International Journal of Speech Technology
https://doi.org/10.1007/s10772-024-10092-x

Speech recognition based on the transformer's multi-head attention in Arabic

Omayma Mahmoudi[1,2] · **Mouncef Filali-Bouami**[1,2] · **Mohamed Benchat**[1,2]

Received: 16 November 2023 / Accepted: 16 February 2024

Abstract
The Transformer model is frequently employed for speech command recognition (SCR) since it supports parallelization and has internal attention. The high learning speed of this design and the absence of sequential operation, like with recurrent neural networks, are its two greatest advantages. In this work, Transformer models and data augmentation techniques to enhance the used dataset were considered to build a system for the automatic command recognition of Arabic speech. Little information is available for developing voice recognition systems in Arabic, which is recognized as a component of numerous significant languages. This study examines how the Transformer network is affected by hyperparameters to address two questions: Which hyperparameter is crucial for both task effectiveness and training efficiency? According to the results of our tests, it was found that using a Transformer with hyperparameter optimization helped the Arabic SCR system operate better.

Keywords Speech command recognition · MFCC · Transformers · RNN · LSTM · GRU · Attention mechanism · Data augmentation

1 Introduction

Modern individuals are starting to use speech technologies in their daily lives. Robotics, communications, and other professional fields offer an interesting scope for speech recognition. The case of SCR, used in voice assistant systems, is paradigmatic, recent studies have employed convolutional neural networks (CNNs) (Gupta et al., 2022) and recurrent neural networks (RNNs) (Mahmoudi & Bouami, 2023a), whereas earlier studies used deep neural networks with the Hidden Markov Model (HMM) (Eddy, 2004).

The attention mechanism is a widely used technique that significantly raises the system's effectiveness in speech recognition and machine translation. This attentional technique is employed by the Transformer model to speed up learning.

For such models to be implemented, a sizable volume of speech command data must be used for training, which presents a challenge for languages with little available training data, such as the Arabic language, as one of the agglutinative language's categories. Systems for identifying Arabic-spoken commands have been created thus far using various RNN models with different training sets (Mahmoudi & Bouami, 2023a). Our work has demonstrated that using Transformers approaches and models to increase Arabic SCR accuracy is a viable route for enhancing system performance and accelerating training. Our models are trained on an ASCR (Arabic SCR) dataset consisting of pairs of audio and corresponding phrase-level labels.

In this work, we proposed a model for the transformer network built with an attention mechanism to recognize commands in the Arabic language, it is distinguished by containing only an encoder, which helped us make the training process faster and smoother. In our study, we also relied on studying the effect of the hyperparameters (Li et al., 2020) of our transformer model, since they significantly improve speech commands recognition tasks, by discovering and

✉ Omayma Mahmoudi
mahmoudi.omayma@ump.ac.ma

Mouncef Filali-Bouami
m.filalibouami@ump.ac.ma

Mohamed Benchat
benchatm@gmail.com

[1] Laboratory of Applied Mathematics and Information Systems, Multidisciplinary Faculty of Nador, Mohammed Premier University, Oujda, Morocco

[2] Clausthal University of Technology, Adolph-Roemer-Str. 2A, 38678 Clausthal-Zellerfeld, Germany

Published online: 29 March 2024

knowing which plays a critical role in improving performance and training speed.

After training our transformer model and knowing the results, it was important to compare the obtained results with those of the RNN-BASED models, since they are among the important algorithms that achieved very impressive results in speech recognition in general (Mahmoudi et al., 2023). Therefore, we decided to compare our model's results with those of the RNN, LSTM, GRU, and BLSTM models.

The next step is the examination of our model using data augmentation (DA). For learning activities, DA is required. According to Salamon and Bello (2017), in addition to offering more training data to lessen the likelihood of overfitting during training, DA can also improve model performance and accuracy. The suggested technique in this study uses the notion of relevant data augmentation to enhance learning performance.

Our study's primary objective is to increase the recognition system's precision for Arabic commands speech and speed up the training process by utilizing Transformer-based models.

This paper's structure is as follows: The related works are described in the second section, and the speech Transformer Encoder approach is presented in the third section. The main topic of the fourth section isthe Data augmentation technique. The fifth section presents our methodology. the sixth section is the Results and discussions. The final section is the conclusion.

2 Literature review

While there has been much research on this subject, our work is the first to adopt the transformers model for SCR in Arabic.

Benamer and Alkishriwo (2020), Introduce the database of spoken Arabic instructions, which includes spoken Arabic numbers and six Arabic control order phrases. Three distinct recognition approaches were examined and their accuracy and performance were compared using the created database: Mel-Frequency Cepstrum Coefficients (MFCC) feature extraction using a K-Nearest Neighbor (KNN) classifier and Wavelet Time Scattering feature extraction using a Support Vector Machine (SVM) classifier. Lastly, their experimental findings demonstrate that the extraction of Wavelet Time Scattering features and an LSTM classifier provided the most accurate forecast of the database instructions and that the MFCC and KNN classifier provided the database with the fastest training time, which was 144 min.

Berg et al. (2021), present Without any pre-training or other data, the Keyword Transformer (KWT), a fully self-attentional design, exceeds state-of-the-art performance across a variety of tasks, is investigated as a means of modifying the Transformer architecture for keyword spotting. Unexpectedly, this straightforward design outperforms more intricate models that combine attentive, recurrent, and convolutional layers. Since KWT achieved accuracy for the 12 and 35-command tasks, 98.6% and 97.7%, respectively, on the Google Speech Commands in the English language dataset, it may be utilized as a direct swap for these models.

In the paper of Obaid et al. (2023), they used Vector Quantization (VQ) and Dynamic Time Warping (DTW) techniques to apply to voice recognition applications with a small vocabulary. The objective was to develop a speaker-independent, isolated-word recognition system with a short vocabulary that would be more successful at identifying Palestinian Arabic numerals between 0 and 9. By doing so, the accepted approach was enhanced, as was the operator´s dependability. The algorithm achieved a high level of accuracy. was accomplished by employing the MFCC method to take out particular speech characteristics, which were then utilized to lower the input voice's dimensionality. Digital Signal Processor (DSP) cards are used to recognize one-digit numbers and transmit orders to the outside world. The algorithm is downloaded to these cards.

Chen et al. (2022), introduced the HTS-AT model: a hierarchical audio transformer that condenses the model size and training time. The model for audio event identification (i.e. localization in time) is then made possible by further combining it with a token-semantic module that maps final outputs into class feature maps. They tested HTS-AT on three audio classification datasets and found that it achieves new state-of-the-art (SOTA) results on AudioSet and ESC50, as well as matching the SOTA on Speech Command V2. It also outperforms earlier CNN-based algorithms in event localization. Furthermore, HTS-AT uses just 35% of the model parameters and 15% of the training time of the prior audio transformer. These findings reveal HTS-AT's superior performance and efficiency.

3 Speech transformer encoder

Our architecture is in Fig. 1. where we extract MFCC (Zheng et al., 2001) features from a waveform diagram on which we apply a 2D convolution.

The Transformer Encoder is composed of two sub-layers: a first layer for Multi-head Attention followed by another for Feedforward. For regularization purposes we added a Dropout and a Normalization blocks, the connection around each sub-layer is residual.

3.1 Positional encoding

As in Liao et al. (2022) the Positional Encoding (PE) concept is introduced to get around the absence of recurrence in

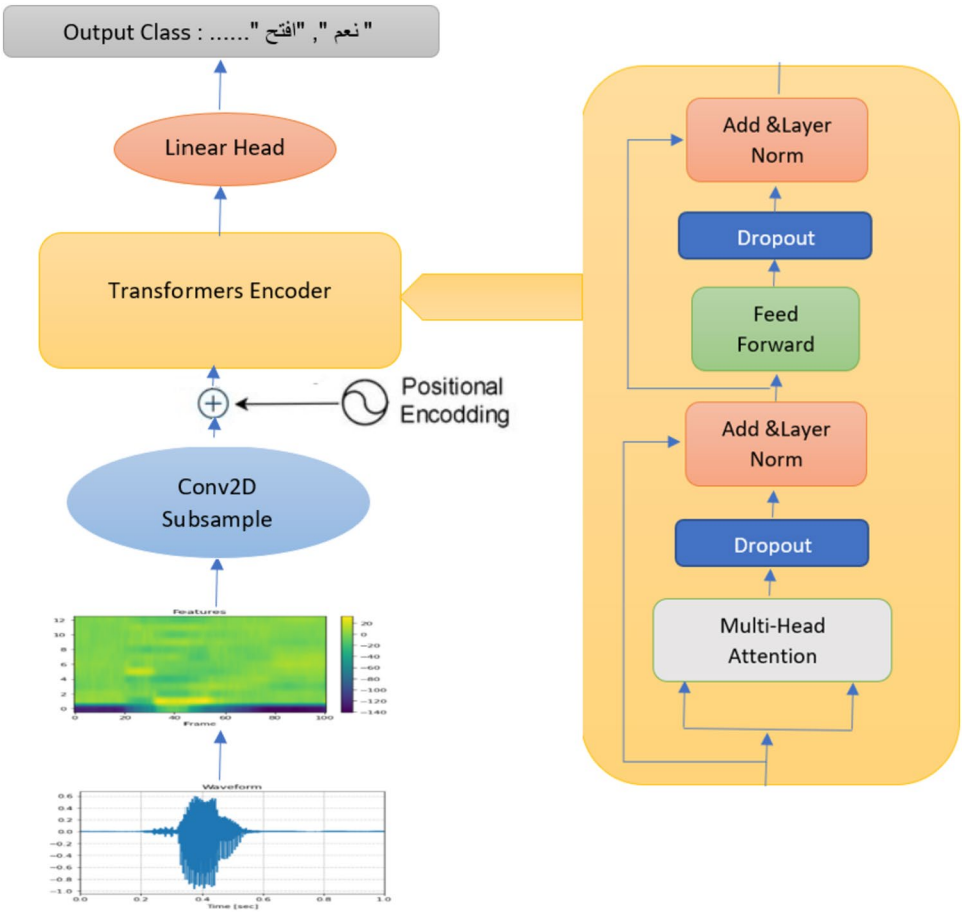

Fig. 1 The architecture of the keyword transformer. A preprocessed MFCC spectrogram is created from audio, and a positional encoding is added. The final class prediction is based on the output of the class after being routed through a linear head

the Transformer encoder. To accomplish this, A few details about the data's place, the PE is intentionally created and appended to the incoming data. PE is supplied by Eq. (1):

$$PE_{(pos,2i)} = sin\left(\frac{pos}{10{,}000^{2i/d_{\text{model}}}}\right)$$
$$PE_{(pos,2i+1)} = cos\left(\frac{pos}{10{,}000^{2i/d_{\text{model}}}}\right) \quad (1)$$

The variables pos, i, and d_{model} represent the location, dimension, and number of attention units, respectively, in an input sequence.

3.2 Multi-head attention

The transformer encoder's attention mechanism (Obaid et al., 2023) is described as Eq. (2):

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (2)$$

where V is the weighted sum of the values, and attention output is the sum of these same values. By combining the queries, Q, and associated keys, K, into a dot product, the weights given to the values are calculated. hence the reference to this mechanism as "Scaled Dot Product Attention". d_k is the dimension of the key "self-attention" case is done when $Q = K = V$.

However, the Transformer encoder uses a multi-head attention (MHA) to perform multiple attention. The attention function is then applied to each of the queries, keys, and values that MHA projected using h different linear transformations. To achieve the final result, the h distinct attention outputs are then concatenated and projected once more. This is how the MHA is described, as shown in Eq. (3):

$$\text{MultiHead}\,(Q, K, V) = Concat\left(H_1, H_2, \cdots, H_h\right)W^O$$
$$H_i = \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right) \quad (3)$$

where the learnable projections are $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$, and $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$. In addition, we have $d_k = d_v = d_{\text{model}}/h$.

3.3 Feed-forward networks

Each location is applied independently and identically via a fully linked feed-forward network (Chen et al., 2022) in addition to the MHA layer. The definition of this, which consists of two linear layers, is as in Eq. (4):

$$\mathrm{FFN}(X(t)) = m\big(0, X(t)W_1 + b_1\big)W_2 + b_2 \tag{4}$$

where $X(t)$ is the t-th frame of the input sequence X, and $W_1 \in \mathbb{R}^{d_{\text{model}} \times d_{ff}}$, $W_2 \in \mathbb{R}^{d_{ff} \times d_{\text{model}}}$, $b_1 \in \mathbb{R}^{d_{ff}}$, $b_2 \in \mathbb{R}^{d_{\text{model}}}$.

4 Data augmentation

The amount of dataset samples is increased using Data Augmentation (DA). Additionally, it is a method for expanding the amount of training datasets usable for neural network training. DA has a significant impact on deep learning with small datasets. It is a helpful technique for resolving overfitting issues, improving model reliability, and boosting generalization (Salamon & Bello, 2017). To increase prediction accuracy while managing enormous amounts of data, research-based deep learning with data augmentation approaches is essential. There are some techniques for enhancing data, such as introducing white noise into an initial sample, altering the pitch, varying the loudness, using various window sizes, and lengthening the duration.

Audio DA is an approach used to intensify A voice dataset's size by adding altered copies of existing data (Mushtaq & Su, 2020). Two common methods for augmenting audio data are gain and polarity inversion (Falcon-Perez). Gain involves applying a uniform random gain to the input signal, which changes the overall loudness of the audio. The gains are selected from a range of -15 dB to 6 dB. Polarity inversion randomly inverts the polarity of the input signal, which flips its waveform upside down. The tests on Arabic samples have been performed and proved the suitability of new data for the forthcoming phases.

There are several Python libraries available for implementing methods for augmenting auditory data, including audiomentations (Ferreira-Paiva et al.). This library provides various augmented methods for example pitch shifting, time-scale modification, time shifting, noise addition, and more. In addition to these techniques, audiomentations also support gain and polarity inversion.

Figure 2 shows the data augmentation stage we performed.

5 Methodology

5.1 Proposed model based on transformer network

Starting from the Transformer Network, the architecture we propose is shown in Fig. 3. The fundamental elements of Speech Transformer seek to extract the character sequence from the equivalent word feature sequence. A spectrogram of frequency with respect to time can be used to represent the feature sequence, which is often many times longer than the character sequence. To avoid this inconvenience, we decided to use convolutional networks to take advantage of the locality of the spectrogram's structure and to compensate for the length mismatch by moving forward in time. Based on the preceding, we offer the Speech Transformer's model architecture and the encoder's specifics as follows:

To prevent GPU (Owens et al., 2008) memory overflow and to approximate the hidden representation length with the character length, we first stack two 3×3 CNN layers with stride 2 for both the time and frequency dimensions. The outputs of the flattened feature map are then subjected to a linear transformation to produce vectors with the dimension d_{model}, also known as input encoding in this context. Following that, the d_{model} dimensional positional encoding is applied to the input encoding, enabling the model to attend to the relative locations. The final encoded outputs may be obtained by feeding the total of input encoding and

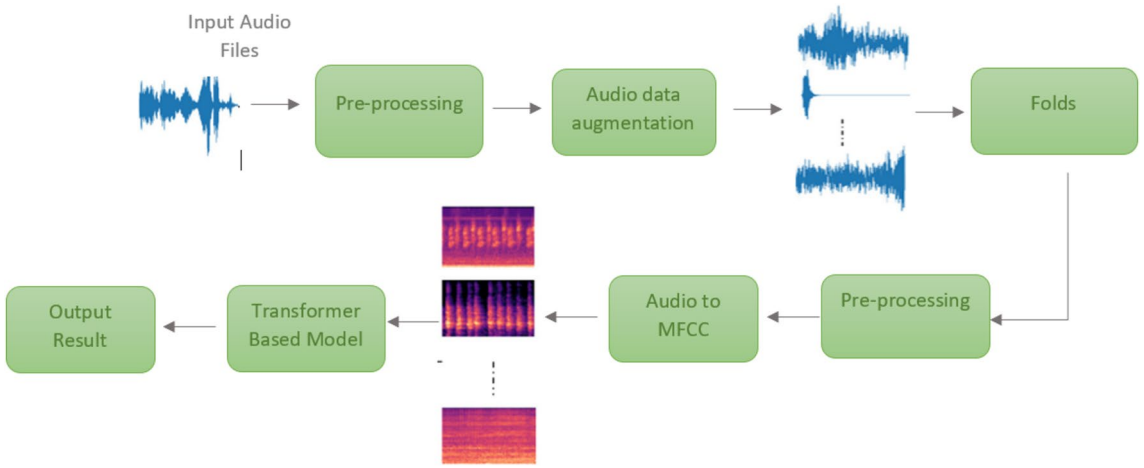

Fig. 2 System of Data augmentation approach

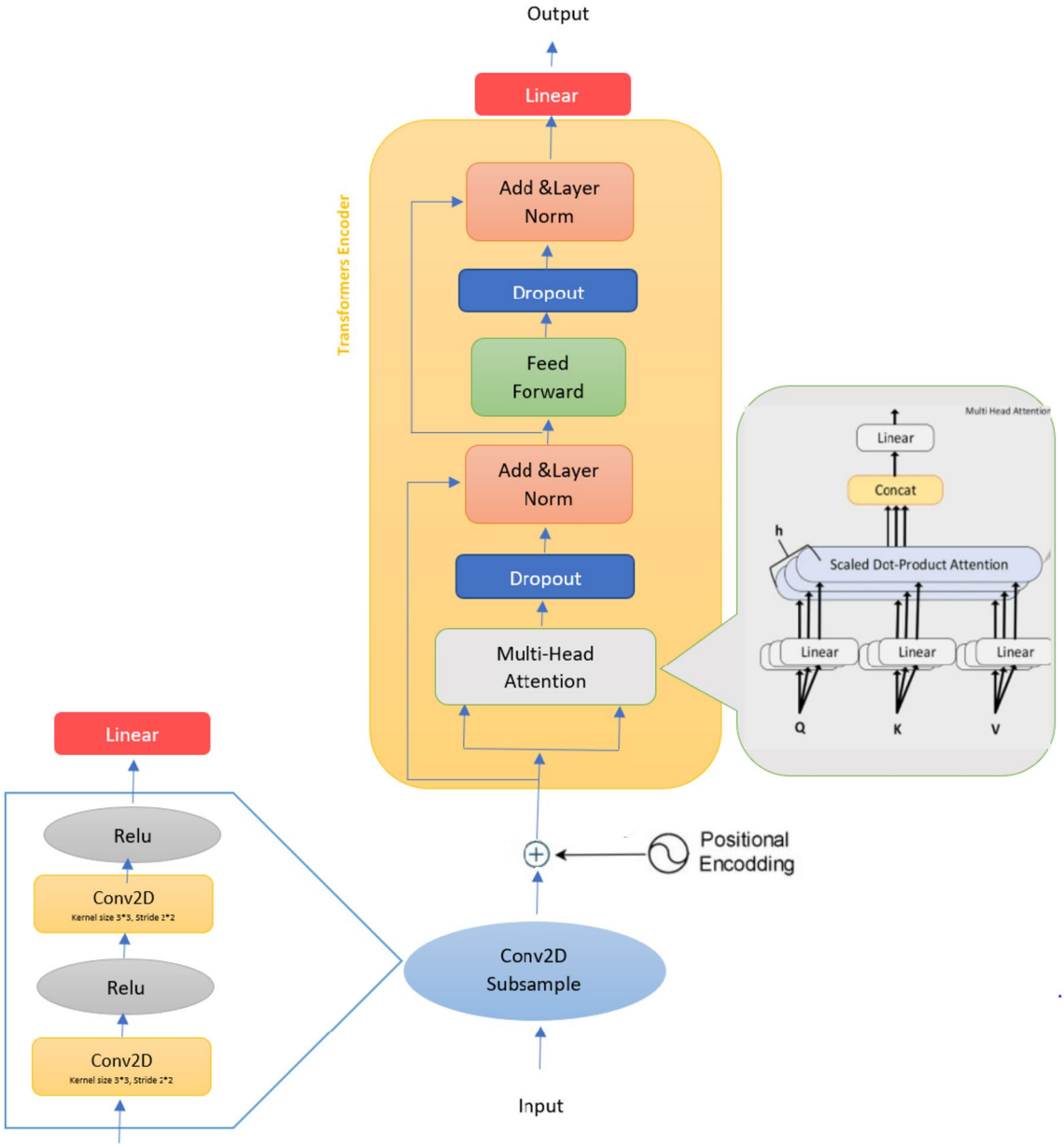

Fig. 3 Proposed Transformer Network Model

positional encoding into a stack of Transformer Encoders, each of which contains two sub-blocks: The first is multi-head attention with queries, keys, and values derived from the preceding block's outputs. The second kind is position-based feed-forward networks. In the meantime, each sub-block receives layer normalization and residual connections for successful training (Usman et al., 2022).

5.2 RNN model

In contrast to feedforward neural networks, RNNs process input sequences utilizing their internal state (memory). All of the inputs to the RNN are connected. Among other things, neural networks increase across an extensive of technical applications, comprising prediction and human behavior modeling. RNNs, in particular, are used to represent temporal and time-series data because they have built-in memory that retains previous information. By including feedback connections, the RNN is extended from the feed-forward neural network to the model sequence knowledge. Given an input sequence of $(x_1, x_2, \ldots, x_n)$, the typical RNN may iteratively acquire another sequence$(y_1, y_2 \ldots y_H)$ using the Eqs. (5) and (6).

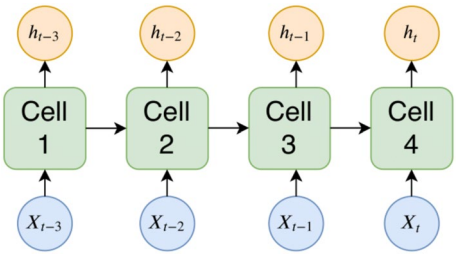

Fig. 4 RNN architecture

$$h_l = \varnothing\left(W_h x_t + U_h h_{t-1} + b_h\right) \tag{5}$$

$$x_t = \varnothing\left(U_y h_t + b_y\right) \tag{6}$$

h denotes the hidden state, φ(-) denotes the activation functions, and W,U,b denote the weights and biases to be learned, Because the gradient norm decreased rapidly during training, i.e., during the exploding and vanishing gradient issues, basic RNNs require control mechanisms. As a result, ordinary RNNs are insufficient for detecting neither long-term nor short-term dependencies. Figure 4 depicts an implementation of the long short-term memory (LSTM) network, a unique RNN design with control mechanisms, to overcome this challenge.

5.3 LSTM model

LSTM (Mahmoudi & Bouami, 2023b) networks are a more advanced variant of RNNs that aid in the recall of previously learned information. This solution resolves the RNN's vanishing gradient problem. The LSTM algorithm is highly adapted for recognizing, analyzing, and predicting time series with uncertain length time delays. Back-propagation training is used to train the model. Training problems might occur in an LSTM network with several parameters. To solve these challenges, very effective and fast training methods for the LSTM architecture have been created. Figure 5 depicts an LSTM design with multiple memory cells.

These memory cells are utilized to store prior historical data to preserve long-term temporal dependencies. An LSTM is a collection of units with identical parameters at all time steps. The LSTM model has three memory cells: input, forget, and output gates, indicated by the symbols i_s f_1, ou, respectively. To train datasets, the learning process changes these gates. As shown in Eqs. (7) to (12):

$$i_1 = \sigma\left(W_i x_1 + U_j h_{i-1} + b_i\right) \tag{7}$$

$$c_i = Q\left(W_c x_1 + U_c h_{j-1} + b_c\right) \tag{8}$$

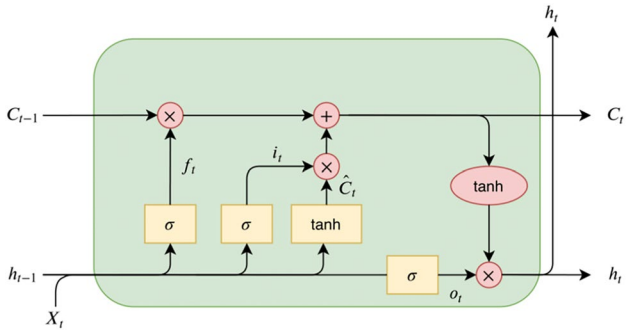

Fig. 5 LSTM architecture

$$f_t = \sigma\left(W_f x_i + U_f h_{i-1} + b_f\right) \tag{9}$$

$$o_1 = \sigma\left(W_0 x_t + U_0 h_{t-1} + b_b\right) \tag{10}$$

$$c_t = f_t \odot c_{1-1} + h_i \odot C_t \tag{11}$$

$$h_t = \varnothing\left(c_t\right) \otimes o_t \tag{12}$$

5.4 GRU model

Mahmoudi and Bouami (2023b) offer a GRU approach that allows each recurrent unit to collect sequence dependencies adaptively across time. The GRU contains a gating mechanism that regulates information flow throughout the unit. In the following procedure, x_t represents the current input, and, h_{t-1} represents the result of the previous step. The update gate z_t is calculated using the following Eq. (13):

$$z_t^j = \sigma\left(W_z x_t + U_z h_{t-1}\right)^j \tag{13}$$

The amount of information that the unit will update is determined by an update gate z_t^j.

The reset gate r_j^t and $\tilde{h}_t^j$ is calculated by the following Eqs. (14) and(15).

$$r_t^j = \sigma\left(W_r x_t + U_r h_{t-1}\right)^j \tag{14}$$

$$\tilde{h}_t^j = tanh\left(W_h x_t + U\left(r_t \cdot h_{t-1}\right)\right)^j \tag{15}$$

where is a multiplication done element-by-element and r_t is a collection of reset gates. The reset gate is off when r_t^j is near 0.

supplied by Eq. (16)

$$h_t^j = \left(1 - z_t^j\right) h_{t-1}^j + z_t^j \tilde{h}_t^j, \tag{16}$$

For a visual representation of the GRU cell, see Fig. 6.

5.5 Feature extraction

Speech parameterization by feature extraction is used to describe the spectral properties of an audio signal, a way to decode an input speech. The MFCC, developed by Davis et al. can mimic human ear activity. The essential processes in the computation of MFCC characteristics are shown in Fig. 7.

5.5.1 Frame blocking

Frame blocking (Pervaiz et al., 2020), in the first phase of the procedure, the streaming audio signal is divided into frames 25 ms long and 10 ms apart.

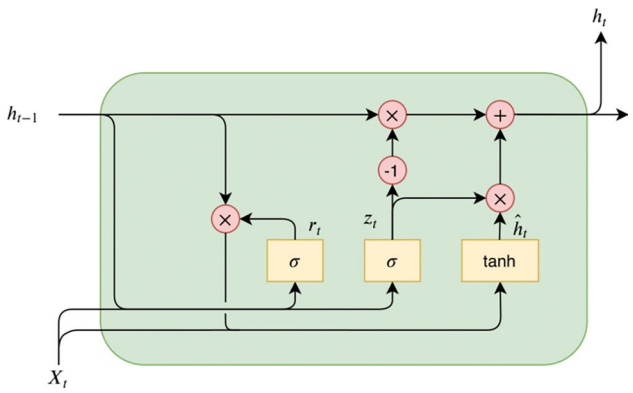

Fig. 6 GRU architecture

5.5.2 Windowing

Following blocking, a windowing (Pervaiz et al., 2020) function multiplies each frame by a window. Although Kaldi offers a variety of windowing options, the Hamming window is most frequently employed, as Eq. (17) illustrates:

$$w(n) = 0.54 - 0.46cos\left(\frac{2\pi n}{N-1}\right) 0 \le n \le N-1 \tag{17}$$

where N represents the total of examples in a frame. Discontinuities at the extremes of each frame are reduced by multiplying each frame by the Hamming window.

5.5.3 Fast fourier transform

Speech signals' spectral analysis revealed that their various timbres had distinctive energy distributions over frequency. Every N sample frame underwent FFT (Pervaiz et al., 2020) to determine its frequency and amplitude of reaction. The time domain signals were converted into the hesitancy domain signals. The Discrete Fourier Transform (DFT) is quickly implemented using FFT.

5.5.4 Mel frequency warping

To obtain each bypass filter's log energy, triangle bypass filters were used to double the FFT's magnitude frequency response on the Mel scale. The Mel frequency imitates how the human ear perceives sound non-linearly by being less selective at higher frequencies and more selective at lower frequencies. On both the scales of hesitancy and capacity, the human ear responds to speech in a logarithmic manner, meaning that the hesitancy resolution is higher (high) at lower hesitancy and lower (low), especially at higher hesitancy. Additionally, audio waves are converted into electrical impulses in the cochlea of

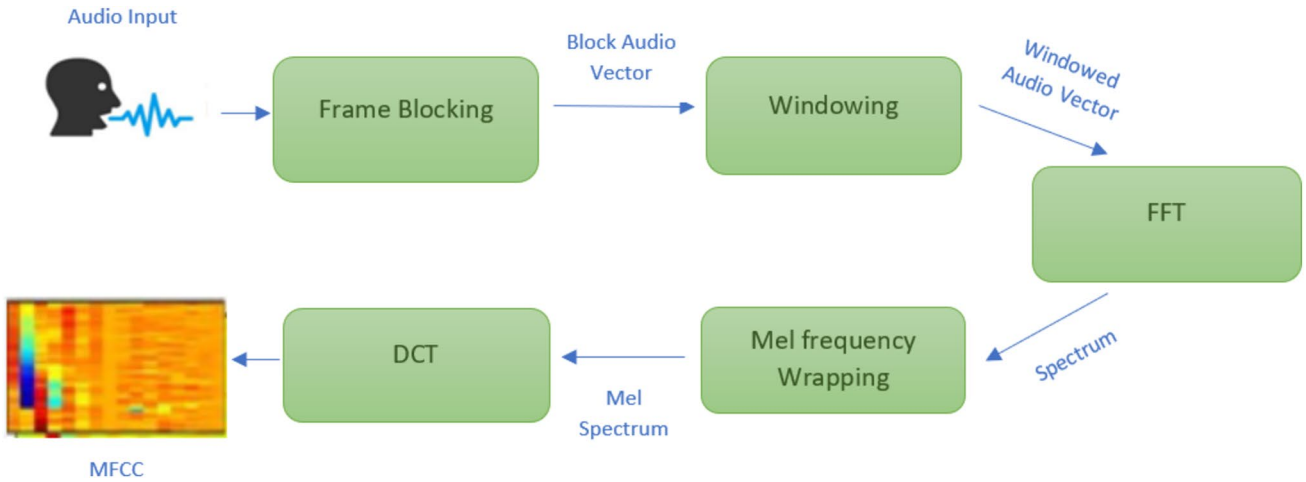

Fig. 7 Steps of MFCC Feature Extraction

the human inner ear, which the brain may then interpret as specific sound frequencies (Pervaiz et al., 2020). This is comparable to how MFCC calculates cepstral coefficients. Using Eqs. (18) and (19), we may translate between Mel hesitancy (m) and hesitancy (f) in Hz.

$$m = 2595log_{10}\left(1 + \frac{f}{700}\right) \tag{18}$$

$$f = 700\left(10^{m/2545} - 1\right) \tag{19}$$

The responses of every triangle filter the bank of filters were one at the center hesitancy and linearly declined to zero until they reached the adjacent hesitancy's center.

5.5.5 Discrete cosine transform

The MFCC was produced by applying DCT (Pervaiz et al., 2020) to the filtered output of log energy calculations. Mel frequency cepstrum coefficients (MFCC) are the names given to the components of the Mel frequency cepstrum. After FFT, the frequency domain is converted into the time-like domain known as the quefrency domain using DCT. By default, the characteristics consist of the top 13 cepstral coefficients. Speech recognition systems can directly employ MFCC characteristics. It is possible to calculate first- and second-order deltas. First-order deltas may be computed using Eq. (20), given a feature vector X.

$$\Delta X_t = \frac{\Sigma_{j=1}^{n} w_l\left(X_{t+1} - X_{t-i}\right)}{2\Sigma_{i=1}^{n} w_i^2} \tag{20}$$

where the window width is n and the regression coefficients are w. Eq. (21) may be used to convert first-order deltas into second-order deltas.

$$\Delta X_t = \frac{\sum_{i=1}^{n} w_i\left(\Delta X_{t+i} - \Delta X_{t-i}\right)}{2\sum_{i=1}^{n} w_i^2} \tag{21}$$

The combined feature vector becomes after the first- and second-order delta calculations, as shown in Eq. (22):

$$\Delta X_f = \left[X_t \Delta X_t \Delta^2 X_t\right] \tag{22}$$

6 Materials and experimental setup

6.1 Dataset

Our data set (Ghandoura, 2021) consists of a list of pairs (x, y), where x represents the input voice signal and y is the associated keyword. The completed dataset is made up of 12,000 such pairings with 40 commands each. One second of acoustic is sampled at a rate of 16 kilohertz for every audio file. Each among the 30 candidates 10 utterances for each of the 40 keywords, as shown in Table 1. As a result, each keyword has 300 audio files, with a total file size of 384 MB for all the recorded keywords. The dataset also contains several background noise recordings we obtained from various natural sources of noise. We saved these audio files in a separate folder with the name background noise and a total size of ~ 49 MB.

6.2 Data split

20% of the dataset was maintained for validation, 20% for testing, and 60% was kept for training.

6.3 Performance evaluation

The tests were carried out on an Ubuntu 18.04 LTS machine with 8 GB of RAM, an NVIDIA GTX 1050Ti GPU with 4 GB. The Python 3.6.9 environment includes PyTorch 1.10.1 + cu102, torchaudio 0.10.1 + cu102, Ray 2.3.0.

Table 1 The 40 chosen keywords and their arabic transcriptions

Translation	Keyword	Translation	Keyword	Translation	Keyword	Translation	Keyword
Zero	صفر	Right	يمين	Enable	تفعيل	Send	إرسال
One	واحد	Left	يسار	Disable	تعطيل	Receive	استقبال
Two	اثنان	Up	اعلى	Ok	موافق	Move	تحريك
Three	ثلاثة	Down	أسفل	Cancel	الغاء	Rotate	تدوير
Four	اربعة	Forward/Font	امام	Open	فتح	Record	تسجيل
Five	خمسة	Backward/Back	خلف	Close	اغلاق	Enter	ادخال
Six	ستة	Yes	نعم	Zoom in	تكبير	Digit	رقم
Seven	سبعة	No	لا	Zoom out	تصغير	Direction	إتجاه
Eight	ثمانية	Start	ابدأ	Previous	السابق	Options	خيارات
Nine	تسعة	Stop	توقف	Next	التالي	Undo	تراجع

Table 2 Hyperparameter optimization for transformers model

Hyperparameter	Value	Search range
Learning rate	7.77125e−05	(0.1, 0.2)
Number of layers	12	[3, 6, 9,12,14]
Number of heads	8	[2,4, 8, 16, 24]
Batch size	32	[8, 16, 32, 64, 128]

6.4 Experiment with hyperparameters setting

With Transformer-based systems, numerous hyperparameters must be optimized to get optimal performance. These variables have an impact on the algorithm's efficiency and time and memory requirements. For machine learning algorithms, the choice of hyperparameters frequently distinguishes between subpar and cutting-edge performance. Optimization is required because the optimal values will vary depending on the type of data used in the comparisons, the specific data sets used, the performance criteria, and various other factors. The Arabic language is considered different and distinct in pronunciation, so its hyperparameters will differ from the hyperparameters used for the Latin languages in general. This is true even though there are some general guidelines in the research community about the appropriate values of these hyperparameters. In Table 2 the hyperparameters that we tuned are listed. The most crucial hyperparameter is probably the learning rate (Mahmoudi et al., 2022). We only tuned the hyperparameters using the Golden dataset to prevent over-tuning the model. We then used the same values on additional datasets. We discovered that 12 layers was the ideal quantity. When we used the numbers 2, 4, 8, 16, and 24 as heads, we discovered that number 8 performed better than the other two heads. We used dropout for regularization since a deep neural network with numerous parameters is prone to overfitting. When a neural network is being trained, units (along with their connections) are randomly removed. 12 layers, 8 heads, 32 batch sizes, and 100 epochs are the hyperparameters that were discovered for our model.

6.5 Performance comparison of models based on RNN:

RNN-based models were investigated, for Arabic speech commands recognition and compared with Transformer-based, finding the best ASCR performance through performance results.

Bidirectional LSTM, GRU, LSTM, and RNN models are used to evaluate performance. The hyperparameter utilized to evaluate the RNN-based models can be seen in Table 3.

Table 3 Hyperparameters are used to examine the models based on RNN

Hyperparameter	Bidirectional LSTM	GRU	LSTM	RNN
Learning rate	0.0001	0.0001	0.0001	0.0001
Layers dim	3	3	3	3
Hidden dim	64	64	64	64
Dropout rate	0.1	0.1	0.1	0.1
Batch size	64	64	64	64

7 Results and discussions

7.1 Result without augmentation data

In the multiple classification task known as Arabic Speech Commands, audio files are categorized as keywords. Testing on a dataset of Arabic speech commands reveals that pre-trained transformer transfer learning outperforms other cutting-edge deep learning models in terms of performance. As shown in Table 4, after using hyperparameter tuning and discovering the best one, we reached the highest accuracy for the transformer model, which is 0.87%.

Table 4 shows the performance and characteristics of different models, including the transformer encoder, LSTM, Bi-LSTM, GRU, and RNN, in the context of a particular task, presumably automatic command recognition in Arabic speech. Here's a discussion and interpretation of the results:

7.1.1 Transformer encoder model

Achieved the best accuracy on both validation and test data (87.51% and 87.38%, respectively).

Had a relatively short training time compared to LSTM, Bi-LSTM, and GRU models.

The Transformer architecture, known for its attention mechanisms and parallel processing capabilities, effectively captured the contextual relationships in the Arabic speech commands, leading to high accuracy.

The shorter training time indicates that the Transformer model learns representations efficiently, making it an attractive choice for tasks where training time is a consideration.

7.1.2 LSTM, Bi-LSTM, and GRU models

Achieved the best accuracy on the training data, reaching 100% accuracy.

Had the longest training times compared to other models.

LSTM and Bi-LSTM models are known for their ability to capture long-term dependencies in sequential data, which may have contributed to their high accuracy on the training data.

Table 4 The performance comparison of the proposed transformer architecture with RNN-based

Model performance	Training		Validation		Time take (s)	Test		Time take (s)
	Loss	Acc	Loss	Acc		Loss	Acc	
RNN	1.3598	0.5752	1.7514	0.5302	3788	0.0269	0.5458	8.1445
LSTM	0.0003	1.000	1.0607	0.8162	4788	0.0154	0.7950	18.93
GRU	0.0003	1.000	1.2770	0.7932	4761	0.0169	0.7296	8.9669
Bi LSTM	0.0003	1.000	1.0058	0.8128	4639	0.0151	0.7450	18.8
TRANSFORMER	0.0009	1.000	0.6013	0.8751	4121	0.6213	0.8738	7.8759

The longer training times suggest that LSTM and Bi-LSTM models may require more iterations to converge compared to the Transformer model.

7.1.3 RNN model

Had the worst accuracy on all three datasets.

Also had the longest test time, indicating inefficiency in processing test data.

RNNs are susceptible to the vanishing gradient problem, which can hinder their ability to capture long-term dependencies effectively, leading to poor performance.

The longer test time suggests that the RNN model may struggle with computational efficiency, especially when processing large amounts of test data.

Overall Interpretation:

The results highlight the superiority of the Transformer encoder model in terms of accuracy and training efficiency for automatic command recognition in Arabic speech.

LSTM and Bi-LSTM models achieve perfect accuracy on the training data but may suffer from longer training times and potential overfitting.

The poor performance of the RNN model on all datasets underscores its limitations in capturing long-term dependencies and processing efficiency.

The choice of model architecture significantly impacts the performance and efficiency of automatic command recognition systems in Arabic speech. The Transformer encoder model emerges as the preferred choice due to its high accuracy, relatively short training time, and effective handling of contextual relationships in speech data.

7.2 Result with augmentation data for transformer models

This part displays our model's performance evaluation of data augmentation. The sum number of augmented audio dataset files is the same as the original data. The total number of augmented and non-augmented data is 24,000.

Table 5 shows the results of training a transformer-based model on a sentiment analysis task, with and without data augmentation. This can help improve the model's ability to generalize to unseen data.

Key observations from the table:

Accuracy: The model's accuracy is higher on the augmented data compared to the original data, for both the validation set and the test set. This suggests that data augmentation improved the model's performance in this case.

Loss: The model's loss is lower on the augmented data compared to the original data, for both the validation set and the test set. This suggests that the model may have learned better representations from the augmented data, even though it achieved higher accuracy.

Time: Training with augmented data takes longer than training with only the original data. This is likely because the model needs to process more data overall.

Additional notes:

The table shows the results for two different training data configurations:

Training on the original data only.

Training on a combination of the original data and the augmented data.

The specific data augmentation techniques that were used are not shown in the table. This could affect the interpretation of the results.

The size of the training, validation, and test datasets is not shown in the table. This could affect the generalizability of the results.

Overall, the table suggests that data augmentation can be an effective way to improve the performance of a transformer base model on a speech command recognition task.

Figure 8 presents the results of the performance evaluation of our model using increased data and compares it with the results of the original data. After training a total of augmented and original datasets, we discovered that the accuracy of the original data test is 0.768% and that of the test of augmented data is 0.61%. For training only on augmented data, the accuracy of the original data test is 0.36% and that of the augmented data test is 0.37%. What

Table 5 Comparing the effectiveness of the proposed transformer architecture using augmented + not augmented data and augmented data only

Transformer based model											
Augmentation technique	Validation original data		Time Take (s)	Validation augmented data		Time take (s)	Test original data		Time take (s)	Test augmented data		Test augmented data
	Loss	Acc		Loss	Acc		Loss	Acc		Loss	Acc	
Training data (with augmented + original data)	0.7681	0.7799	5221	1.2528	0.6386	5770	0.0249	0.7688	15.09	0.0438	0.6079	5.86
Training data (with augmented data)	2.1230	0.4098	3283	2.1876	0.3856	3112	0.0685	0.3654	4.878	0.0688	0.3738	5.4672

prompted us to train the augmented data was our curiosity to know the test results. However, what interests us, in this case, is the results of training on the total original and augmented data sets.

Our research has been contrasted with other studies that looked into the Arabic language's speech recognition. Table 6 displays a variety of findings from earlier research.

MFCC + Transformer Encoder achieves the highest accuracy of 95% in 2024. This suggests that this approach is currently the most effective for speech recognition.

The accuracy of most approaches has improved over time. For example, MFCC + CNN improved from 82% in 2021 to 86% in 2020. This shows that research in speech recognition is making progress.

Some approaches, such as MFCC + DTW and DHMM, have seen a decrease in accuracy over time. This could be due to several factors, such as the limitations of these approaches or the increasing difficulty of the datasets.

It is not clear which approach is best for all tasks and datasets. The best approach may vary depending on the specific requirements of the task.

8 Conclusion

In this paper, we present our proposal for Transformer architecture for speech command recognition in Arabic. Since our transformer model's hyperparameters greatly enhance voice command recognition tasks, we also used them to analyze their effects to determine which hyperparameters are most important for enhancing performance and training efficiency. After training our transformer model and extracting the results, it achieved a high accuracy rate compared to the results of the RNN-BASED models, since they are among the important algorithms that produced great results in speech recognition in general. We then augmented the data using data augmentation (DA) to lessen the likelihood of overfitting during training and improve model performance and accuracy.

While previous works focus on applying different deep learning techniques to some existing or self-build datasets, our objective was to introduce a proper model based on one of the ultimate and most effective algorithms such as the Transformer Model. The dataset, the subject of our experiments, is robust enough to justify the carrying out of this research and validate the results obtained.

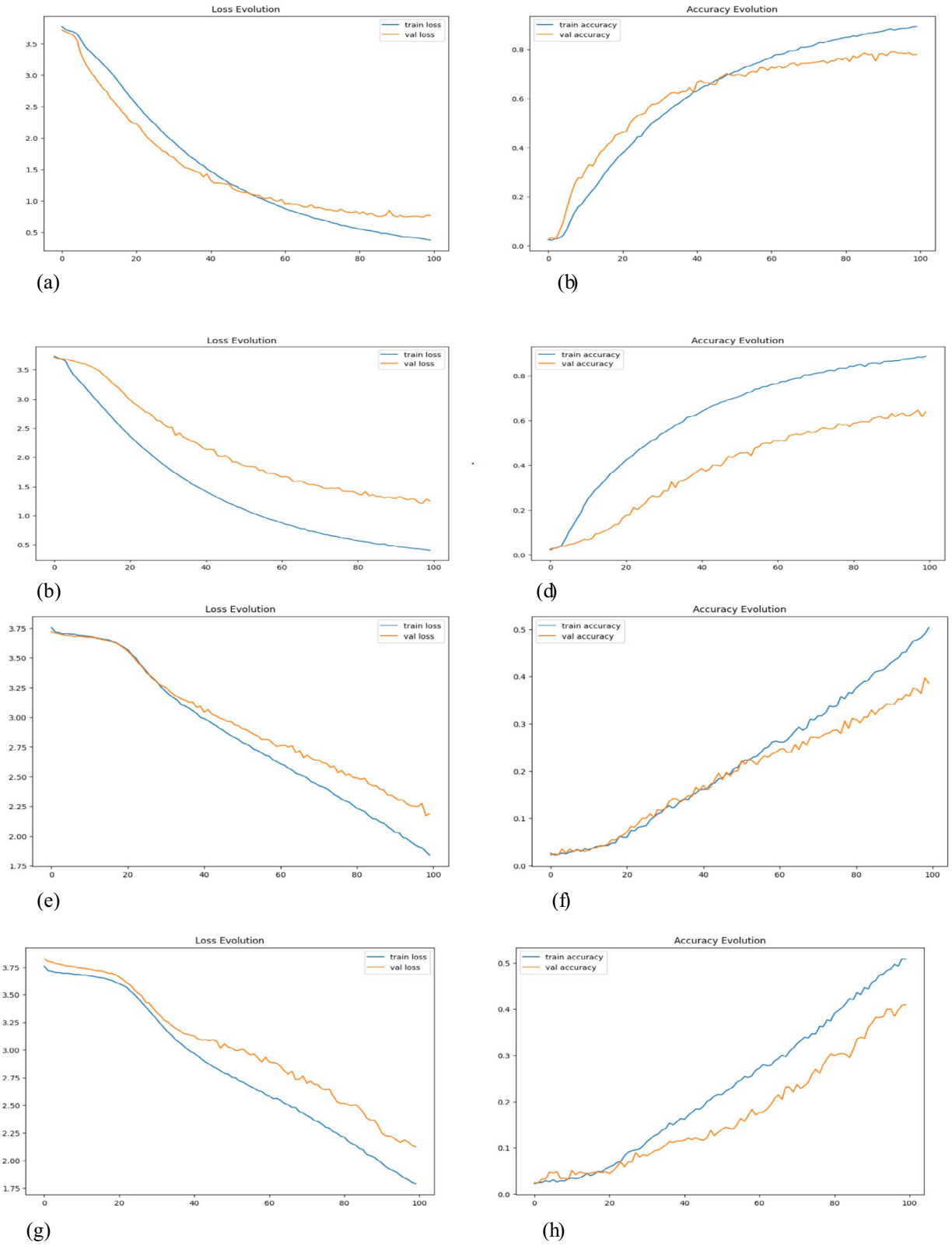

Fig. 8 The suggested method accuracy compared to validation accuracy of Transformer architecture **a**, **b** are related to augmented and original data for training and original data for validation, **c**, **d** belong to augmented and original data for training and augmented data for validation; and **e**, **f** is associated with augmented data for training and original data for validation. **g**, **h** are associated with augmented data for training and augmented data for validation

Table 6 Results of different approaches

Refs.	Year	Approach	Accuracy (%)
Our work	2024	MFCC + Transformer encoder	87
[28]	2021	MFCC + CNN	82
[29]	2020	MFCC + HMM and CNN	84
[30]	2020	MFCC + SVM	86
[31]	2019	MFCC + RNN	69
[32]	2011	MFCC + DTW and DHMM	92

Declarations

Conflict of interest None. We hereby confirm that all the Figures and Tables in the manuscript are ours. Besides, the Figures and images, which are note ours, have been given permission for re-publication attached with the manuscript. The autors declare they have no financial interests. Our manuscript has associated data in a data repository.

References

Benamer, L., & Alkishriwo, O. (2020). Database for Arabic speech commands recognition, In *CEST the third conference for engineering sciences and technology,* 1–3 December, Al Khums, Libya.

Berg, A., O'Connor, M., & Cruz, M. T. (2021). Keyword transformer: A self-attention model for keyword spotting. arXiv preprint arXiv:2104.00769.

Buntine, W. L., & Weigend, A. S. (1994). Computing second derivatives in feed-forward networks: A review. *IEEE Transactions on Neural Networks, 5*(3), 480–488.

Chen, K., Du, X., Zhu, B., Ma, Z., Berg-Kirkpatrick, T., & Dubnov, S. (2022). HTS-AT: A hierarchical token-semantic audio transformer for sound classification and detection. In *2022 IEEE international conference on acoustics, speech and signal processing (ICASSP 2022)* (pp. 646–650). IEEE.

Eddy, S. R. (2004). What is a hidden Markov model? *Nature Biotechnology, 22*(10), 1315–1316.

Falcon-Perez, R. (2022). Curriculum learning with audio domain data augmentation for sound event localization and detection. In *Challenge of detection and classification of acoustic scenes and events.*

Ferreira-Paiva, L., Alfaro-Espinoza, E., Almeida, V. M., Felix, L. B., & Neves, R. V. (2022). A survey of data augmentation for audio classification, In *XXIV Brazilian congress of automatics (CBA)*. https://doi.org/10.20906/CBA2022/3469.

Ghandoura, A. (2021). Arabic speech commands dataset (v1.0). Zenodo. https://doi.org/10.5281/zenodo.4662481.

Gupta, J., Pathak, S., & Kumar, G. (2022). Deep learning (CNN) and transfer learning: A review. *Journal of Physics: Conference Series 2273*(1), 012029.

Li, H., Chaudhari, P., Yang, H., Lam, M., Ravichandran, A., Bhotika, R., & Soatto, S. (2020). Rethinking the hyperparameters for fine-tuning. arXiv preprint arXiv:2002.11770.

Liao, L., Afedzie Kwofie, F., Chen, Z., Han, G., Wang, Y., Lin, Y., & Hu, D. (2022). A bidirectional context embedding transformer for automatic speech recognition. *Information, 13*(2), 69.

Mahmoudi, O., Bouami, M. F., & Badri, M. (2022). Arabic language modeling based on supervised machine learning. *Revue d'Intelligence Artificielle, 36*(3), 467.

Mahmoudi, O., & Bouami, M. F. (2023). RNN and LSTM models for Arabic speech commands recognition using PyTorch and GPU. In *International conference on artificial intelligence & industrial applications* (pp. 462–470). Springer.

Mahmoudi, O., & Bouami, M. F. (2023). Arabic speech commands recognition with LSTM & GRU models using CUDA toolkit implementation. In *2023 3rd international conference on innovative research in applied science, engineering and technology (IRASET)* (pp. 1–4). IEEE.

Mahmoudi, O., & Bouami, M. F. (2023). Arabic speech emotion recognition using deep neural network. In *International conference on digital technologies and applications* (pp. 124–133). Springer.

Mushtaq, Z., & Su, S. F. (2020). Environmental sound classification using a regularized deep convolutional neural network with data augmentation. *Applied Acoustics, 167*, 107389.

Niu, Z., Zhong, G., & Yu, H. (2021). A review on the attention mechanism of deep learning. *Neurocomputing, 452*, 48–62.

Obaid, M., Hodrob, R., Abu Mwais, A., & Aldababsa, M. (2023). Small vocabulary isolated-word automatic speech recognition for single-word commands in Arabic spoken. *Soft Computing, 78*, 1–14.

Owens, J. D., Houston, M., Luebke, D., Green, S., Stone, J. E., & Phillips, J. C. (2008). GPU computing. *Proceedings of the IEEE, 96*(5), 879–899.

Pervaiz, A., Hussain, F., Israr, H., Tahir, M. A., Raja, F. R., Baloch, N. K., & Zikria, Y. B. (2020). Incorporating noise robustness in speech command recognition by noise augmentation of training data. *Sensors, 20*(8), 2326.

Salamon, J., & Bello, J. P. (2017). Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters, 24*(3), 279–283.

Usman, M., Zia, T., & Tariq, A. (2022). Analyzing transfer learning of vision transformers for interpreting chest radiography. *Journal of Digital Imaging, 35*(6), 1445–1462.

Zheng, F., Zhang, G., & Song, Z. (2001). Comparison of different implementations of MFCC. *Journal of Computer Science and Technology, 16*, 582–589.