameworks

The success of our experiments heavily relied on various software libraries and frameworks, which are essential components of the machine learning and deep learning pipeline. We employed the following key libraries and tools:

i. TensorFlow (Version 2.13.0): TensorFlow served as our primary deep learning framework. We leveraged its extensive ecosystem of tools and pre-built models to streamline our experimentation process.
ii. Keras: Keras, integrated with TensorFlow, was used for its high-level API, simplifying the implementation of neural network architectures and models.
iii. Scikit-Learn: Scikit-Learn played a crucial role in data preprocessing, feature engineering, and the evaluation of our models. It facilitated the implementation of traditional machine learning algorithms for comparative analysis.
iv. Python (Version 3.10.12): Python served as the programming language for our entire research pipeline, offering a wealth of scientific computing libraries and tools.

Origin of Trained Weights

The pre-trained weights and models used in our research were obtained from publicly available sources, which ensures the transparency and reproducibility of our work. We used the following pre-trained models:

i. **VGG-19**: We initialized our VGG-19 model with weights from the ImageNet database. This widely recognized dataset contains millions of labeled images and enables the model to capture general features and patterns in images.
ii. **ResNet-50**: For ResNet-50, we also utilized weights pretrained on the ImageNet dataset. The ImageNet pretraining allows the model to extract hierarchical features and leverage its deep architecture effectively.

By employing pre-trained weights, we aimed to harness the knowledge and feature representations learned from vast datasets, which can significantly boost the performance and convergence speed of our models when applied to our specific task.

In conclusion, our research benefits from the computational resources, software libraries, and pre-trained models used in our experimental setup. This detailed account ensures that our experiments are both credible and reproducible, laying a strong foundation for the validity of our findings and results.

2.3 Model Architecture Selection
2.3.1 VGG-19 as the pretrained model

To categorize images into 1000 object categories, Simonyan and Zisserman [21] proposed the utilization of VGG-19, a convolutional neural network consisting of 16 convolutional layers and 3 fully connected layers. This model requires a 224 x 224 image as input, and it provides the object label as its output. The architectural layout of VGG-19 is visually depicted in Figure 2.

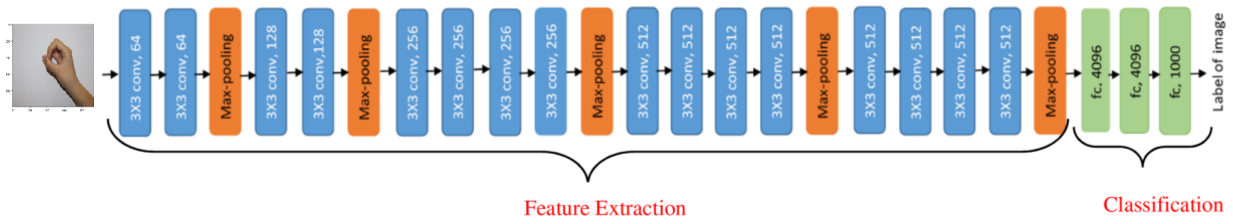


Fig. 2. Architecture of VGG-19 model (Source: Bansal *et al.,* [22])

To extract features from an image using the VGG-19 model, we remove the last layer, which is used for classification. The remaining layers are then used to generate a representation of the image, which contains information about its features, such as shape, color, and texture. Next, to classify images into one of 10 classes, we use fully connected layers. This layer takes the feature representation from the previous layer and predicts the image class. The VGG-19 Model Summary shown in Table 2.

Table 2
VGG-19 Model Summary

Model: "sequential"		
Layer (type)	Output Shape	Param#
Vgg-19 (Functional)	(None, 7, 7, 512)	20024384
global_average_pooling2d (GlobalAveragePooling2D)	(None, 512)	0
dense (Dense)	(None, 1024)	525312
dense_1 (Dense)	(None, 10)	10250
Total params: 20,559,946		
Trainable params: 535,562		
Non-trainable params: 20,024,384		

2.3.2 ResNet-50 as the pretrained model

ResNet-50 is a pre-trained convolutional neural network that is commonly used in transfer learning. It is trained on the ImageNet database, which contains a million images of 1000 categories. ResNet-50 is advantageous for transfer learning because it has learned a wide range of features that can be useful for a variety of tasks. ResNet-50 is also a deep network, which means it has many layers, allowing it to capture complex features in images. However, ResNet-50 is less prone to overfitting than VGG-19 due to its residual connections, which allow the network to learn more efficiently [23-25].

The architecture of ResNet-50 consists of 50 layers, including convolutional layers, pooling layers, and fully connected layers. The convolutional layers are grouped into blocks, with each block containing multiple convolutional layers followed by a max pooling layer. The residual connections in ResNet-50 allow the network to learn more efficiently by allowing the gradient to flow directly through the network without being diminished by the activation function. Figure 3 shows the visual layout of ResNet-50's architecture.

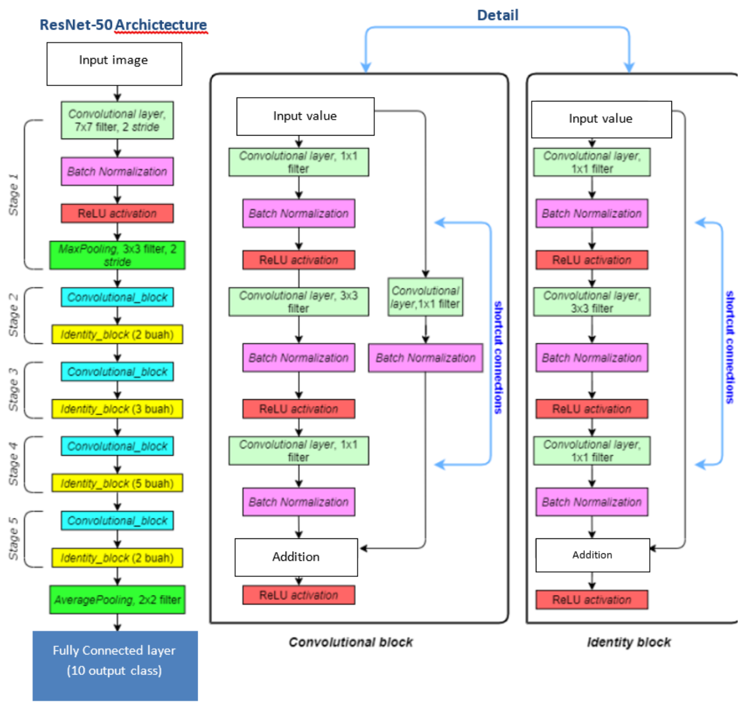


Fig. 3. Architecture of ResNet-50 Model

Table 3 shows the summary of the pre-trained ResNet-50 model used in this paper. Features are extracted from the images using the ResNet-50 model, followed by Global Average Pooling to reduce the number of dimensions. The classification method is then applied to the extracted features.

Table 3
ResNet-50 Model Summary

Model: "sequential"		
Layer (type)	Output Shape	Param#
resnet50 (Functional)	(None, 7, 7, 2048)	23587712
global_average_pooling2d (GlobalAveragePooling2D)	(None, 2048)	0
dense (Dense)	(None, 1024)	2098176
dense_1 (Dense)	(None, 10)	10250
Total params: 25,696,138		
Trainable params: 2,108,426		
Non-trainable params: 23,587,712		

2.4 *Proposed Ensemble Learning*

The ensemble method is a well-established approach in machine learning, widely recognized for its capability to enhance accuracy by combining predictions from multiple models [26-28]. In the specific context of hand-sign digit image classification, we can leverage the stacking method with the concatenate operator to effectively merge the predictions generated by the VGG-19 and ResNet-50 models. This ensemble strategy harnesses the complementary strengths of these models, ultimately contributing to improved classification performance [29]. We investigate two ensemble methods, namely stacking and bagging, because these two methods are the most well-known among other ensemble methods.

2.5 Performance Metrics

We use the confusion matrix as the determinant to assess how well the suggested model performs.

2.5.1 Accuracy

Accuracy is a key factor that determines how accurately a model or architecture performs the classification. The following equation shows the formula:

$$Accuracy = \frac{\sum_{i=1}^{N} Correct_i}{N}$$

2.5.2 Precision

A model's precision is a parameter used to gauge how well it can categorize positive cases.

$$Precision = \frac{\sum_{i=1}^{N} TruePositive_{ic}}{\sum_{i=1}^{N} PredictedPositive_c}$$

2.5.3 Sensitivity (Recall)

A metric called sensitivity or recall gauges how well a model or architecture can distinguish between positive and negative cases.

$$Recall = \frac{\sum_{i=1}^{N} TruePositive_{ic}}{\sum_{i=1}^{N} ActualPositive_{c}}$$

2.5.4 F1-score

The F1-score, which ranges between 0 and 1, is a crucial indicator for determining how accurately a classification system performs; in reality, it assesses the efficiency of two parameters, accuracy and precision.

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

3. Results

In this research, we give the same treatment (data splitting and augmentation) to each dataset before entering the model, both ensembles learning and individual mode. For transfer learning, we set trainable property of VGG-19 layers and ResNet-50 layers to false, which prevented the weights (obtained by VGG-19 and ResNet-50 from ImageNet) from being updated during training with the ASL dataset. Each model was trained for 50 epochs, and validation loss was calculated for each epoch. We used SparseCategoricalCrossentropy as loss function because we have multiclass single label on target. The optimization algorithm for all model was Adam, with a learning rate of 0.001.

3.1 Classification Using Individual Model
3.1.1 VGG-19

Figure 4 illustrates the VGG-19 with 50 epochs' training vs validation accuracy (right side) and training vs validation loss (left side). The loss graph shows the loss value of the VGG-19 model at each epoch. Loss is a measure of how well the machine learning model predicts the target data. The lower the loss value, the better the machine learning model predicts the target data.

The accuracy graph shows the accuracy value of the VGG-19 model at each epoch. Accuracy is a measure of how often the machine learning model predicts the target data correctly. The higher the accuracy value, the more often the machine learning model predicts the target data correctly.

In the Figure 4, the loss graph shows that the loss value decreases as the epoch increases. This means that VGG-19 model becomes more accurate in predicting the target data as the epoch increases. The accuracy graph shows that the accuracy value increases as the epoch increases. This means that VGG-19 model becomes more accurate in predicting the target data as the epoch increases.

Based on this information, it can be concluded that VGG-19 model becomes more accurate in predicting the target data as the epoch increases. This means that VGG-19 model successfully learned from the training data and became better at predicting the target data.

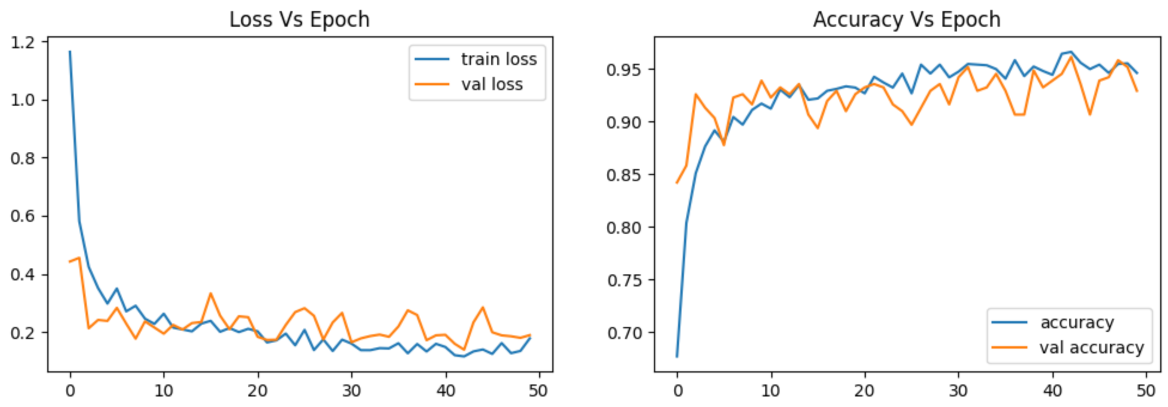
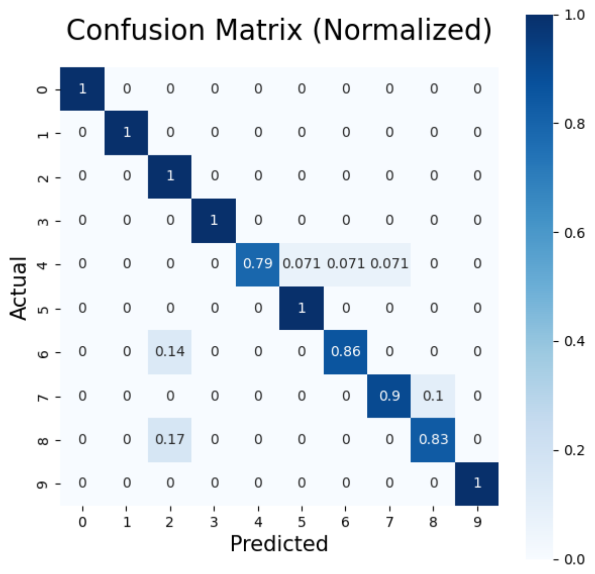


Fig. 4. Training Result of VGG-19

Confusion Matrix of VGG-19 given in Figure 5. From the figure, hand sign number 4 is most often misclassified, followed by number 8, 6 and then 7.



	0	1	2	3	4	5	6	7	8	9
0	1	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0
4	0	0	0	0	0.79	0.071	0.071	0.071	0	0
5	0	0	0	0	0	1	0	0	0	0
6	0	0	0.14	0	0	0	0.86	0	0	0
7	0	0	0	0	0	0	0	0.9	0.1	0
8	0	0	0.17	0	0	0	0	0	0.83	0
9	0	0	0	0	0	0	0	0	0	1

Fig. 5. Confusion Matrix of VGG-19

3.1.1 ResNet-50

From Figure 6, almost the same as VGG-19 model, ResNet-50 model successfully learned from the training data and became better at predicting the target data.

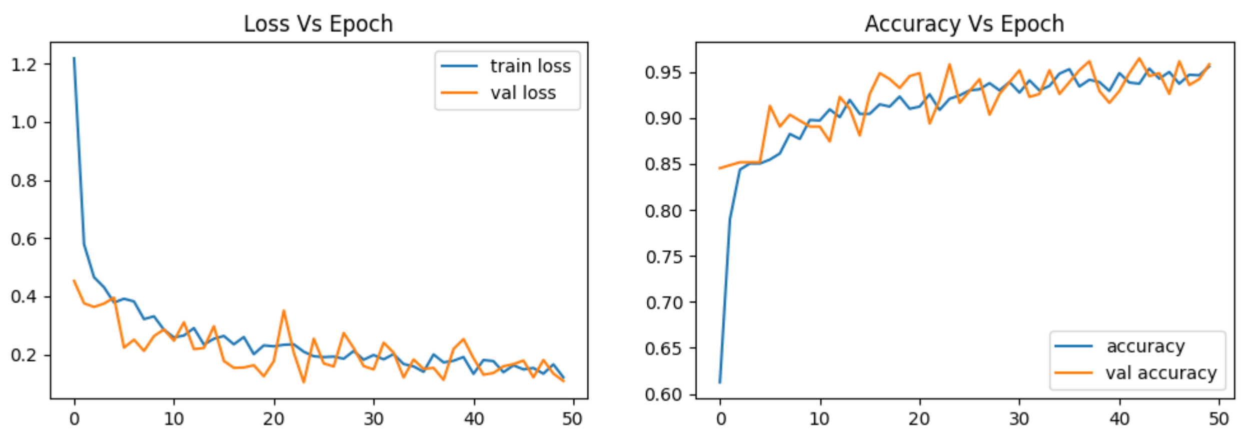
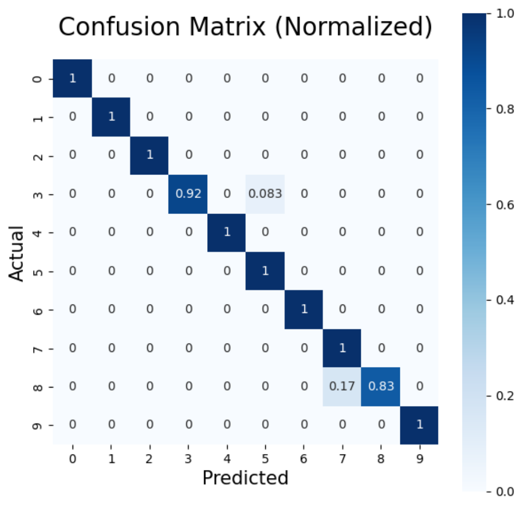


Fig. 6. Training Result of ResNet-50

Figure 7 shows the ResNet-50 Confusion Matrix. As individual model, ResNet-50 has higher accuracy than VGG-19 on the ASL digit dataset, there are only two sign languages that are misclassified, namely number 8 around 17% of the data test and number 3 at 8%of the data test.



	0	1	2	3	4	5	6	7	8	9
0	1	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0
3	0	0	0	0.92	0	0.083	0	0	0	0
4	0	0	0	0	1	0	0	0	0	0
5	0	0	0	0	0	1	0	0	0	0
6	0	0	0	0	0	0	1	0	0	0
7	0	0	0	0	0	0	0	1	0	0
8	0	0	0	0	0	0	0	0.17	0.83	0
9	0	0	0	0	0	0	0	0	0	1

Fig. 7. Confusion matrix of ResNet-50

3.2 Classification Using Ensemble Model
3.2.1 Classification using stacking

In this paper, we use the stacking method to ensemble the model. The ensemble stacking method uses several base classifiers in the learning process. There are two stages to learning stacking. Stage 1, Each base classifier (VGG-19 and ResNet-50) used is trained using the same dataset so as to produce the results of each prediction. Stage 2, the meta-classifier, takes the predicted results from the base classifier as input to determine which class is most likely to be based on the test data. The predictions from these models are combined using the "concatenation" operation before being fed to the merging model (meta-learner) for the final prediction. This operation allows the combining model to take information from each individual model and decide how best to combine the predictions. The architecture of the ensemble model is illustrated in Figure 8. We applied some combination in **Stage 2**, to find the best model.

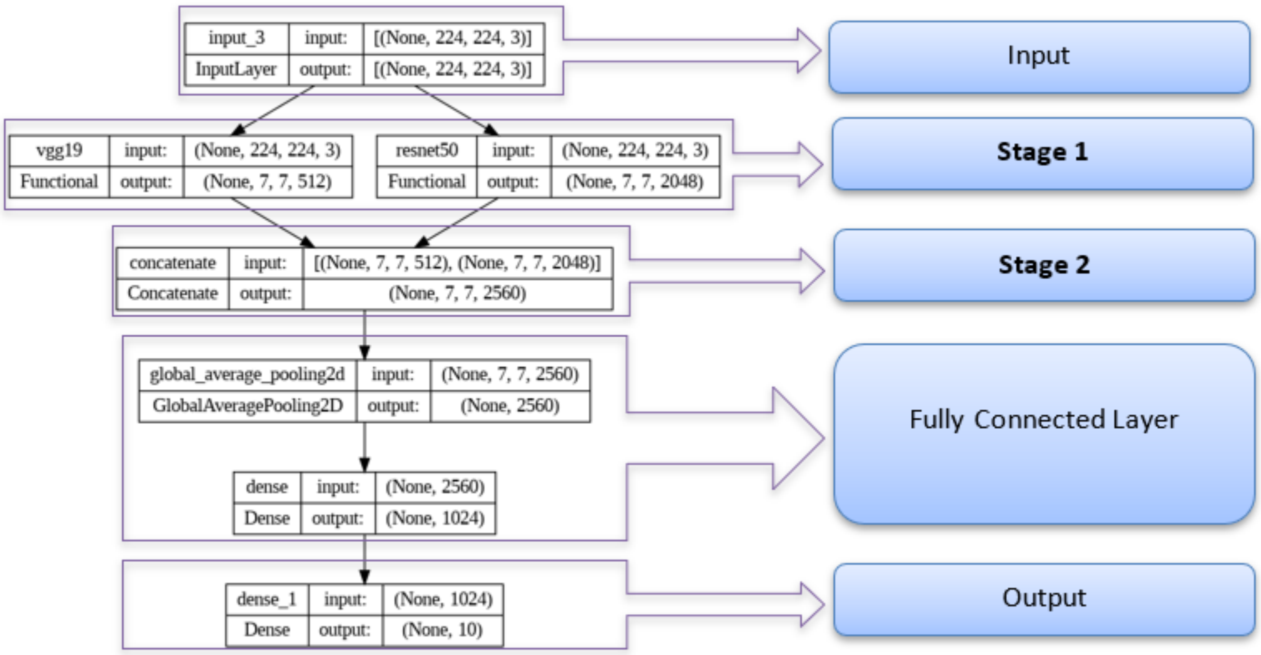

input_3 | input: | [(None, 224, 224, 3)]
InputLayer | output: | [(None, 224, 224, 3)]
vgg19 | input: | (None, 224, 224, 3)
Functional | output: | (None, 7, 7, 512)
resnet50 | input: | (None, 224, 224, 3)
Functional | output: | (None, 7, 7, 2048)
concatenate | input: | [(None, 7, 7, 512), (None, 7, 7, 2048)]
Concatenate | output: | (None, 7, 7, 2560)
global_average_pooling2d | input: | (None, 7, 7, 2560)
GlobalAveragePooling2D | output: | (None, 2560)
dense | input: | (None, 2560)
Dense | output: | (None, 1024)
dense_1 | input: | (None, 1024)
Dense | output: | (None, 10)
Input
Stage 1
Stage 2
Fully Connected Layer
Output

Fig. 8. Architecture of Ensemble Model

3.2.1.1 Ensemble stacking model I (*1 ResNet50 and 1 VGG-19*)

From Figure 9-14, we find that with the right layer combination can increase the accuracy of the model.

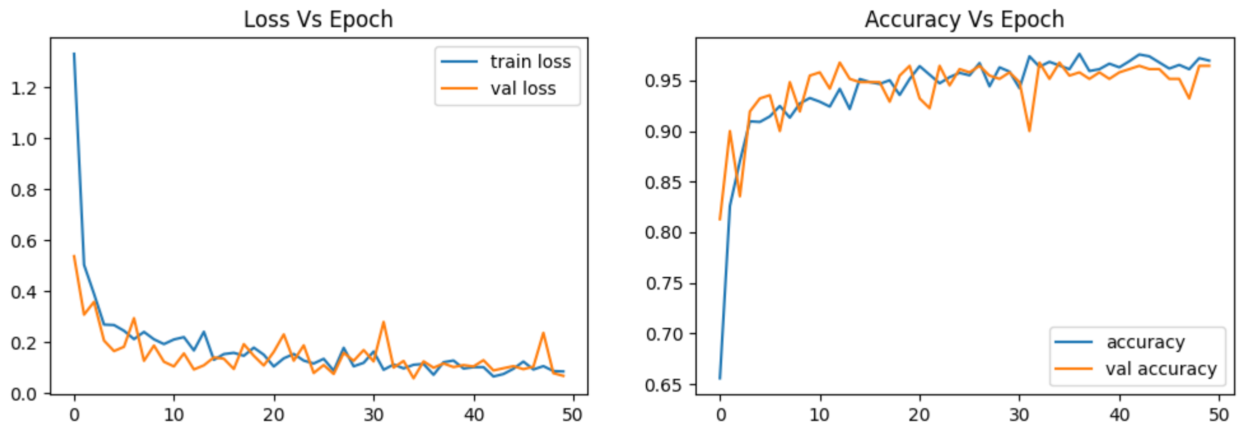


Fig. 9. Training Result of Ensemble with 1 ResNet-50 and 1 VGG-19

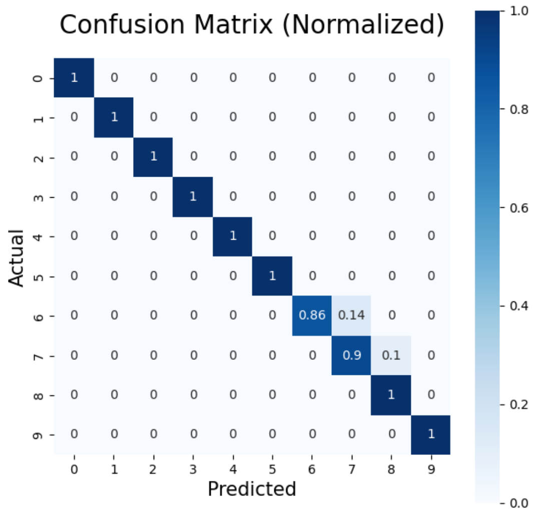


Fig. 10. Confusion matrix of Ensemble Model I

3.2.1.2 Ensemble stacking model II (2 ResNet50 and 1 VGG-19)

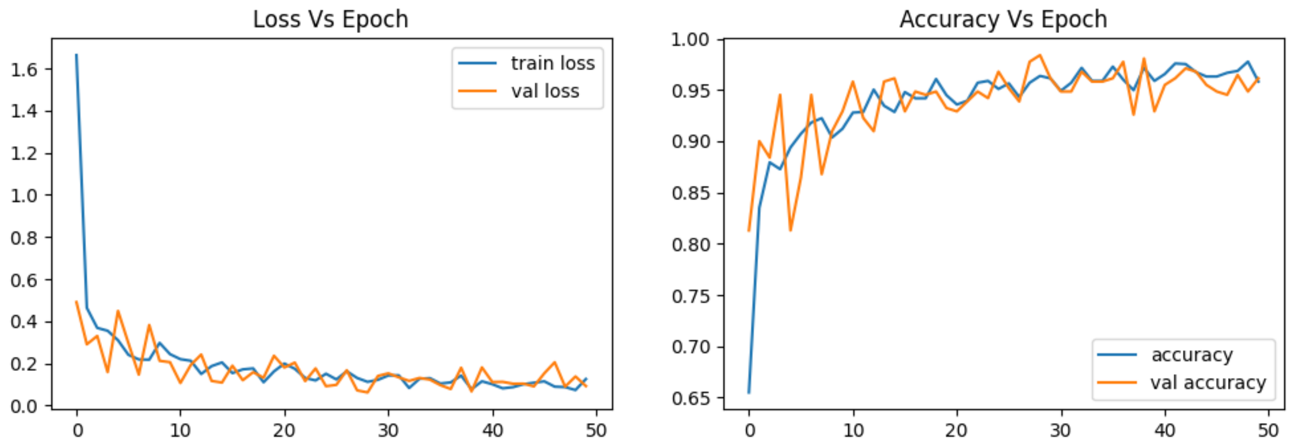


Fig. 11. Training Result of Ensemble with 2 ResNet-50 and 1 VGG-19

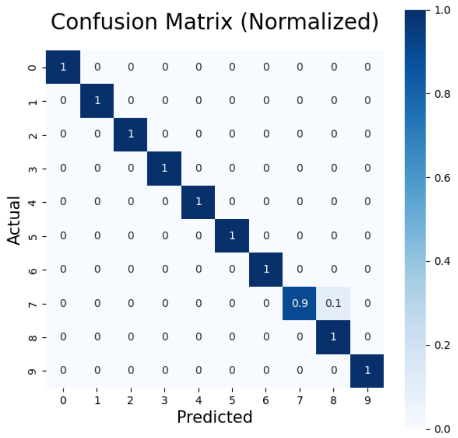


Fig. 12. Confusion matrix of Ensemble Model II

3.2.1.3 Ensemble stacking model III (*1 ResNet50 and 2 VGG-19*)

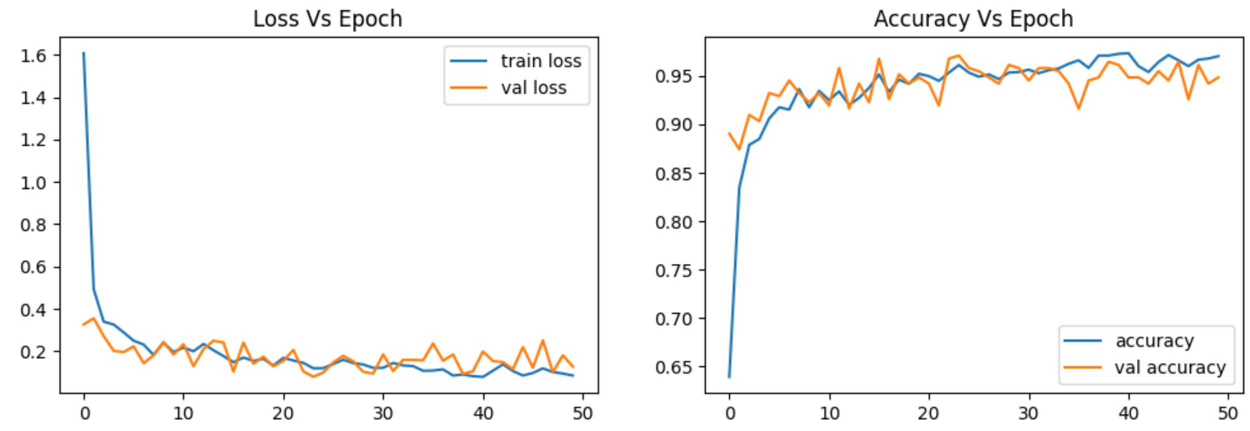


Fig. 13. Training Result of Ensemble with 1 ResNet-50 and 2 VGG-19

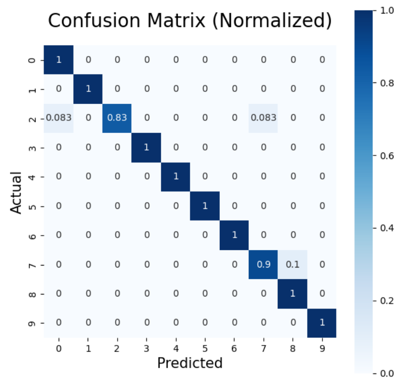


Fig. 14. Confusion matrix of Ensemble Model III

3.2.2 Classification using bagging

To implement bagging technique, we start by training multiple different base models using the same dataset. These models can come from different architectures or may simply have different initializations. Once these models are trained, we combine the predictions from all the models by taking the average of their predictions. This is a straightforward yet effective method to produce a more reliable final prediction. Figure 15 show the architecture of bagging ensemble.

The primary advantage of Ensemble Bagging with Average Weight is increased accuracy, especially when the base models used are diverse in terms of architecture, initialization, or hyperparameters. This technique also helps reduce uncertainty in predictions, leading to more consistent results.

In the world of machine learning, the key to achieving optimal results is to experiment with various ensemble techniques, as discussed above. By employing Ensemble Bagging with Average Weight, we can maximize the potential of our base models and produce stronger, more reliable predictions. This method is one of many tools that can assist us in tackling a variety of machine learning tasks and data analysis.

Figure 16 shows a plot of the difference between training loss and accuracy. The training loss plot shows that the loss decreases gradually as the number of epochs increases. This shows that the training model is learning and reducing its errors.

The training accuracy plot shows that the accuracy increases gradually as the number of epochs increases. This shows that the training model is getting better at predicting the correct label. At epoch 50, the training loss is 0.2 and the training accuracy is 0.8. This means that the training model has reached a high accuracy.

From Figure 17, it shows that there are still prediction misses in digits 7. However, overall the performance of this model is quite good at 98.06%.

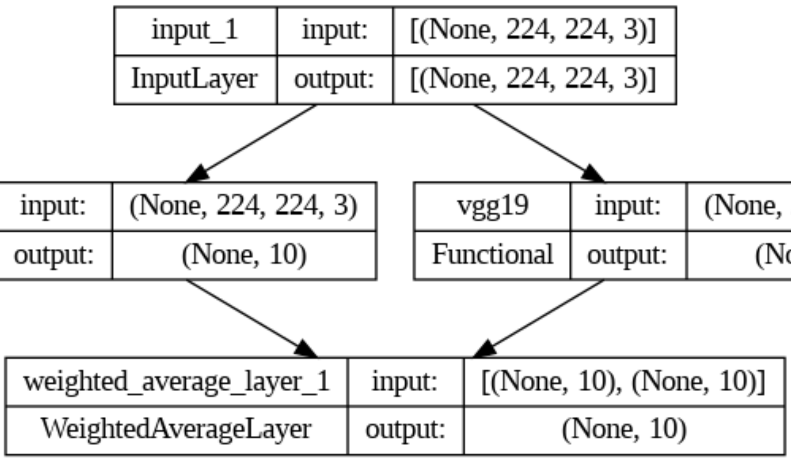

input_1 | input: | [(None, 224, 224, 3)]
InputLayer | output: | [(None, 224, 224, 3)]
resnet50 | input: | (None, 224, 224, 3)
Functional | output: | (None, 10)
vgg19 | input: | (None, 224, 224, 3)
Functional | output: | (None, 10)
weighted_average_layer_1 | input: | [(None, 10), (None, 10)]
WeightedAverageLayer | output: | (None, 10)

Fig. 15. Bagging Architecture

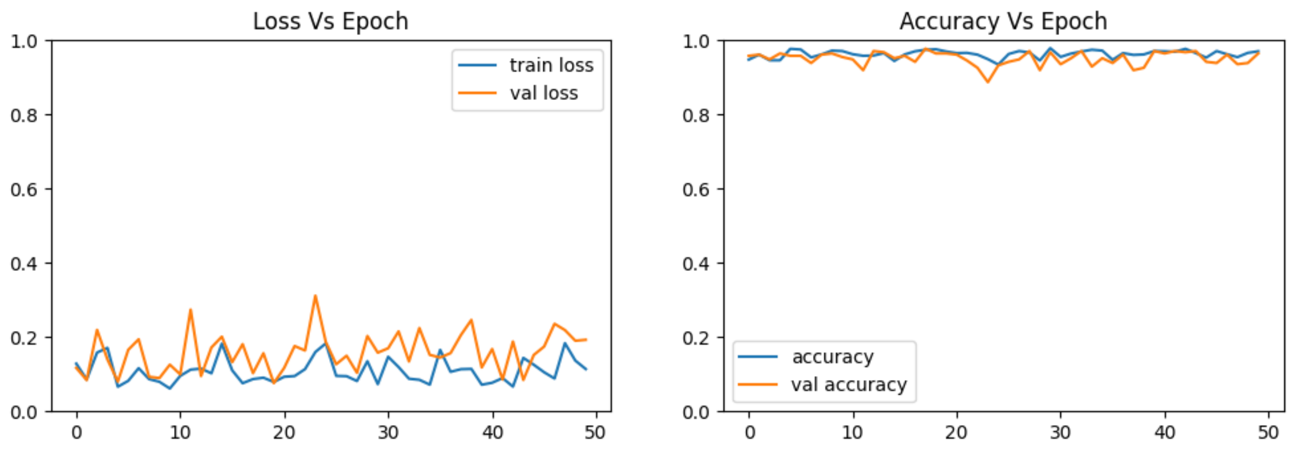
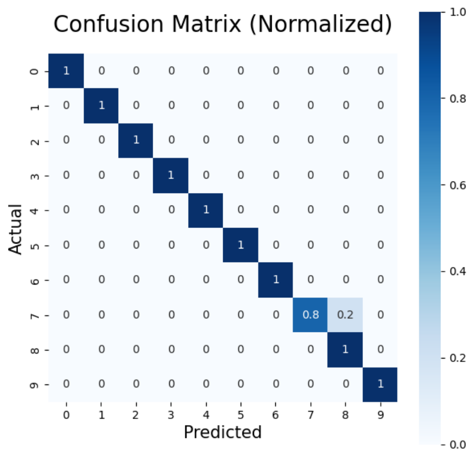


Fig. 16. Training Result of Bagging



	0	1	2	3	4	5	6	7	8	9
0	1	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0
4	0	0	0	0	1	0	0	0	0	0
5	0	0	0	0	0	1	0	0	0	0
6	0	0	0	0	0	0	1	0	0	0
7	0	0	0	0	0	0	0	0.8	0.2	0
8	0	0	0	0	0	0	0	0	1	0
9	0	0	0	0	0	0	0	0	0	1

Fig. 17. Confusion matrix of Bagging

4. Conclusions

This research addresses the important task of hand signal digit recognition, which has significant applications in sign language communication and assistive technologies. Conventional methods face challenges due to the complexity of hand gestures, relying on manual feature engineering and shallow machine learning algorithms. To address these challenges, a new ensemble transfer learning approach is proposed, leveraging the power of convolutional neural networks (CNN) and pre-trained models such as VGG-19 and ResNet-50.

Evaluation results show that the Ensemble Model, which combines VGG-19 and ResNet-50 synergistically in a stacked ensemble framework, almost completely outperforms the individual models. With 99.03% accuracy, the Ensemble Model shows superior performance in hand gesture digit classification. This innovation has the potential to advance the introduction of sign language, making it more accessible and effective for individuals with hearing loss. From table 4, it can be seen that even though the dataset we use is small, it still produces high accuracy, when compared to other models with larger datasets.

Table 4
Comparison of this study with existing related studies

Study	Model	Dataset	Accuracy(%)
This Study	VGG-19	ASL Digits	94.17
This Study	ResNet-50	ASL Digits	98.06
This Study	Stacking Model I	ASL Digits	98.06
This Study	Stacking Model II	ASL Digits	99.03
This Study	Stacking Model III	ASL Digits	97.09
This Study	Bagging	ASL Digits	98.06
Dyal *et al.,* [30]	Combines unsupervised domain adaptation (UDA)	ASL Digits	91.32
Tyagi *et al.,* [31]	HFSC	ISL	97.87
Mistree *et al.,* [32]	MobileNet	ISL	97.27

Additionally, the application of transfer learning and ensemble techniques promises to improve model adaptability and generalization, even in situations with limited labelled datasets. In addition to its direct impact on sign language recognition, this research has broader implications in the fields of human-computer interaction and assistive technology, thereby providing a multidisciplinary approach to solving real-world challenges. Ultimately, these efforts contribute to increased accessibility and communication for individuals with hearing loss and hold the promise of a more inclusive society. For future research, it is hoped that other ensemble models can be applied, and using more extensive dataset.

Acknowledgement
This research was funded by a grant from HIBAH PENELITIAN INTERNAL FAKULTAS MIPA UNIVERSITAS HASANUDDIN TAHUN 2023 (944/UN4.11/PT.01.02/2023).

References

[1] Rao, G. Anantha, K. Syamala, P. V. V. Kishore, and A. S. C. S. Sastry. "Deep convolutional neural networks for sign language recognition." In *2018 conference on signal processing and communication engineering systems (SPACES)*, pp. 194-197. IEEE, 2018. https://doi.org/10.1109/SPACES.2018.8316344

[2] Wangchuk, Karma, Panomkhawn Riyamongkol, and Rattapoom Waranusast. "Real-time Bhutanese sign language digits recognition system using convolutional neural network." *Ict Express* 7, no. 2 (2021): 215-220. https://doi.org/10.1016/j.icte.2020.08.002

[3] Wang, Shuai, Xiaojun Xia, Lanqing Ye, and Binbin Yang. "Automatic detection and classification of steel surface defect using deep convolutional neural networks." *Metals* 11, no. 3 (2021): 388. https://doi.org/10.3390/met11030388

[4] Alom, Md Shahin, Md Jahid Hasan, and Md Ferdous Wahid. "Digit recognition in sign language based on convolutional neural network and support vector machine." In *2019 International Conference on Sustainable Technologies for Industry 4.0 (STI)*, pp. 1-5. IEEE, 2019. https://doi.org/10.1109/STI47673.2019.9067999

[5] Hossain, Md Bipul, Apurba Adhikary, and Sultana Jahan Soheli. "Sign language digit recognition using different convolutional neural network model." *Asian Journal of Research in Computer Science* 6, no. 2 (2020): 16-24. https://doi.org/10.9734/ajrcos/2020/v6i230154

[6] Kalam, Md Abul, Md Nazrul Islam Mondal, and Boshir Ahmed. "Rotation independent digit recognition in sign language." In *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, pp. 1-5. IEEE, 2019. https://doi.org/10.1109/ECACE.2019.8679172

[7] Lin, Hsien-I., Ming-Hsiang Hsu, and Wei-Kai Chen. "Human hand gesture recognition using a convolution neural network." In *2014 IEEE International Conference on Automation Science and Engineering (CASE)*, pp. 1038-1043. IEEE, 2014. https://doi.org/10.1109/CoASE.2014.6899454

[8] Paul, Pias, Moh Anwar-Ul-Azim Bhuiya, Md Ayat Ullah, Molla Nazmus Saqib, Nabeel Mohammed, and Sifat Momen. "A modern approach for sign language interpretation using convolutional neural network." In *Pacific Rim International Conference on Artificial Intelligence*, pp. 431-444. Cham: Springer International Publishing, 2019. https://doi.org/10.1007/978-3-030-29894-4_35

[9] Sordo, Margarita, and Qing Zeng. "On sample size and classification accuracy: A performance comparison." In *International Symposium on Biological and Medical Data Analysis*, pp. 193-201. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. https://doi.org/10.1007/11573067_20.

[10] Prusa, Joseph, Taghi M. Khoshgoftaar, and Naeem Seliya. "The effect of dataset size on training tweet sentiment classifiers." In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pp. 96-102. IEEE, 2015. https://doi.org/10.1109/ICMLA.2015.22

[11] Rahman, M. Shafiqur, and Mahbuba Sultana. "Performance of Firth-and logF-type penalized methods in risk prediction for small or sparse binary data." *BMC medical research methodology* 17, no. 1 (2017): 1-15. https://doi.org/10.1186/s12874-017-0313-9

[12] Althnian, Alhanoof, Duaa AlSaeed, Heyam Al-Baity, Amani Samha, Alanoud Bin Dris, Najla Alzakari, Afnan Abou Elwafa, and Heba Kurdi. "Impact of dataset size on classification performance: an empirical evaluation in the medical domain." *Applied Sciences* 11, no. 2 (2021): 796. https://doi.org/10.3390/app11020796

[13] Camgoz, Necati Cihan, Simon Hadfield, Oscar Koller, and Richard Bowden. "Using convolutional 3d neural networks for user-independent continuous gesture recognition." In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 49-54. IEEE, 2016. https://doi.org/10.1109/ICPR.2016.7899606

[14] Andrew, Andrew, and Handri Santoso. "Compare VGG19, ResNet50, Inception-V3 for review food rating." *Sinkron: jurnal dan penelitian teknik informatika* 7, no. 2 (2022): 845-494. https://10.33395/sinkron.v7i2.11383

[15] Shah, Syed Rehan, Salman Qadri, Hadia Bibi, Syed Muhammad Waqas Shah, Muhammad Imran Sharif, and Francesco Marinello. "Comparing Inception V3, VGG 16, VGG 19, CNN, and ResNet 50: A Case Study on Early Detection of a Rice Disease." *Agronomy* 13, no. 6 (2023): 1633. https://10.3390/agronomy13061633

[16] Ikechukwu, A. Victor, S. Murali, R. Deepu, and R. C. Shivamurthy. "Global Transitions Proceedings." https://10.1016/j.gltp.2021.08.027

[17] Pangilinan, John Raphael, Jericho Legaspi, and Noel Linsangan. "InceptionV3, ResNet50, and VGG19 Performance Comparison on Tomato Ripeness Classification." In *2022 5th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, pp. 619-624. IEEE, 2022. https://10.1109/ISRITI56927.2022.10052920

[18] Siddik, A. Muh Amil. "Comparison of Transfer Learning Algorithm Performance in Hand Sign Language Digits Image Classification." *Jurnal Matematika, Statistika dan Komputasi* 20, no. 1 (2023): 75-89. https://doi.org/10.20956/j.v20i1.26503

[19] Muñoz-Saavedra, Luis, Elena Escobar-Linero, Javier Civit-Masot, Francisco Luna-Perejón, Antón Civit, and Manuel Domínguez-Morales. "A Robust Ensemble of Convolutional Neural Networks for the Detection of Monkeypox Disease from Skin Images." *Sensors* 23, no. 16 (2023): 7134. https://doi.org/10.3390/s23167134

[20] Tasci, Erdal, Caner Uluturk, and Aybars Ugur. "A voting-based ensemble deep learning method focusing on image augmentation and preprocessing variations for tuberculosis detection." *Neural Computing and Applications* 33, no. 22 (2021): 15541-15555. https://doi.org/10.1007/s00521-021-06177-2

[21] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014). https://doi.org/10.48550/arXiv.1409.1556

[22] Bansal, Monika, Munish Kumar, Monika Sachdeva, and Ajay Mittal. "Transfer learning for image classification using VGG19: Caltech-101 image data set." *Journal of ambient intelligence and humanized computing* (2021): 1-12. https://doi.org/10.1007/s12652-021-03488-z

[23] Mascarenhas, Sheldon, and Mukul Agarwal. "A comparison between VGG16, VGG19 and ResNet50 architecture frameworks for Image Classification." In *2021 International conference on disruptive technologies for multi-disciplinary research and applications (CENTCON)*, vol. 1, pp. 96-99. IEEE, 2021. https://10.1109/CENTCON52345.2021.9687944

[24] Rathi, Pulkit, Raj Kuwar Gupta, Soumya Agarwal, and Anupam Shukla. "Sign language recognition using resnet50 deep neural network architecture." In *5th International Conference on Next Generation Computing Technologies (NGCT-2019)*. 2020. https://10.2139/ssrn.3545064

[25] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770-778. 2016. https://10.1109/CVPR.2016.90

[26] Wolpert, David H. "Stacked generalization." *Neural networks* 5, no. 2 (1992): 241-259. https://10.1016/S0893-6080(05)80023-1

[27] Džeroski, Saso, and Bernard Ženko. "Is combining classifiers with stacking better than selecting the best one?." *Machine learning* 54 (2004): 255-273. https://10.1023/B:MACH.0000015881.36452.6e

[28] Ma, Zhiyuan, Ping Wang, Zehui Gao, Ruobing Wang, and Koroush Khalighi. "Ensemble of machine learning algorithms using the stacked generalization approach to estimate the warfarin dose." *PloS one* 13, no. 10 (2018): e0205872. https://10.1371/journal.pone.0205872

[29] Zhang, Hongyan, Manhui Lin, Guangyi Yang, and Liangpei Zhang. "ESCNet: An end-to-end superpixel-enhanced change detection network for very-high-resolution remote sensing images." *IEEE Transactions on Neural Networks and Learning Systems* (2021). https://doi.org/10.1109/TNNLS.2021.3089332

[30] Dayal, Aveen, M. Aishwarya, S. Abhilash, C. Krishna Mohan, Abhinav Kumar, and Linga Reddy Cenkeramaddi. "Adversarial Unsupervised Domain Adaptation for Hand Gesture Recognition Using Thermal Images." *IEEE Sensors Journal* 23, no. 4 (2023): 3493-3504. https://doi.org/10.1109/JSEN.2023.3235379

[31] Tyagi, Akansha, and Dr Sandhya Bansal. "Hybrid FAST-SIFT-CNN (HFSC) approach for vision-based Indian sign language recognition." *International Journal Of Computing and Digital System* (2021). http://dx.doi.org/10.12785/ijcds/110199.

[32] Mistree, Kinjal, Devendra Thakor, and Brijesh Bhatt. "Indian alphabets and digits sign recognition using pretrained model." In *Smart Intelligent Computing and Applications, Volume 2: Proceedings of Fifth International Conference on Smart Computing and Informatics (SCI 2021)*, pp. 13-20. Singapore: Springer Nature Singapore, 2022. https://doi.org/10.1007/978-981-16-9705-0_2