INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact Factor: 6.078
(Volume 10, Issue 1 - V10I1-1198)
Available online at: https://www.ijariit.com

A Comprehensive Study of Sign Language Recognition using Technology and AI

Hakik PACI
hpaci@fti.edu.al
Polytechnic University of Tirana, Tirana, Albania

Evis Trandafili
etradnafili@fti.edu.al
Polytechnic University of Tirana, Tirana, Albania

Nelda Kote
nkote@fti.edu.al
Polytechnic University of Tirana, Tirana, Albania

Egisa Fusha
egisa.fusha@fti.edu.al
Polytechnic University of Tirana, Tirana, Albania

ABSTRACT

Sign language plays a crucial role in facilitating communication within the hearing-impaired community, yet it often poses a challenge as a communication barrier with the broader society. Recent technological and Artificial Intelligence (AI) advancements present an opportunity to bridge this communication gap effectively. This research delves into the essential components of developing a Sign Language Recognition system, exploring aspects such as sign-capturing techniques, selection of sign datasets, preprocessing methods, and integrating a deep learning module featuring both CNN and CNN-SVM modules.

Keywords: *Sign Language Recognition, Deep Learning, CNN, CNN-SVN*

I. INTRODUCTION

Communication is vital for human interaction, enabling the sharing of thoughts, feelings, and ideas. Sign language is a visual language that utilizes hand gestures, facial expressions, and movements which serves as a unique and expressive form of communication. Beyond being a tool for the deaf or hard of hearing, it represents a diverse cultural phenomenon, with over 300 sign languages globally spoken by over 70 million people facing hearing challenges [1]. However, the integration of hearing-impaired individuals into society poses challenges in areas like education, employment, healthcare, and transportation.

Sign language recognition is a challenging task due to the complexity and variability of sign language. However, machine learning techniques have shown promising results in this field. A complete sign language system comprises three main modules: the input capturing approach, the sign language dataset and the machine learning framework. The most common ways for sign capturing are vision-based approaches [3] where cameras are used to capture signer's hand gestures; data-glove-based approaches [4] that use gloves equipped with sensors to capture the signer's hand movements; and depth-based approaches which use depth sensors [5,6] such as Microsoft Kinect to capture the signer's hand gestures. The captured data should then be processed using computer vision and machine learning techniques to recognize the signs.

In this paper we engage the publicly available American Sign Language alphabet image dataset [7] to address a multi-class problem with 24 letter classes and explore two deep learning models, the CNN model and the CNN-SVM hybrid model, for sign language recognition. The CNN model [8] serves as a feature extractor and classifier, while the hybrid model combines CNN

architecture for feature extraction with SVM for classification [9]. We compare and evaluate the performance of these models using a real-world dataset, emphasizing the importance of integrating machine learning in addressing complex classification issues.

The methodology, experimental setup, dataset, preprocessing steps, architectures, training/testing procedures, and evaluation metrics are detailed in the section below and finally the study's results are reported and discussed in the context of comparing the two models.

II. ENVIRONMENT AND DATASET PREPARATION

For the objectives of this study, we used the Jupyter Notebook environment with python language and several libraries like: NumPy, Pandas, Matplotlib, Seaborn, Scikit-Learn and Keras. These libraries enhance efficient data processing, analysis, and model development.

The dataset employed is the American Sign Language alphabet dataset [7], accessible on Kaggle. The selection of the dataset was based on three criteria: grayscale images, abundance of data, and community references. The dataset features training and test data, with labels ranging from 0 to 25 corresponding to letters A to Y, excluding J and Z due to their movement-based nature. The dataset size, comprising 27,455 training cases and 7,172 test cases, is approximately half that of the MNIST benchmark.

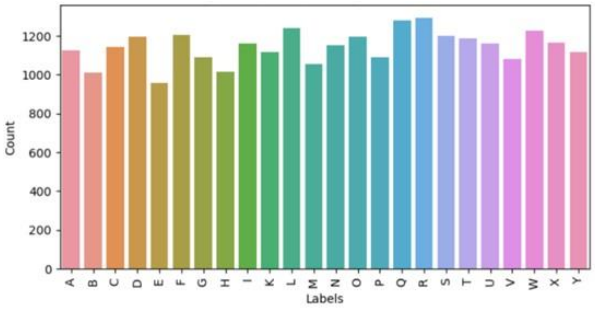


Figure 1. Visualizing Data Distribution: Insights from Histogram Analysis

Each data entry includes a label (0-25) and pixel values (pixel1 to pixel784), representing a 28x28 grayscale image with intensity values from 0 to 255. The chart in Figure 1 indicates a generally balanced dataset, a crucial factor in preventing bias results in data analysis and ML models. During the initial preprocessing step, we separated label columns from input feature columns (images), facilitating the model's learning process and applied normalization to bring pixel values to a consistent interval (0 to 1) to prevent dominance by specific pixel values, aid convergence and mitigate overfitting. Transform parameters like rotation range, zoom range, and flip settings were controlled using the "ImageDataGenerator" class in Keras. Finally, we converted categorical labels into binary vectors, as it is a necessity for loss functions like "categorical cross-entropy," employed in our task. This encoding allowed us to model and treat each class independently, avoiding ordinal relationships between categories.

III. EXPERIMENTAL DESIGN

The problem of Sign Language Recognition is modeled as a classification problem where the model predicts the correct label for the input data. The model undergoes training with American Sign Language alphabet dataset and is evaluated using a part of the dataset for testing, allowing it to make predictions on new, unknown data.

We employed a CNN model to serve as both a feature extractor and classifier. Feature extraction is achieved through the convolutional layer and the pooling layer. The convolutional layer serves to extract features from the image by performing a mathematical operation called convolution [10] while the pooling layer serves to reduce the number of parameters and computation in the network, and to make the network more robust to variations in the position of the features in the image [11]. In our task, we employed multiple convolutional and pooling layers to enhance the extraction of meaningful data. We incorporated additional techniques, including batch normalization and dropout, to further optimize performance [12]. In the neural network's final stage, we use a fully connected layer with ReLU and SoftMax as activation functions. The model was compiled and trained using the Adam optimizer.

For multi-class classification, the "categorical_crossentropy" loss function indicates the disparity between predicted and actual labels. The "epochs" parameter, indicates the dataset's complete passes through the model. Finding the right balance avoids underfitting or overfitting, ensuring optimal generalization and performance.

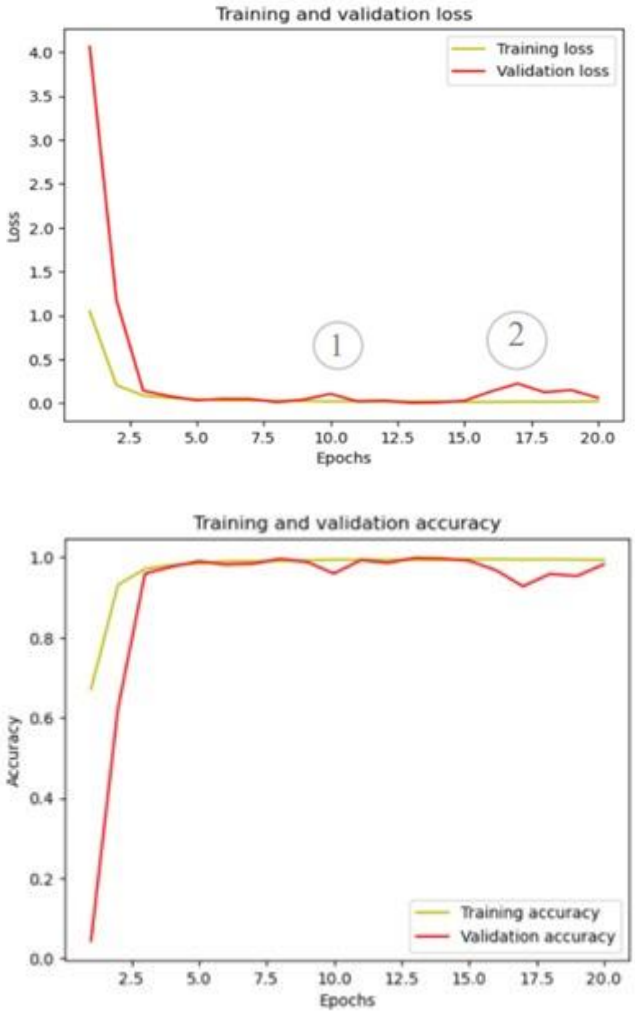


Figure 2: Training and validation loss and accuracy

The graph in Figure 2, depicts training loss (yellow) and validation loss (red) across epochs. Training loss averages errors in each iteration during training, while validation loss does so during testing. Ideally, both should exhibit a decreasing trend, as seen in the chart. However, instances (1) and (2) reveal overfitting, where the model memorizes training data excessively, leading to poor performance on test data. Accuracy is used to determine the overall model performance. In the graph, the aim is an upward trend for both training and testing gradients. Minor fluctuations are noted in the validation gradient for the same reasons as discussed earlier.

Repeating the experiments performed above, we replaced the dense layer with Support Vector Machine algorithm. A CNN-SVM hybrid model leverages the advantages of both CNN and SVM, such as the ability to learn complex features from images and the ability to handle imbalanced and noisy data. The final features extracted serve as inputs for SVM training and testing.

IV. MODEL EVALUATION

To assess the efficiency and resource allocation of each model, training time serves as a key indicator. It sheds light on how models handle large datasets and their effectiveness in time-critical scenarios. Below are the training times for both models, the CNN model, and the hybrid CNN–SVM model.

```
Training Time: 1451.987488269806

Training Time: 1360.0498116016388
```

The CNN model, serving as both a feature extractor and classifier, exhibits a longer training time compared to the CNN–SVM hybrid model. This outcome can be attributed to the following factors:

Model Architecture: The CNN model, with its multiple convolutional layers and pooling for feature extraction, alongside a dense layer for classification, is more intricate. In contrast, the hybrid model adopts a simpler architecture with fewer layers. The

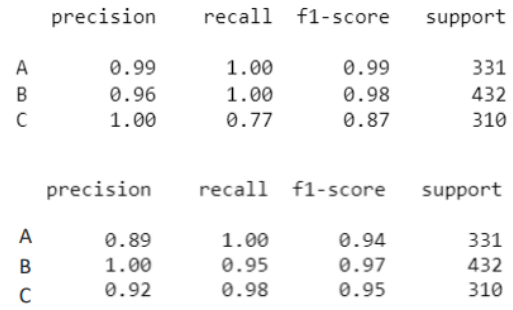
complexity of the architecture contributes to an extended training duration, especially given the CNN model's greater number of parameters.

Hyperparameters: The CNN model involves more hyperparameters, including the number of epochs, impacting convergence speed and overall training time.

Table 1 : Results from confusion matrices

	CNN model as feature extractor and classifier	CNN – SVM hybrid model
Total number of correct predictions	6345	5652
Total number of incorrect predictions	827	1520

Analysis of confusion matrices reveals that the CNN model correctly predicted 6345 labels, surpassing the hybrid model's prediction of 5652 labels. Despite its complex architecture, the CNN model, tailored for in-depth feature and pattern learning, demonstrates superior performance in the data classification task by accurately predicting a larger number of labels.

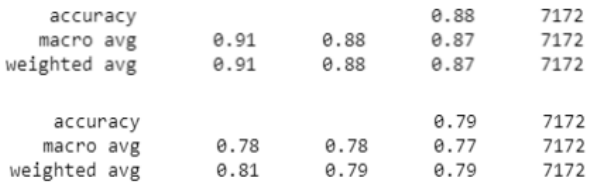


	precision	recall	f1-score	support
A	0.99	1.00	0.99	331
B	0.96	1.00	0.98	432
C	1.00	0.77	0.87	310

	precision	recall	f1-score	support
A	0.89	1.00	0.94	331
B	1.00	0.95	0.97	432
C	0.92	0.98	0.95	310

Figure 4: Evaluation of comparative parameters between two models

Observing the precision, recall, and F1-score for three labels, disparities are evident. When the F1-score attains a higher value, it signifies a superior equilibrium between precision and recall, with the maximum achievable value being 1. Specifically considering label A, the CNN model exhibits elevated precision, indicating its adeptness at accurately identifying instances for label A in comparison to the hybrid model.



accuracy			0.88	7172
macro avg	0.91	0.88	0.87	7172
weighted avg	0.91	0.88	0.87	7172

accuracy			0.79	7172
macro avg	0.78	0.78	0.77	7172
weighted avg	0.81	0.79	0.79	7172

Figure 5: Accuracy score

In our analysis, the discrepancy in accuracy between the two models is not substantial. Notably, the CNN model outperforms the hybrid model in terms of accuracy for the dataset under consideration. Thus, for the specific dataset utilized in this study, the CNN model, serving both as a feature extractor and classifier, proves superior in addressing classification tasks.

V. CONCLUSIONS

Deep learning has played a significant role in advancements in sign language recognition. In this paper we model the Sign language recognition problem as a multi-class classification problem and develop an accurate system capable of recognizing sign language gestures. The evaluation of input modalities is omitted from the study and a publicly available image dataset for alphabets in American Sign Language is employed. A comparative study of two deep learning algorithms CNN and CNN-SVM is performed. From the results presented, we noticed that the CNN model excelled in accuracy but demanded more computational

resources and training time, whereas the CNN–SVM hybrid model prioritized time efficiency over performance. The choice between performance and runtime depends on the problem's specific requirements. Applications like medical diagnosis prioritize accuracy, even with longer execution times, while real-time tasks, such as autonomous driving, demand faster processing. Future work involves fine-tuning hyperparameters and incorporating data preprocessing techniques. While hold-out cross-validation was used, exploring k-folds cross-validation could enhance model evaluation.

In our analysis, the discrepancy in accuracy between the two models is not substantial. Notably, the CNN model outperforms the hybrid model in terms of accuracy for the dataset under consideration. Thus, for the specific dataset utilized in this study, the CNN model, serving both as a feature extractor and classifier, proves superior in addressing classification tasks.

VI. REFERENCES

[1] World Federation of the Deaf. “About Us.” World Federation of the Deaf, 2021. Retrieved Octobre, 2023, from http://wfdeaf.org/

[2] M. Ebrahim Al-Ahdal and M. T. Nooritawati, "Review in Sign Language Recognition Systems," 2012 IEEE Symposium on Computers & Informatics (ISCI), Penang, Malaysia, 2012, pp. 52-57, doi: 10.1109/ISCI.2012.6222666.

[3] S. Antony, K. B. V. Santhosh, N. Salimath, S. H. Tanmaya, Y. Ramyapriya and M. Suchith, "Sign Language Recognition using Sensor and Vision Based Approach," 2022 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI), Chennai, India, 2022, pp. 1-8, doi: 10.1109/ACCAI53970.2022.9752580.

[4] Z. R. Saeed, Z. B. Zainol, B. B. Zaidan and A. H. Alamoodi, "A Systematic Review on Systems-Based Sensory Gloves for Sign Language Pattern Recognition: An Update From 2017 to 2022," in IEEE Access, vol. 10, pp. 123358-123377, 2022, doi: 10.1109/ACCESS.2022.3219430.

[5] H. V. Verma, E. Aggarwal and S. Chandra, "Gesture recognition using kinect for sign language translation," 2013 IEEE Second International Conference on Image Information Processing (ICIIP-2013), Shimla, India, 2013, pp. 96-100, doi: 10.1109/ICIIP.2013.6707563.

[6] Agarwal and M. K. Thakur, "Sign language recognition using Microsoft Kinect," 2013 Sixth International Conference on Contemporary Computing (IC3), Noida, India, 2013, pp. 181-185, doi: 10.1109/IC3.2013.6612186.

[7] Grassknoted. (2018). Image data set for alphabets in the American Sign Language. Retrieved November 23, 2023, from Kaggle website: https://www.kaggle.com/datasets/grassknoted/asl-alphabet

[8] Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. arXiv preprint arXiv:1608.06993. https://arxiv.org/abs/1608.06993

[9] Agarap, A. F. (2017). An Architecture Combining Convolutional Neural Network (CNN) and Support Vector Machine (SVM) for Image Classification. arXiv preprint arXiv:1712.03541. https://arxiv.org/abs/1712.03541

[10] Rosebrock, A. (2021). Convolutional Neural Networks (CNNs) and Layer Types. PyImageSearch. Retrieved Octobre , 2023, from https://pyimagesearch.com/2021/05/14/convolutional-neural-networks-cnns-and-layer-types/

[11] GeeksforGeeks. (2023). CNN | Introduction to Pooling Layer. Retrieved Octobre, 2023, from https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/

[12] Holbrook, R., & Cook, A. (n.d.). Dropout and Batch Normalization. Retrieved November 23, 2023, from https://www.kaggle.com/ryanholbrook/dropout-and-batch-normalization

