Mugla Journal of Science and Technology

TURKISH SIGN LANGUAGE EXPRESSIONS RECOGNITION USING DEEP LEARNING AND LANDMARK DATA

Cumhur TORUN, Faculty of Engineering and Architecture, Kastamonu University, Turkey, cumhurtorun@gmail.com
(https://orcid.org/0009-0000-9984-1384)
Abdulkadir KARACI*, Faculty of Engineering and Natural Sciences, Samsun University, Turkey, akaraci@gmail.com
(https://orcid.org/0000-0002-2430-1372)

Received: 15.10.2024, Accepted: 17.11.2024
*Corresponding author
Research Article
DOI: 10.22531/muglajsci.1567197

Abstract

Sign language is a vital communication tool for hearing-impaired individuals to express their thoughts and emotions. Turkish Sign Language (TSL) is based on hand gestures, facial expressions, and body movements. In this study, deep learning models were developed to recognize 41 commonly used TSL expressions. An original dataset was created using the Media Pipe Holistic framework to capture the 3D landmarks of hand, face, and body movements. The study trained and evaluated GRU, LSTM, and Bi-LSTM models, as well as hybrid architectures such as CNN+GRU, GRU+LSTM, and GRU+Bi-LSTM. In the training of the models, a hold-out validation method was used. 80% of the dataset was allocated for training and 20% for testing. Additionally, 20% of the training data was used for validation. Among Deep Learning models, the CNN+GRU hybrid model achieved the highest accuracy rate of 96.72%, outperforming similar studies in the literature. Our results demonstrate that deep learning techniques can effectively classify TSL expressions, with the CNN+GRU combination showing particularly high performance. Future work will focus on expanding the dataset and developing real-time recognition systems that incorporate both skeleton images and landmarks.

Keywords: Turkish sign language, Sign language recognition, Media pipe, Deep learning, Recurrent neural network

DERİN ÖĞRENME VE YER İŞARETİ VERİLERİNİ KULLANARAK TÜRK İŞARET DİLİ İFADELERİNİ TANIMA

Özet

İşaret dili, işitme engelli bireylerin düşüncelerini ve duygularını ifade etmeleri için hayati bir iletişim aracıdır. Türk İşaret Dili (TİD), el hareketleri, yüz ifadeleri ve vücut hareketlerine dayanır. Bu çalışmada, yaygın olarak kullanılan 41 TİD ifadesini tanımak için derin öğrenme modelleri geliştirilmiştir. El, yüz ve vücut hareketlerinin 3D yer işaretlerini yakalamak için Media Pipe Holistic çerçevesi kullanılarak orijinal bir veri seti oluşturulmuştur. Çalışmada, GRU, LSTM, Bi-LSTM modelleri ve hibrit mimarilere sahip olan CNN+GRU, GRU+LSTM, GRU+Bi-LSTM modelleri eğitilmiş ve değerlendirilmiştir. Modellerin eğitiminde dışarda tutma doğrulama yöntemi kullanılmıştır. Veri setinin %80'i eğitim ve %20'si test için ayrılmıştır. Ayrıca eğitim için ayrılan verinin %20'si doğrulama için kullanılmıştır. Derin öğrenme modelleri arasında en yüksek doğruluk oranını %96,72 ile CNN+GRU hibrit modeli elde etmiştir ve literatürdeki benzer çalışmalardan daha yüksek performans göstermiştir. Sonuçlarımız, derin öğrenme tekniklerinin TİD ifadelerini etkili bir şekilde sınıflandırabileceğini ortaya koymaktadır ve özellikle CNN+GRU kombinasyonu yüksek performans sağlamıştır. Gelecek çalışmalar, veri setinin genişletilmesi ve iskelet görüntüleriyle birlikte yer işaretlerinin de kullanıldığı gerçek zamanlı tanıma sistemlerinin geliştirilmesine odaklanacaktır.

Anahtar Kelimeler: Türk işaret dili, İşaret dili tanıma, Media pipe, Derin öğrenme, Tekrarlayan sinir ağı

Cite

Torun, C., Karaci, A., (2024). "Turkish Sign Language Expressions Recognition Using Deep Learning and Landmark Data", Mugla Journal of Science and Technology, 10(2), 52-58.

1. Introduction

Sign language (SL) is an essential communication tool for people who are deaf or hard of hearing, enabling them to convey their emotions, ideas and needs to others. This form of nonverbal communication relies on using hands, arms, heads, facial expressions, and body language. The World Health Organization reported in 2021 that approximately 466 million people globally experience hearing loss, including 34 million children. Projections indicate that this figure may rise to over 700 million by 2050 [1]. Individuals with hearing impairments utilize Sign Language to communicate, not just among themselves but also with others. Sign Languages function similarly to spoken languages, varying from one country to another and even possessing different dialects within

a single country. This diversity poses a significant challenge in developing a universal tool for all hearing-impaired individuals globally [2].

One of the sign languages, Turkish Sign Language (TSL), is typified by hand gestures and occasionally face and body indications. It has unwritten grammar [3]. In TSL, gestures represent commonly used words and the letters of the Turkish alphabet. Although Sign Language is influenced by spoken language, it possesses its unique grammatical structure. The relationship between TSL and Turkish cannot be taken for granted [4].

This study focuses on TSL expression recognition. There are a few studies in literature that stand out regarding TSL. Katılmış and Karakuzu[3], created a dataset using a Leap Motion Controller (LMC) for 26 dynamic Turkish Sign Language (TSL) words performed with both hands. They applied the Meta Extreme Learning Machine (MELM) method to this dataset and conducted training and testing using 5-fold cross-validation. Pacal and Alaftekin[5], classified numbers (0-9) in TSL using ResNet, MobileNet, VGG, DenseNet, and EfficientNet models. Özcan and Baştürk [6], created a dataset consisting of videos recorded from various angles by 49 individuals with hearing and speech impairments. The videos featured 25 words commonly used in hospital emergency departments. For classification tasks, a transfer learning model based on GoogLeNet was employed. Kırcı et al. [7], used Media Pipe to detect hand landmarks and created a dataset for 29 TSL letters. They trained a Long Short-Term Memory (LSTM) network using this dataset. However, no information regarding the recognition performance obtained in the study was provided. Karacı et al. [4], developed machine learning models based on a cascade voting approach to recognize the 29-letter TSL alphabet. They used a dataset obtained from the Leap Motion Controller (LMC) device, from which they extracted handcrafted features. The models they developed were tested in real-time and achieved high accuracy rates. Çelik and Odabaş [8] developed a model that translates sign language gestures into text by combining CNN and LSTM networks to automate the TSL recognition process. In this context, they used a camera image dataset consisting of 10 digits and 29 letters. Demircioğlu and Bülbül [9] aimed to recognize the basic hand movements of TSL using the LMC device. In their study, they developed a real-time system based on 18 selected hand movements from TSL. To evaluate the system's performance, they used machine learning methods such as Random Forest and Multilayer Perceptron (MLP), comparing the results. The system was noted for its advantage in achieving high accuracy, especially with a small amount of data. Haberdar and Albayrak [10] prepared a dataset of 50 static TSL words using a Leap Motion device. They trained a model on this dataset using Hidden Markov Models (HMM). Memiş and Albayrak [11], created a dataset of 111 static TSL words using an MS Kinect device. They trained a k-NN algorithm on this dataset using 3-fold cross-validation.

In this study, Deep Learning (DL) based sign language recognition models were developed for sign language expressions frequently used in TSL. These models were trained and tested on the original dataset created in this study. The main contributions of this paper are as follows:

1. The authors developed an original dataset of TSL expressions that has not been used in previous studies. This data set is expected to be useful for future research in the field of TSL recognition.

2. Since the dataset consists of landmarks, it can be processed by DL models using fewer resources.

3. CNN (Convolutional Neural Network), GRU (Gated Recurrent Unit), LSTM and Bi-LSTM (Bidirectional Long Short-Term Memory) DL models and hybrid models (CNN+GRU) by connecting these models, GRU+Bi-LSTM and GRU+LSTM) were created.

4. The hybrid model CNN+GRU achieved an impressive accuracy rate of 96.72%, leaving behind similar studies in literature and demonstrating the effectiveness of convolutional and recurrent neural networks for sign language recognition.

2. Materials and Methods

An overview of our methodology is shown in Figure 1. In this study, a dataset was created using the Media Pipe framework for 41 commonly used sign language expressions in Turkish Sign Language (TSL). Landmark points were obtained from video frames using Media Pipe and saved. The dataset was normalized before being fed into the deep learning (DL) models. GRU, LSTM, and Bi-LSTM models were trained and tested on this dataset. Additionally, training and testing procedures were also performed on hybrid models such as CNN+GRU, GRU+Bi-LSTM, and GRU+LSTM.

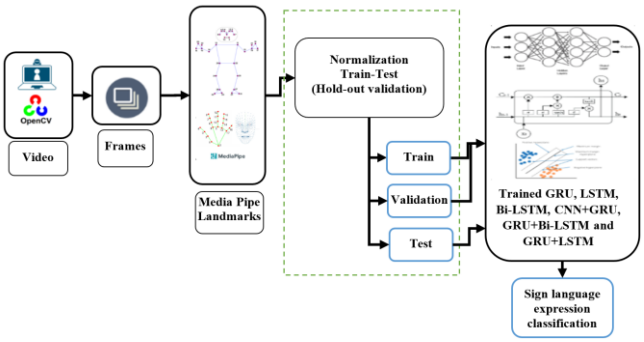

Figure 1. Overall methodology of the research.

2.1. Dataset

In all sign languages, including Turkish Sign Language (TSL), words are divided into two main categories. These are static sign language words and dynamic sign language words. Static sign language words are signs made with the hands remaining in a fixed position or with very little movement. Dynamic sign language words involve hand or body movement, and changes in hand positions or shapes. These signs generally consist of a combination of several movements and are more complex [12].

In this study, 41 words were selected from the basic communication patterns in units 1 and 2 of the Turkish Sign Language Course Teaching Material book published by the Ministry of National Education, General Directorate of Special Education and Guidance. Of these words, 35 are dynamic, and 6 are static. These words, their types, and the labels to be used while training DL algorithms are presented in Table 1.

Table 1. TSL words were used in the creation of the dataset.

Labels	Words	Type	Labels	Words	Type
0	Man (Adam)	Dynamic	**21**	Dog (Köpek)	Dynamic
1	Smart (Akıllı)	Dynamic	**22**	In the box (Kutuda)	Dynamic
2	Five (Beş)	Static	**23**	Hello (Merhaba)	Dynamic
3	White (Beyaz)	Dynamic	**24**	Why (Neden)	Dynamic
4	One (Bir)	Static	**25**	Cheerful (Neşeli)	Dynamic
5	Here (Burada)	Dynamic	**26**	Student (Öğrenci)	Dynamic
6	Grandpa (Dede)	Dynamic	**27**	Sorry (Özür dilerim)	Dynamic
7	Not (Değil)	Dynamic	**28**	Calm (Sakin)	Dynamic
8	Ice Cream Vendor (Dondurmacı)	Dynamic	**29**	Respect (Saygı)	Dynamic
9	Sir (Efendi)	Dynamic	**30**	Classroom (Sınıf)	Statik
10	Disabled (Engelli)	Static	**31**	In the classroom (Sınıfta)	Static
11	Goodbye (Güle güle)	Dynamic	**32**	Angry (Sinirli)	Dynamic
12	Good morning (Günaydın)	Dynamic	**33**	Black (Siyah)	Dynamic
13	Which (Hangisi)	Dynamic	**34**	Doesn't Know (Tanınmıyor)	Dynamic
14	Gift (Hediye)	Dynamic	**35**	Lazy (Tembel)	Statik
15	Excited (Heyecanlı)	Dynamic	**36**	Thanks (Teşekkürler)	Dynamic
16	Nice to be here (Hoşbulduk)	Dynamic	**37**	Doing (Yapıyor)	Dynamic
17	Welcome (Hoşgeldiniz)	Dynamic	**38**	To Do (Yapmak)	Dynamic
18	Cat (Kedi)	Dynamic	**39**	Naughty (Yaramaz)	Dynamic
19	Who (Kim)	Dynamic	**40**	Place (yer)	Dynamic
20	Girl (Kız)	Static			

The dataset was obtained using an application developed in Python with the Media Pipe Holistic framework. While creating the dataset, the person detected on the camera was cropped to center within a rectangular frame. The x, y, and z coordinates of 543 landmarks from the face, right hand, left hand, and body were recorded through the Media Pipe Holistic framework. Seven randomly selected words ("goodbye", " good morning", " nice to be here", " welcome", " hello", " why", and " sorry") were obtained from five individuals, while the remaining words were obtained from three individuals. This aims to examine whether the number of examples affects learning performance. Each person repeated the sign language word 10 times, and 30 frames were captured for each repetition. The total number of frames in the dataset is 41,100. However, feeding each set of 30 frames obtained for each trial as-is to machine learning is not appropriate. Therefore, during the pre-processing phase, these 30 frames were concatenated into a single frame. As a result, a total of 1,370 data points were obtained.

2.2. LSTM and GRU

LSTM is a subset of Recurrent Neural Network (RNN) architecture that captures and represents long-term dependencies in sequential input. Several gating mechanisms help to enable this capability. An LSTM unit consists of several key components: a cell state, an input gate, an output gate, and a forget gate. The cell structure stores information for long periods, while each of the three gates functions as a standard artificial neuron, similar to those seen in multi-layer or feedforward neural networks. Essentially, these gates calculate an activation based on a weighted sum [13].

LSTM networks, created in 1997 for language processing, became well-known for their outstanding ability to hold long short-term dependencies [14]. However, the complexity of its architecture sometimes results in extensive training cycles. To solve this, GRU networks were designed as a more efficient alternative to LSTM, with a simpler structure that speeds up the training process [15]. The GRU is similar to an LSTM in that it includes a forget gate, but it has fewer parameters since it does not include an output gate. [16]. The input and output structure of GRU is comparable to standard RNN, while its internal structure is similar to the LSTM. Figures 2 show an internal structural comparison of LSTM and GRU.

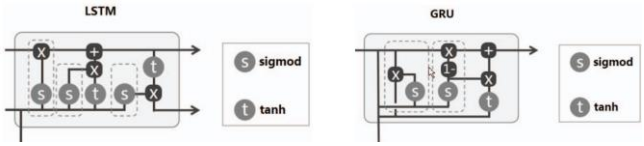

Figure 2. Internal structure of LSTM and GRU.

2.3. Bi-LSTM

The traditional LSTM processes previous data by only receiving inputs in the backward direction through hidden states. In contrast, the Bi-LSTM network differs from the unidirectional LSTM, which relies solely on backward propagation to gather preliminary insights on temporal data. The Bi-LSTM network trains LSTM models in both forward and backward directions simultaneously, enabling it to extract features from both past and future timeframes of the data [17]. This capability results in improved prediction outcomes compared to LSTM. Bi-LSTM typically outperforms LSTM, although it requires more time for training [16]. The overall structure of the Bi-LSTM is illustrated in Figure 3 [17].

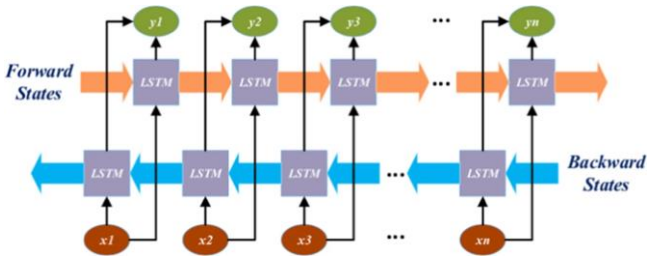

Figure 3. Architecture of the Bi-LSTM network.

2.4. Performance Metrics

The models were evaluated using metrics such as precision (P), recall (R), F1-score (F1), and accuracy (acc). These metrics have been commonly utilized in prior research and are widely recognized. These metrics are explained below [18].

Precision: This parameter evaluates the proportion of true positives among all predicted positives. Consequently, it is influenced by the values of True Positives (TP) and False Positives (FP).

$$\text{Precision(P)} = TP/(TP + FP) \quad (1)$$

Recall: Recall is the proportion of true positives that the model correctly classified. It is computed for all positive samples in the dataset.

$$\text{Recall(R)} = TP/(TP + FN) \quad (2)$$

F1-Score: The F1 score is a metric that combines both precision and recall, offering a comprehensive assessment of the model's accuracy. It is calculated by finding the harmonic mean of the precision and recall values.

$$\text{F1-Score} = 2*(P*R)/(P+R) \quad (3)$$

3. Experimental Results and Discussion

3.1. Performance Metrics

In this study, the GRU, LSTM, Bi-LSTM, CNN+GRU, GRU+LSTM, and GRU+Bi-LSTM models were trained and tested using Tensorflow and Keras Library in Python on Google Colab with Tesla P100-PCIE-16 GB GPU, Intel(R) Xeon(R) 2.30 GHz CPU, and 25 GB RAM system components. The models' training approach, as well as their parameters, are detailed below.

3.2. Training and testing of models

The GRU, LSTM, Bi-LSTM, CNN+GRU, GRU+LSTM, and GRU+Bi-LSTM models were trained and tested using the hold-out validation method. The dataset was initially split into 80% for training and 20% for testing. Additionally, 20% of the data allocated for training was used for validation.

Some parameters used in training the models and giving the highest classification performance are given in Table 2 and other parameters were used by default. Optimization refers to the learning algorithm responsible for determining how the vast number of parameters, often ranging from millions to billions, should be adjusted during the training process [19]. During the training of the models, various optimizers were tested, including Adam, Adadelta, SGD, RMSprop, Adamax, and Nadam. The highest classification accuracy for the LSTM model was achieved with the Adadelta optimizer, whereas for the other models, it was achieved using the Adam optimizer.

Another key factor in model training is the learning rate. The predictions may become unstable and fluctuate if the learning rate is too high. Conversely, the training process can take much longer if the learning rate is too low. Therefore, various learning rates were tested during the training phase, and the rate that delivered the best performance was selected. Additionally, the early stopping feature was used for each model. As a result, the number of epochs used during the training of the models varied from one another.

Table 2. Parameters setting of the GRU, LSTM, Bi-LSTM, CNN+GRU, GRU+LSTM, and GRU+Bi-LSTM models

Model	Optimization Algorithm	Learning Rate	Batch Size	Epoch	Activation Function	Loss Fuction
GRU	Adam	0.0001	64	284	Relu Softmax	Categorical crossentropy
CNN+GRU	Adam	0.0001	32	273		
GRU+LSTM	Adam	0.00001	32	311		
GRU+Bi-LSTM	Adam	0.00001	32	262		
LSTM	Adadelta	0.1	64	86		
Bi-LSTM	Adam	0.0001	32	123		

In the study, the highest classification accuracy was obtained in the CNN+GRU model. The architecture of this model is presented in Table 3. This model combines 1D convolutional layers and GRU layers to capture both spatial and temporal patterns in the input. The model starts with two 1D convolutional layers (Conv1D) with 64 and 128 filters and then uses MaxPooling layers to reduce the dimensionality. After each convolution layer, Batch Normalization is applied to stabilize the learning process and Dropout is added to prevent overfitting. After extracting 296,448 convolutional features, temporal dependencies in the sequential data are captured using two GRU layers, respectively. The second GRU layer passes its output to a fully connected layer. L2 regularization is applied to prevent over-learning in fully connected layers, and Batch Normalization is applied to ensure faster convergence of the model. Finally, the output layer has a softmax activation function to estimate the class probabilities among the 41 classes.

Table 3. Architecture of CNN+GRU model

Layer (type)	Output Shape	Param #
conv1d_20 (Conv1D)	(None, 28, 64)	319,168
max_pooling1d_20	(None, 14, 64)	0
batch_normalization_35	(None, 14, 64)	256
dropout_45	(None, 14, 64)	0
conv1d_21 (Conv1D)	None, 12, 128)	24,704
max_pooling1d_21	(None, 6, 128)	0
batch_normalization_36	None, 6, 128)	512
dropout_46	(None, 6, 128	0
gru_20 (GRU)	(None, 6, 256)	296,448
dropout_47	(None, 6, 256)	0
gru_21 (GRU)	(None, 128)	148,224
dropout_48	(None, 128)	0
dense_25 (Dense)	(None, 256)	33,024
batch_normalization_37	(None, 256)	1,024

dense_26	(None, 256)	65,792
batch_normalization_38	(None, 256)	1,024
dense_27 (Dense)	(None, 41)	10,537

The loss and accuracy curves of the CNN+GRU model are presented in Figure 4. The curves of the other models are not included to enhance the readability of the study. These curves indicate that the model did not overfit the data and performed well in terms of generalization. The model exhibited some oscillations up to 200 epochs, but after this epoch value, it demonstrated reduced oscillations and improved generalization.

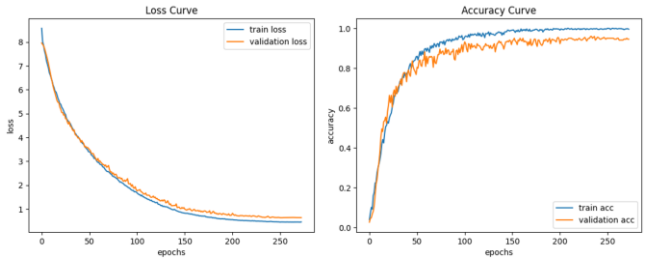

Figure 4. Accuracy and loss curves of the CNN+GRU model

3.3. Classification Results

The classification performances of the models are presented according to the recall (R), precision (P), F1 score (F1), and accuracy (ACC) measurements in Table 4. The R, P, and F1 values in Table 4 are calculated as weighted averages. As can be seen in this table, the highest classification performance was achieved in the hybrid model created by sequentially combining the CNN and GRU models.

The second highest classification performance was obtained in the other hybrid model, the GRU+Bi-LSTM model. The ACC and R values of the CNN+GRU model are 96.72%. This model classifies 41 sign language words correctly at a rate of 96.72%. This rate is 95.62%, 95.26%, 94.53%, 93.80%, and 92.34% in the GRU+Bi-LSTM, Bi-LSTM, GRU, GRU+LSTM, and LSTM models, respectively. Thus, three models demonstrate a classification accuracy of 95% or higher. This rate is high for a data set of 41 classes and the models classify sign language words sufficiently. In addition, the CNN+GRU and GRU-Bi-LSTM models have high precision(P) values. These values are 97.11% and 96.27%, respectively. This indicates that these models have a low likelihood of making false positive (FP) classifications.

Table 4. Hold-out test results of the GRU, LSTM, Bi-LSTM, CNN+GRU, GRU+LSTM, and GRU+Bi-LSTM models.

Models	R (%)	P (%)	F1 (%)	ACC (%)
CNN+GRU	96.72	97.11	96.69	96.72
GRU+Bi-LSTM	95.62	96.27	95.52	95.62
Bi-LSTM	95.26	95.87	95.13	95.26
GRU	94.53	95.24	94.53	94.53
GRU+LSTM	93.80	94.78	93.70	93.80
LSTM	92.34	93.47	92.24	92.34

The confusion matrix of the CNN+GRU model, which gives the highest classification performance, is presented in Figure 5. According to this matrix, 33 words were classified 100% correctly (R=1), while the remaining 8 words could not be classified 100% correctly. These words and their correct classification rates are as follows: smart (83.33%), here (83.33%), hello (80%), why (90%), calm (83.33%), classroom (83.33%), in the classroom (83.33%) and place (83.33%). Only the word "hello" has a false negative count of 2, while the others have a count of 1. The word "smart" is “doesn't know”, the "here" word is “excited”, the word "hello" is “welcome” and “black”, the word "why" is “dog”, the word "calm" is “doing”, the word "classroom" is “in the classroom”, the word "in the classroom" as “one” and finally the word “place" is misclassified as “in the box”. These words consist of 6 dynamic and 2 static words. No dynamic word has been misclassified as a static word, and no static word has been misclassified as a dynamic word. Furthermore, it has been observed that obtaining samples from more individuals does not proportionally improve model performance. This is because, among the 7 words sampled from 5 individuals, 2 were misclassified, while among the 34 words sampled from 3 individuals, 6 were misclassified.

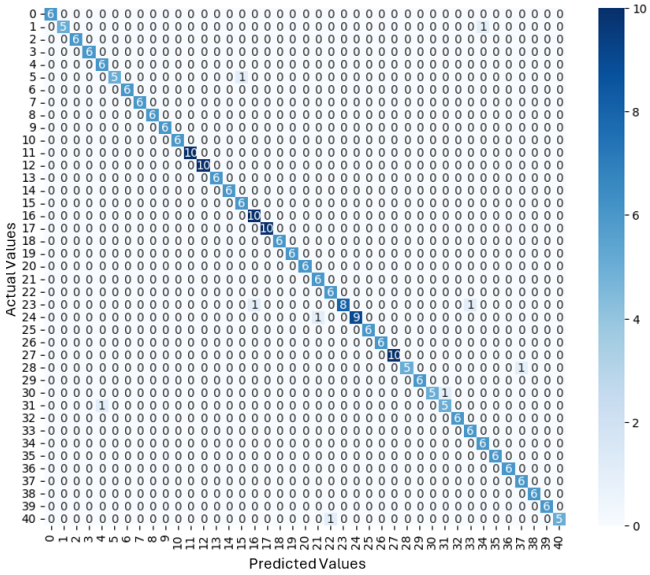

Figure 5. Confusion matrix of CNN+GRU model.

3.4. Discussion

In this study, the results obtained were compared only with studies conducted specifically on TSL to ensure fairness. The studies in the field of TSL are presented in Table 5. Among these studies, only five studies [3, 6, 9–11] are directly related to our work, as they focus on recognizing static or dynamic words. The classification accuracies of these studies are lower than those of our work, with values of 93%, 93.93%, 93.59%, 95.7%, and 90%, respectively. Additionally, two studies [4, 5] that are not directly related to our work demonstrate higher classification performance. The classification accuracy of these studies is 97.5% and 98.76%, respectively. However, the first of these studies [4] classifies only 29

TSL letters, while the other [5] classifies 10 TSL digits. In other words, it does not classify an expression or word. The process of classifying words or expressions is more complex and difficult. As a result, our study has a higher classification accuracy than similar studies in the literature.

Table 5. Comparison of our study with previous studies on TSL.

Study	Classifier	Data Acquisition Method	Dataset	ACC %
Haberdar and Albayrak [10]	HMM	LMC	50 static word	95.7
Memiş and Albayrak [11]	KNN	Kinect	111 static word	90
Demircioğlu et al. [9]	RF, MLP	LMC	18 static word	93.59
Çelik and Odabaş [8]	CNN+LSTM	InceptionV3, real-time	10 number 29 letter	97
Özcan and Baştürk [6]	CNN	Video	25 Dynamic word	93.93
Kircı, Durusan, Ozsahin [7]	LSTM	MediaPipe	29 letter	None
Pacal and Alaftekin [5]	CNN-ResNet	Video	10 nımber	98.76
Katılmış and Kazakuru [3]	Meta-ELM	LMC	26 dynamic word	93
Karacı et al. [4]	DNN	LMC, real-time	29 letter	97.5
This study	CNN+GRU GRU+Bi-LSTM Bi-LSTM GRU GRU+LSTM LSTM	MediaPipe	35 dynamic word 6 static word	96.72 95.62 95.26 94.53 93.80 92.34

4. Conclusion

This study aims to recognize Turkish Sign Language (TSL) expressions with various deep learning models (GRU, LSTM, Bi-LSTM, CNN+GRU, GRU+LSTM, and GRU+Bi-LSTM). The models were trained and tested using the hold-out validation method. According to the results, the highest classification performance was achieved with the hybrid model that sequentially used CNN and GRU, which accurately classified 41 sign language words with an accuracy rate of 96.72%.The GRU+Bi-LSTM model provided the second-highest accuracy rate with 95.62%. The results show that deep learning models can classify Turkish Sign Language words with very high accuracy rates and that CNN and GRU combinations are especially effective in sign language recognition problems. In addition to the high accuracy rate of the CNN+GRU model, it was also found that the probability of making incorrect classifications is low with low false positive rates. In addition, when the classification results were analyzed, it was seen that incorrect classifications occurred mostly among dynamic words. The absence of significant classification errors between dynamic and static words indicates that the models can effectively distinguish between these two types of words. As a future study, a new dataset will be created for 452 TİD words included in Turkish Sign Language Course Teaching Material. This dataset will not only include landmarks points but also skeletal images. DL models will be created on both landmark data and skeleton images and these models will be combined. Additionally, the classification performance of these deep learning models for TİD words will be demonstrated in real-time.

5. Acknowledgements

This study was supported by Scientific and Technological Research Council of Turkey (TUBITAK) under the Grant Number 124E379. The authors thank to TUBITAK for their supports.

6. References

[1] Alaftekin, M., Pacal, I., and Cicek, K., “Real-Time Sign Language Recognition Based on YOLO Algorithm”, *Neural Comput Appl*, vol. 36, no. 14, 7609–7624, 2024.

[2] Yirtici, T. and Yurtkan, K., “Regional-CNN-based Enhanced Turkish Sign Language Recognition”, *Signal Image Video Process*, vol. 16, no. 5, 1305–1311, 2022.

[3] Katılmış, Z. and Karakuzu, C., “Double handed Dynamic Turkish Sign Language Recognition Using Leap Motion with Meta Learning Approach”, *Expert Syst Appl*, vol. 228, 120453, 2023.

[4] Karacı, A., Akyol, K., and Turut, M. U., “Real-Time Turkish Sign Language Recognition Using Cascade Voting Approach with Handcrafted Features”, *Applied Computer Systems*, vol. 26, no. 1, 12-21, 2021.

[5] Pacal, I. and Alaftekin, M., “Türk İşaret Dilinin Sınıflandırılması için Derin Öğrenme Yaklaşımları”, *Iğdır Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, vol. 13, no. 2, 760-777, 2023.

[6] Özcan, T. and Baştürk A., “ERUSLR: A new Turkish Sign Language Dataset and Its Recognition Using Hyperparameter Optimization Aided Convolutional Neural Network”, *Journal of the Faculty of Engineering and Architecture of Gazi University*, vol. 36, no. 1, 527-542, 2021.

[7] Kirci, P., Durusan, B. B., and Özşahin, B., “El Hareketlerinden İşaret Dilini Algılayıp Yazıya Dönüştürme”, *European Journal of Science and Technology*, 32-35, 2022.

[8] Çelik, Ö. and Odabas, A., “Sign2Text: Konvolüsyonel Sinir Ağları Kullanarak Türk İşaret Dili Tanıma”, *European Journal of Science and Technology*, no. 19, 923 – 934, 2020.

[9] Demircioğlu, Bülbül, B., G., and Köse, H., “Turkish Sign Language Recognition with Leap Motion”, *in 2016 24th Signal Processing and Communication Application Conference, SIU 2016 - Proceedings*, 2016, 24.

[10] Haberdar, H. and Albayrak, S., "Real Time Isolated Turkish Sign Language Recognition from Video Using Hidden Markov models with Global Features", *in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 3733, 677–687, 2005.

[11] Memiş, A. and Albayrak, S., "A Kinect Based Sign Language Recognition System Using Spatio-Temporal Features", *Proc. SPIE 9067, Sixth International Conference on Machine Vision (ICMV 2013)*, 2013, 6.

[12] Martinez-Seis, B., Pichardo-Lagunas, O., Rodriguez-Aguilar, E., and Saucedo-Diaz, E.-R., "Identification of Static and Dynamic Signs of the Mexican Sign Language Alphabet for Smartphones using Deep Learning and Image Processing", *Research in Computing Science*, vol. 148, no. 11, 199-211, 2019.

[13] Aburass, S., Dorgham, O., and Al Shaqsi, J., "A hybrid Machine Learning Model For Classifying Gene Mutations in Cancer Using LSTM, BiLSTM, CNN, GRU, and GloVe", *Systems and Soft Computing*, vol. 6, 200110, 2024.

[14] Hochreiter, S. and Urgen Schmidhuber, J., "Long Shortterm Memory", *Neural Comput.*, vol. 9, no. 8, 1735–1780, 1997.

[15] Cho, K., Merriënboer, B. van, Bahdanau, D., and Bengio, Y., "On The Properties of Neural Machine Translation: Encoder–Decoder Approaches", *in Proceedings of SSST 2014 - 8th Workshop on Syntax, Semantics and Structure in Statistical Translation*, 2014, 8.

[16] Chung, J., Gulcehre, C., Cho, K., and Bengio, Y., "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling,", *ArXiv*, 2014. arXiv:1412.3555

[17] Karacı, A. and Akyol, K., "YoDenBi-NET: YOLO + DenseNet + Bi-LSTM-Based Hybrid Deep Learning Model for Brain Tumor Classification", *Neural Comput Appl.*, vol. 35, no. 17, 12583–12598, 2023.

[18] Karacı, A., "Predicting COVID-19 Cases on a Large Chest X-Ray Dataset Using Modified Pre-trained CNN Architectures", *Applied Computer Systems*, vol. 28, no. 1, 44–57, 2023.

[19] Maas et al., A. L., "Building DNN Acoustic Models for Large Vocabulary Speech Recognition", *Comput. Speech Lang.*, vol. 41, 195–213, 2017.