Study on real-time gesture recognition using convolutional neural network based on SMOTE method

Pengyu Lin *

Maynooth International Engineering College, Fuzhou University, Fuzhou, China 350000

* Corresponding Author Email: 832204219@fzu.edu.cn

Abstract. Today, there are still many deaf individuals in the world, and enabling barrier-free communication for them remains a global challenge. This study aims to investigate real-time gesture recognition using convolutional neural networks and the mediapipe library, addressing data imbalance through the SMOTE method. In this paper, we investigate the developmental trajectory, underlying principles, and multifaceted applications of sign language recognition technology across various domains. We also introduce a sign language recognition system utilizing a deep learning algorithm to improve accuracy and stability, thereby contributing to the continuous progress of sign language recognition technology. The results demonstrate that SMOTE effectively enhances the experimental data, while mediapipe plays a significant role in improving recognition accuracy. The proposed method shows promise in real-time gesture recognition for the deaf, offering a viable solution to enhance their communication. This research contributes to the field of assistive technologies for individuals with disabilities and has potential for further optimization and real-world applications.

Keywords: CNN, gesture recognition, smote, mediapipe.

1. Introduction

Sign language recognition emerges as a pivotal field dedicated to translating human gestures into comprehensible digital signals. With the continuous evolution of computer vision and machine learning technologies, sign language recognition has become instrumental across diverse sectors such as human-computer interaction, virtual reality, augmented reality, healthcare, and security surveillance. In this digital era, sign language recognition not only represents technological progress but also serves as a vital conduit for fostering communication and interaction between humans and technology.

From its nascent stages to today's highly sophisticated and precise recognition systems, sign language recognition technology has made remarkable strides. It not only identifies fundamental gesture movements but also comprehends intricate gesture combinations and even detects subtle finger movements. This precision empowers sign language recognition technology to play an increasingly pivotal role in human-computer interaction, enabling users to naturally and intuitively interact with computers via gestures, thereby enhancing user experience and operational efficiency without reliance on keyboards or mice.

Furthermore, sign language recognition technology holds significant promise in the healthcare domain. By interpreting patients' gestures, medical devices can execute more accurate operations, aiding healthcare professionals in diagnosing and treating diseases more effectively. Additionally, this technology can support rehabilitation training, facilitating physical therapy and functional recovery for rehabilitation patients.

Despite substantial advancements, sign language recognition technology encounters various challenges and hurdles. Issues like gesture recognition in complex backgrounds, identification of gestures from multiple individuals, and the impact of lighting and shadows on recognition accuracy represent ongoing challenges that necessitate innovative solutions. Hence, continued research and innovation are imperative to propel the evolution of sign language recognition technology.

In this paper, we delve into the developmental trajectory, underlying principles, and multifaceted applications of sign language recognition technology across diverse domains. Additionally, we

propose A recognition system for sign language using deep learning algorithm designed to enhance accuracy and stability, contributing to the ongoing advancement of sign language recognition technology.

2. Related works

The research landscape in hand gesture recognition has evolved significantly over the years, marked by the dedicated efforts of numerous researchers. By employing various methodologies and techniques, these scholars have collectively advanced the field, addressing a multitude of challenges to enhance accuracy and efficiency in recognizing sign language gestures.

Asadi-Aghbolaghi et al. (2017) [1] delved into the realm of sign language recognition, likely leveraging deep learning methodologies to enhance accuracy and efficiency. Their work stands as a testament to the growing influence of modern computational techniques in this domain. Building upon this foundation, Ferreira et al. (2019) [2] undertook a comprehensive exploration of sign language recognition, tackling specific challenges such as inter and intra-subject variability and varying illumination conditions. Their findings provided valuable insights into mitigating these challenges to improve recognition performance.

Guo et al. (2017) [3] further propelled the field forward by introducing novel algorithms and techniques aimed at bolstering accuracy and robustness in sign language recognition systems. Their contributions underscored the importance of continual innovation in algorithmic development. Addressing practical challenges inherent in real-world applications, Lim et al. (2019) [4] focused on refining sign language recognition systems to handle partial occlusions, varying viewpoints, and background artifacts. Their research highlighted the significance of robustness in ensuring reliable performance across diverse scenarios. Meanwhile, Oberweger et al. (2015) [5] explored techniques to enhance the resilience of sign language recognition systems under variable conditions, laying the groundwork for improved adaptability in real-world settings. Rastgoo et al. (2018) [6] built upon these advancements by proposing novel methodologies and algorithms aimed at further enhancing recognition accuracy and efficiency. Their contributions added depth to the repertoire of techniques available for tackling the complexities of SLR. Zheng et al. (2017) [7] directed their research efforts towards addressing domain-specific challenges in SLR, leading to the development of more resilient and accurate systems tailored to the unique characteristics of sign language gestures. Additionally, the work of Wadhawan and Kumar (2020) [8] on hand sign and pose estimation provided practical insights with direct applications in sign language recognition systems, further enriching the interdisciplinary discourse in this field. While modern advancements in deep learning have significantly shaped contemporary research in sign language recognition, it's important to acknowledge the pioneering efforts of Zimmerman et al. (1987) [9]. Although their work predates the deep learning era, their exploration of hand gesture interfaces using magnetic flux sensors laid a foundational framework for subsequent research endeavors.

Together, these seminal contributions underscore the collective endeavor towards advancing sign language recognition, characterized by a diverse array of methodologies and techniques aimed at overcoming the inherent challenges in gesture recognition. Through continual innovation and interdisciplinary collaboration, researchers continue to push the boundaries of what is achievable in this critical field.

3. Method

In the experiment, CNN, SMOTE, and Mediapipe were employed. Remarkably promising outcomes were obtained, with the SMOTE technique notably exhibiting exceptional performance, yielding outstanding results in this study. Hereafter, the methodologies utilized in our research will be delineated.

3.1. CNN

An important approach in the field of deep learning is the convolutional neural network (CNN), a sort of feedforward neural network that is well-known for its convolutional computations and deep structure. CNNs are dubbed SIANNs because they are excellent at representation learning, which allows them to classify incoming data according to its hierarchical structure while maintaining shift-invariant capabilities.

The architecture of CNNs typically comprises three layers: the input layer, hidden layer, and output layer. The hidden layer further encompasses the convolutional layer, pooling layer, and fully connected layer. Let's delve into the distinct functions of each layer and elucidate their operations.

Input layer. Additional data can be inserted into CNN through its input layer. In both one- and two-dimensional convolutional neural networks, the input layers make up the majority of the network architecture. An array can be one or two dimensions in the case of the one-dimensional convolutional neural network input layer, and two or three dimensions in the case of the two-dimensional convolutional neural network input layer. Two-dimensional convolutional neural networks are typically used to accept black-and-white or RGB-channel images because CNN is extensively utilized in the field of computer vision.

The learned data must be normalized before being fed into the convolutional neural network because it learns using the gradient descent algorithm. The original pixel values distributed at [0,255] must be normalized to the interval [0,1] in the case of pixels. Standardization of input features helps convolutional neural networks learn more effectively and perform better.

Convolution layer. Feature extraction from input data is the main function of the convolutional layer. It is made up of several convolutional kernels, each of which is similar to a neuron in a feedforward neural network and represents a weight coefficient and bias vector. While in operation, the convolutional kernel repeatedly scans the input features, averaging deviation quantities through matrix element multiplication and summation inside the receptive field.

$$Z^{l+i}(i,j) = \left[Z^{l} \otimes \omega^{l+i}\right](i,j) + b = \sum_{k=1}^{K_l} \sum_{x=1}^{f} \sum_{y=1}^{f} \left[Z_k^l(s_0 i + x, s_0 j + y)\omega_k^{l+1}(x,y)\right] + b$$
$$(i,j) \in \{0,1,\ldots L_{l+1}\} \quad L_{l+1} = \frac{L_l + 2p - f}{s_0} + 1 \tag{1}$$

Finding the cross-correlation is the same as solving for the summation portion of the formula. L_{l+1} is the size of the Z_{l+1} feature map, which is assumed to be the same length and width. b is the deviation quantity. Z^{l} and $\mathrm{Z}^{\mathrm{l+1}}$ represent the $\mathrm{l}+1$ convolution input and output of the first layer, also known as the feature map. The feature map's corresponding pixel is $Z(i,j)$, its number of channels is K, f, s_0, and p are its parameters for the convolution layer, which stand for the convolution kernel's size, the convolution stride, and the number of padding layers. The stride parameter determines the movement of the convolutional filter across the input data, with a larger stride resulting in a smaller output feature map size. Padding is utilized to preserve the spatial dimensions of the input data as it traverses through the convolutional layer. By adjusting these parameters, the behavior and effectiveness of the convolutional layer can be significantly impacted.

To aid in the expression of complex features, the convolutional layer includes incentive functions. These are expressed as follows:

$$A_{i,j,k}^{l} = f(Z_{i,j,k}^{l}) \tag{2}$$

In this experiment, this paper used the ReLU function as the excitation function:

$$ReLU(X) = Max(0, X) \tag{3}$$

Pooling layer. There exist various types of pooling techniques, including random pooling, mixture pooling, spectrum pooling, max pooling, mean pooling, among others. Given the utilization of max pooling in the experiments, focus will be directed towards maximum pooling.

Max Pooling serves as a subsampling technique within Convolutional Neural Networks (CNNs). Its primary function lies in reducing the dimensionality of feature maps, thereby mitigating computational burden, simplifying model complexity, and enhancing model robustness. Operationally, it entails selecting the maximum value within a local receptive field as the output to preserve salient features.

The principal objective of maximum pooling revolves around parameter reduction during neural network training while retaining essential features, thereby accelerating training duration. In practice, the pooling effect can be manipulated by adjusting parameters such as the size of the pooling window (kernel_size), stride, padding, among others. For instance, when employing a pool window size of 2x2 and a stride length of 2, both the height and width of the feature map are halved, while the channel count remains unaltered.

$$y_{m,n}^{d} = \max_{i \in R_{m \times n}^{d}} x_i \tag{4}$$

Where x_i is the activity value of each neuron in the region.

Fully connected layer. The fully connected layer, also referred to as the dense layer, serves as the final component of the hidden layers within a convolutional neural network. Its primary function is to propagate signals to other fully connected layers. Within this layer, the spatial topology of the feature map is lost as it is flattened into a vector and then passed through an activation function for further processing.

In a convolutional neural network, features are extracted from the input data by the convolutional and pooling layers, and the fully connected layer integrates these features in a nonlinear way to produce the output. Essentially, the fully connected layer does not have its own feature extraction capabilities but instead leverages higher-order features extracted by the preceding layers to achieve the learning objectives.

Output layer. The fully connected layer preceding the output layer in a CNN follows the same structural and operational principles as the output layer in a standard feedforward neural network. In the specific context of image classification, the output layer utilizes logical or normalized exponential functions, such as the softmax function, to generate classification labels. In the experiment, we employ the softmax function to produce category labels for the classification task.

$$Softmax(x) = \frac{e^{x}}{\sum_i e^{x_i}}, x \in R^{n} \tag{5}$$

Where n represents the number of categories.

In this experiment, the CNN structure is shown in figure 1.

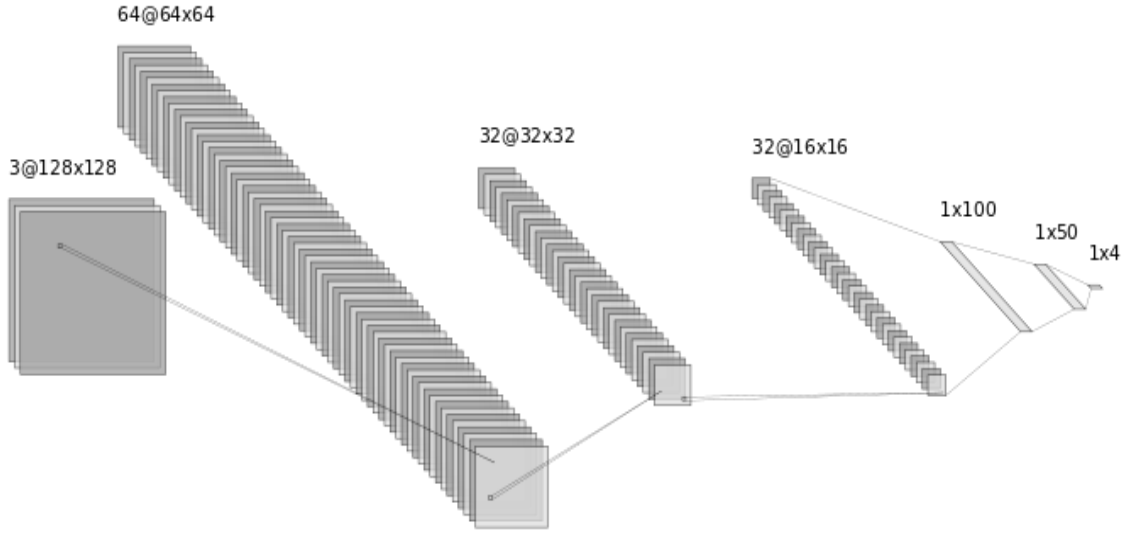


Figure 1. Schematic diagram of CNN model structure (Photo/Picture credit : Original)

Loss function. In machine learning and deep learning, the Cross-Entropy Loss function is frequently used, particularly for classification tasks. In order to reduce errors and increase accuracy, it is especially useful for assessing performance and directing the learning process in multi-class classification scenarios. This is because it measures the discrepancy between the model's output and the actual target label.

$$L = \frac{1}{N}\sum_{i} L_i = -\frac{1}{N}\sum_{i}\sum_{C=1}^{M} y_{ic}\log(p_{ic}) \tag{6}$$

Where M is the number of classes, y_{ic} is the symbolic function (0 or 1), if the true class of sample i is equal to c take 1, otherwise take 0, p_{ic} is the prediction probability that observed sample i belongs to class c.

The cross entropy loss function was selected for this study due to its strong performance in multi-class classification problems and adherence to the maximum likelihood estimation principle. Additionally, it can effectively address the gradient disappearance issue by measuring the difference between the real label and the model prediction directly.

3.2. SMOTE

SMOTE functions as a data preprocessing technique commonly employed for unbalanced datasets, addressing the class imbalance issue by generating new data through linear interpolation between a few class samples and their K similarities, thus equalizing the number of minority and multiple class samples. In this experiment, an imbalanced image dataset was captured to evaluate SMOTE's impact on gesture recognition. The approach involved computing the vector difference between a given sample's feature vector and its nearest feature point, multiplying this difference by a random number between 0 and 1, and adding it back to the feature vector. This procedure effectively selects a random point along the line segment connecting two specific feature vectors, yielding an example that encapsulates the characteristics of the original sample and its closest feature points.

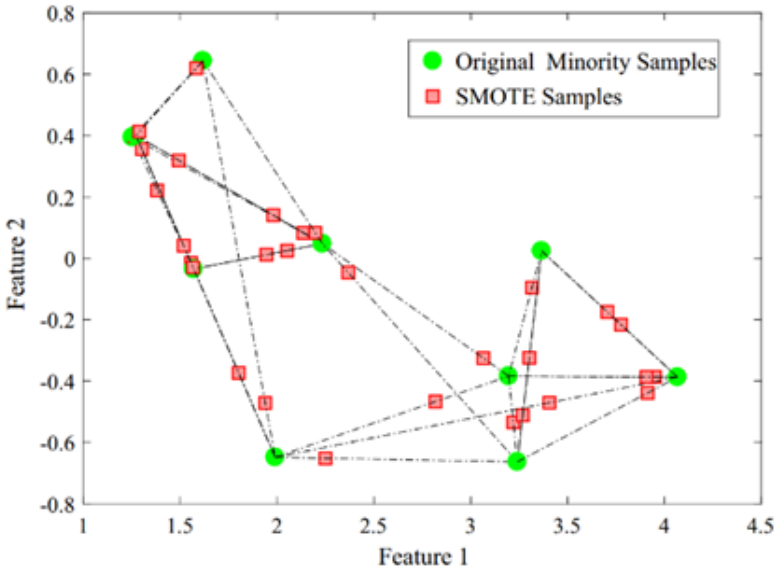


Figure 2. SMOTE generation mechanism [10]

$$P_Z(z) = (N-K)\binom{N-1}{K}\int_x P_x(x)\int_{r=\|z-x\|}^{\infty} P_x\left(x+\frac{(z-x)r}{\|z-x\|}\right)\left(\frac{r^{d-2}}{\|z-x\|^{d-1}}\right)\times B\left(1-I_{B(x,r)};N-K-1,K\right)drdx \tag{7}$$

Dina Elreddy et al. [10] formulated the theoretical analysis of SMOTE density, establishing the mathematical groundwork for our experiments involving SMOTE (Figure 2). The objective of employing the SMOTE technique is to rectify discrepancies in sample sizes among different classes by generating synthetic samples, thereby augmenting the sample space of underrepresented classes to render the dataset more comprehensive and conducive to improved and smoother training. In this

experiment, the SMOTE method demonstrates effectiveness; however, notable differences in data fluctuation between SMOTE and non-SMOTE methods are observed. It can be inferred that the SMOTE approach yields favorable outcomes in the gesture recognition project dealing with unbalanced datasets.

3.3. Mediapipe

Mediapipe is a powerful open-source cross-platform framework providing a rich set of tools and libraries for developing various vision- and perception-based sensing programs, including gesture recognition, face detection, and face tracking. Due to its real-time recognition capability and efficient processing of large-scale multimedia data, it is highly suitable for scenarios requiring real-time feedback and interaction. Consequently, it was applied in both the training and prediction phases of this experiment, demonstrating excellent performance. A comparison between the utilization of Mediapipe and its absence revealed significant improvements in real-time gesture prediction, highlighting its beneficial impact on the experiment's outcomes.

4. Experiment

4.1. Picture processing

In image processing, we use a variety of different preprocessing methods, such as corrosion, expansion, grayscale processing, Gaussian filtering, etc., which help our images become more consistent and convenient for subsequent reading and training. A good data set is often the key to the success of the experiment. Although the data set is not perfect, it also plays a role in the experiment.

4.2. Estimate

In order to analyze the data collected in the training more intuitively and adjust various parameters in time, this paper drew the cross loss curve and accuracy curve with the change of training times, and compared the influence of smote and smote method on the cross loss curve and accuracy curve of the training data. The following four pictures are as follows: Are the results of the training with and without smote method (Figure 3, figure 4, figure 5 and figure 6).

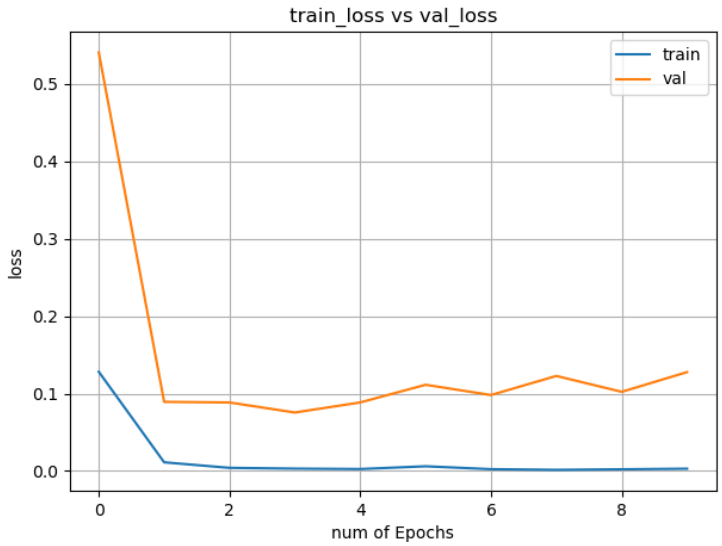


Figure 3. CrossEntropyLoss curve with smote (Photo/Picture credit : Original)

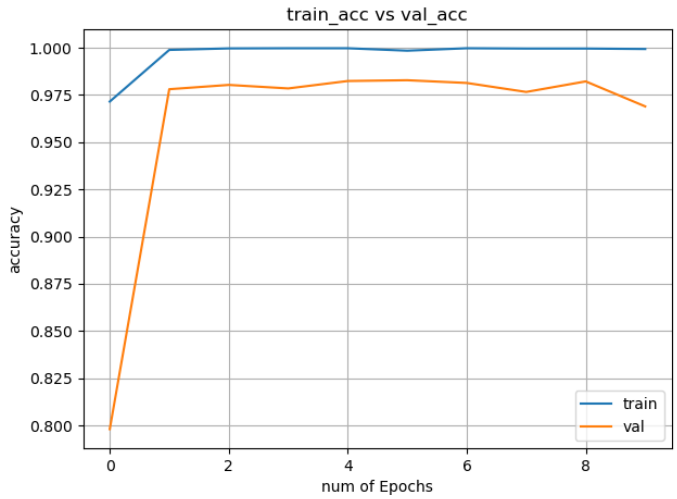


Figure 4. Accuracy curve with smote (Photo/Picture credit : Original)

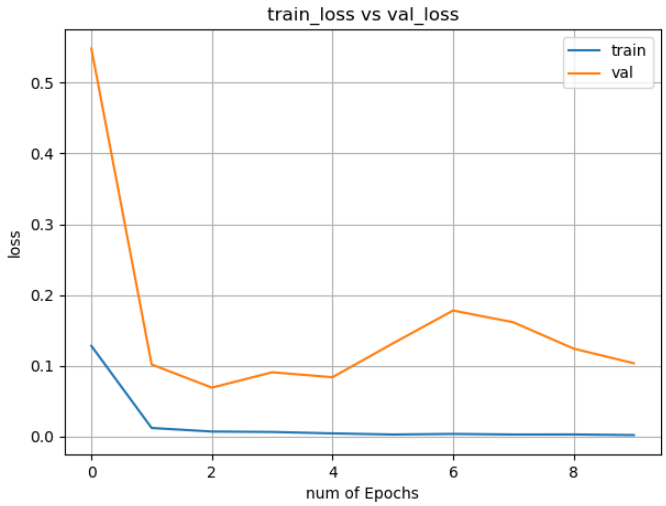


Figure 5. CrossEntropyLoss curve without smote (Photo/Picture credit : Original)

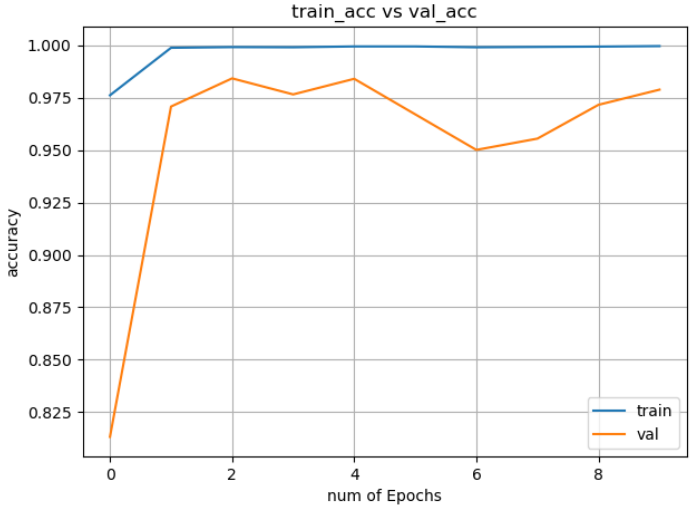


Figure 6. Accuracy curve without smote (Photo/Picture credit : Original)

After many comparisons, this paper found that the model performed best when the epoch was trained 10 times. It is not difficult to see from the figure that the model has been hovering near 100% accuracy until the last time when it is infinitely close to 100% accuracy, indicating that the model is effective, stable, and no fitting phenomenon has occurred. Comparing the other two line charts without smote, it is not difficult to find that the fitting efficiency of the loss function without smote was slower than that with smote. The loss rate did not drop to near 0 until the fifth training, while the

loss function with smote fell to 0 the second time. At the same time, the VAL loss and VAL accuracy with smote will be more stable than without smote. When the training frequency increases, the VAL accuracy and VAL loss of the model with smote become stable and slow, while the VAL accuracy and VAL loss of the model without smote have a large fluctuation, which also proves that smote enhances the stability of the model.

4.3. Confusion matrix

The confusion matrix is a table used to evaluate the performance of a classification model, often used in supervised learning problems. It compares the model's predictions with real labels to show the model's classification accuracy across different categories. It is not difficult to see from Figure 7 that the model performs well in classification.

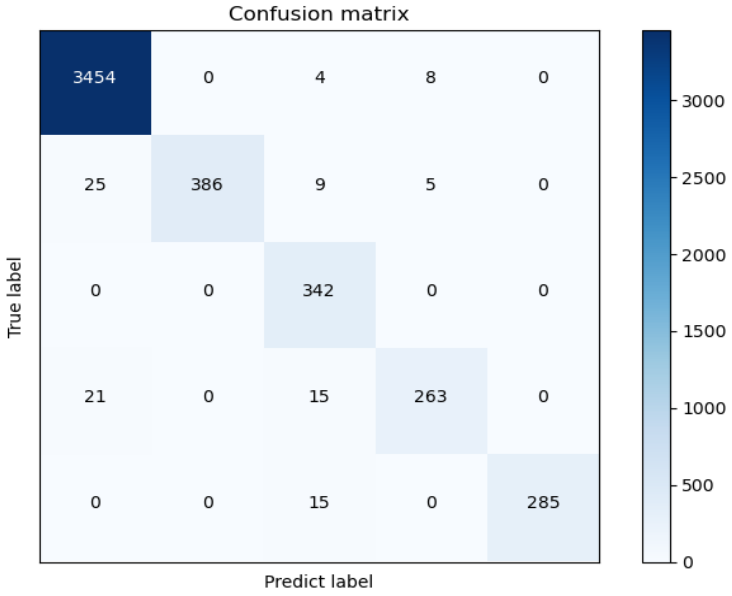


Figure 7. Confusion matrix (Photo/Picture credit : Original)

It is not difficult to see from the chart that with the addition of smote, even the lowest group had an accuracy of over 88%, and the highest had a accuracy of 100%. The most striking effect was that the group with the fewest data in the model was more than 98% accurate.

4.4. Accuracy, recall, F1, precision

We used the macro average to evaluate the precision, recall and F1 values of the whole model, and obtained the results as follows: the average accuracy is 98.8%, the average recall is 96.4%, the average precision is 92.9%, and the average F1 value is 94.5%. This shows that the model has a strong ability to identify various categories, does not miss too many positive samples, and the accuracy of predicting positive samples is also high. This shows that the items are, well, able to effectively distinguish between different categories and accurately classify samples.

4.5. Kappa

In addition, this paper will also use Kappa values to show the excellent consistency of the model. The closer the Kappa value is to 1, the higher the consistency.

$$Kappa = \frac{p_0 - p_e}{1 - p_e} \tag{8}$$

Where p_0 is the overall classification accuracy of the model and pe is the expected accuracy of the model. According to the calculation, it can be concluded that the Kappa value of the model is 0.95, which is a very high value, indicating that there is a high consistency between the classification results of the model and the actual results under the consideration of random classification, indicating that the model has excellent performance.

5. Conclusion

Today, there continues to be a significant population of individuals who are deaf worldwide, and the global issue of facilitating barrier-free communication for this community persists. This study aims to investigate the application of convolutional neural networks and the mediapipe library for real-time gesture recognition, addressing the challenge of imbalanced databases through reference to the SMOTE method. The results demonstrate that SMOTE performed effectively in significantly enhancing the experimental data, while mediapipe also made substantial contributions. Furthermore, compared to not utilizing the mediapipe library, employing the SMOTE method led to a notable improvement in forecasting accuracy.

The experimental results demonstrate that the proposed method effectively recognizes the gestures of deaf-mutes in real time and offers a viable solution to enhance their communication. This research contributes to the advancement of assistive technologies for individuals with disabilities and has potential for further optimization and application in practical settings. Subsequent research efforts could concentrate on expanding gesture recognition databases, enhancing recognition accuracy, and developing user-friendly applications tailored for deaf individuals. This research is anticipated to significantly enhance the quality of life for individuals who are deaf and promote their inclusion in society. Subsequent research could concentrate on enhancing the real-time accuracy of gesture recognition systems to ensure effective support for this population. Additionally, more advanced features such as custom gesture recognition and gesture control could be explored to offer more personalized communication methods. Alternatively, integrating algorithms with hardware devices, such as wearables or smartphone apps, may facilitate wider adoption and ease of use. Through ongoing improvement and innovation, this research has the potential to become a practical tool for individuals who are deaf, enabling them to integrate into society and have an equitable communication experience with others. In-depth research and interdisciplinary collaboration can advance the field and provide greater support and opportunities for people with disabilities.

References

[1] Asadi-Aghbolaghi M, Clapés A, Bellantonio M, et al. Deep learning for action and gesture recognition in image sequences: A survey. Gesture Recognition, 539-578 (2017).

[2] Ferreira P M, Cardoso J S, Rebelo A. On the role of multimodal learning in the recognition of sign language. Multimedia Tools and Applications, 78 (8): 10035-1005 (2009).

[3] Guo X L, Yang T T. Gesture recognition based on HMM-FNN model using a Kinect. Journal on Multimodal User Interfaces, 11: 1-7 (2017).

[4] Peng Y, Lin M, Chen Y, et al. Personal Touch: Improving touchscreen usability by personalizing accessibility settings based on individual user's touchscreen interaction, 2019 CHI Conference on Human Factors in Computing Systems, 1-11 (2019).

[5] Oberweger M, Wohlhart P, Lepetit V. Hands deep in deep learning for hand pose estimation. arXiv preprint arXiv:1502.06807 (2015).

[6] Rastgoo R, Kiani K, Escalera S. Multi-modal deep hand sign language recognition in still images using restricted Boltzmann machine. Entropy, 20 (11): 809 (2018).

[7] Zhang Q, Yang M, Zheng Q, et al. Segmentation of hand gesture based on dark channel prior in projector-camera system, 2017 IEEE/CIC International Conference on Communications in China (ICCC). IEEE, 1-6 (2017).

[8] Wadhawan A, Kumar P. Deep learning-based sign language recognition system for static signs. Neural computing and applications, 32 (12): 7957-7968 (2020).

[9] Bilaniuk L T, Atlas S W, Zimmerman R A. Magnetic resonance imaging of the orbit. Radiologic Clinics of North America, 25 (3): 509-528 (1987).

[10] Elreedy D, Atiya A F, Kamalov F. A theoretical distribution analysis of synthetic minority oversampling technique (SMOTE) for imbalanced learning. Machine Learning, 1-21 (2023)