Penerbit Akademia Baru

Journal of Advanced Research Design

Journal homepage:
https://akademiabaru.com/submit/index.php/ard
ISSN: 2289-7984

Application of Genetic Algorithm in Training Automatic Speech Recognition

Dilnoz Muxamediyeva Tulkunovna[1], Nilufar Niyozmatova A'loxanovna[2,*]

[1] Digital technologies and Artificial Intelligence, Tashkent Institute of Irrigation and Agricultural Mechanization Engineers, National Research University, 100000 Tashkent, Uzbekistan
[2] Institute of Fundamental and Applied Research, Tashkent Institute of Irrigation and Agricultural Mechanization Engineers, National Research University, 100000 Tashkent, Uzbekistan

ARTICLE INFO

Article history:
Received 23 July 2024
Received in revised form 23 December 2024
Accepted 7 April 2025
Available online 25 April 2025

Keywords:
Automatic speech recognition; hidden Markov models; mel-cepstral coefficients; fast Fourier transform; expectation-maximization algorithm; Baum-Welch algorithm

ABSTRACT

The application of a genetic algorithm in training automatic speech recognition is considered. In the learning process, hidden Markov models are used to evaluate the statistical properties of each word, including the sequence of cepstral coefficients, as well as the transition probabilities between model states. As the model trains, it seeks to improve the fit between the input cepstral coefficients and the target words in order to improve the accuracy of speech recognition. Once trained, the system is used to recognize new speech inputs and identify the corresponding words. This is done by applying trained Hidden Markov Models to new sequences of cepstral coefficients. Genetic algorithms can be useful for optimizing some aspects of speech recognition systems, such as selecting optimal parameters for processing speech signals and choosing the most appropriate models for specific tasks. The results of a comparative analysis based on typical examples are presented, demonstrating the advantages and high efficiency of the genetic algorithm. The main advantages of the GA are indicated, such as independence from initial conditions and the ability to overcome local extrema, which makes it a promising tool for training speech recognition models. The conclusion emphasizes the significance of the research results and their contribution to the development of SMM training methods for speech recognition.

1. Introduction

Speech is one of the most natural and fastest ways of communication between people. Computer speech recognition systems make it possible to use this method of communication and interaction between a person and a computer. They find applications in various fields such as medicine, automotive, telecommunications, audiovisual technology, natural language processing and many others [1,2]. The development of speech recognition technologies in recent years has significantly improved the quality of recognition and increased the usability of such systems [3].

The first electronic machine to synthesize English speech was presented at a trade show in New York in 1939 and was called voder. Voder, introduced in 1939, was one of the first speech synthesis

* *Corresponding author*
E-mail address: n_niyozmatova@mail.ru

https://doi.org/10.37934/ard.127.1.115

devices. He used electronic components and mechanical devices to create sounds, but the quality of the speech produced was limited.

The first speech recognition device capable of recognizing numbers was developed in 1952. This was an important breakthrough in the field of automatic speech recognition, although its capabilities were limited and it could only work with a limited set of phrases [3-5].

Since then, speech synthesis and recognition technologies have continued to evolve and today we have better systems capable of generating speech, as well as accurately recognizing and interpreting speech signals in various contexts and languages [6]. These technologies have a wide range of applications and continue to evolve every year.

Early in the development of speech recognition, a system was created that could recognize numbers (0 to 9) based on dynamic time programming. It was one of the first successful speech recognition systems to demonstrate the feasibility of using statistical methods for this task. In the mid-1970s, HMMs were first used for speech recognition tasks. This led to significant improvements in recognition accuracy and paved the way for further development in this area.

In the late 1980s and early 1990s, various companies such as IBM and Dragon Systems began producing commercially available speech recognition systems. This was an important step in the spread and application of speech recognition technology in various fields, including medicine, telecommunications and document management. Over the past few years, voice assistants such as Apple's Siri, Amazon's Alexa and Google Assistant have become increasingly popular. These systems are based on advanced speech recognition algorithms, allowing users to interact with devices and perform various tasks using voice commands [5].

The presence of acoustic interference and distortion is another problem. Speech may be distorted by various acoustic interferences such as noise, echo, voice trembling, etc., which may reduce recognition accuracy. The presence of speech interference can also become a problem in speech recognition. Speech interference can include other voices, noises, music, etc., which can also make recognition difficult [7].

Despite significant improvements in speech recognition accuracy, the technology may still have limitations in recognizing different accents, dialects or non-standard speech, which can lead to errors and misunderstandings. Speech recognition technology may be less effective in noisy or unpredictable environments, such as on the street or in vehicles. This may reduce its applicability in certain use cases.

Many existing speech recognition systems are designed primarily for specific languages and cultural contexts, limiting their applicability in multilingual or multicultural societies. Some advanced speech recognition algorithms may be computationally intensive, which may be a barrier to use on mobile devices or in environments where computing power is limited.

Considering these criticisms and limitations helps to understand the current challenges in speech recognition and direct efforts to overcome them to create more reliable and efficient systems.

The positive aspects of this technology are the improved accessibility. Speech recognition can greatly improve the accessibility of information for people with disabilities, such as low vision or blind people. This technology allows such people to interact with devices and receive information using only their voice. Voice interfaces based on speech recognition make interaction with technology and software applications more natural and convenient. Instead of typing text or controlling devices with a keyboard or mouse, the user can simply speak. In some fields, such as medicine or law, where a large amount of documentation or recording is required, speech recognition technology can significantly improve productivity by automating the transcription or recording process. Speech recognition is driving the development of new innovative applications and services, such as voice assistants, automated dialling systems or speech-based health monitoring systems.

Hidden Markov Model (HMM)-based speech recognition systems are widely used in voice interfaces and voice assistants to issue commands, ask questions, retrieve information and control devices. Such technologies are becoming increasingly in demand in everyday life, providing convenience and efficiency of interaction with devices.

HMM-based speech recognition plays an important role in the development of learning technologies, including adaptive learning systems, voice learning and automatic pronunciation assessment. This allows for the creation of innovative educational platforms that can adapt to the needs of each student and provide more effective learning.

HMM-based speech recognition is also used in biometric authentication systems such as voice authentication systems and access control systems. This helps provide a high level of security and convenience in a variety of applications, including banking, secure facilities and information systems.

A HMM is a statistical model that is used to describe a sequence of observations, where each observation is associated with some hidden state. In the context of speech recognition, observations can be sound fragments (such as phonemes) and hidden states can be phonemes or words that are spoken. The key idea is that although we cannot directly observe hidden states, we can observe the results of their actions (observations) and, based on this data, make inferences about hidden states and transitions between them. HMMs are widely used in speech recognition systems due to their ability to model sequences of data and take into account the temporal dependencies between them. They also have their own variations and extensions that take into account various aspects of speech, such as variability in pronunciation rate, acoustic conditions, etc.

They allow you to model the relationship between phoneme sequences and acoustic features using a statistical approach.

The main idea of HMM is that the observed sequence of acoustic features (for example, a spectrogram) is generated by a hidden sequence of states that model phonemes. In this case, the transition probabilities between states and the probabilities of generating each acoustic feature in each state are predetermined [8].

HMMs are trained on large speech corpora, where the corresponding sequence of acoustic features is known for each phoneme. After training, the model can be used for speech recognition, in which the input sequence of acoustic features is converted into a sequence of latent states and then used to determine the most probable sequence of phonemes [9].

HMMs have a number of advantages, such as the ability to model relationships between phonemes and acoustic features, resistance to various forms of noise and interference and the ability to adapt to new speakers and recording conditions. However, they also have disadvantages, such as the difficulty of training and high computational complexity in recognition [10].

As for genetic algorithm learning, this approach also remains relevant in the context of speech recognition technology and other areas of artificial intelligence. Genetic algorithms can be used to optimize HMM parameters, select optimal models and improve recognition quality. This approach allows us to create more efficient and adaptive speech recognition systems that can achieve high accuracy even in difficult conditions.

Four isolated trends in speech technology development represent key aspects of research and development in this area. Speech recognition focuses on converting the speech acoustic signal into a string of characters or words. Speech recognition systems can vary in many ways, such as vocabulary size (from a few words to thousands), number of speakers, pronunciation style (isolated commands, continuous speech), branching ratio, signal-to-noise ratio and quality of communication channels. The performance of systems is usually assessed by the reliability of word recognition or the percentage of errors. Determining a speaker's identity involves verifying and identifying the speaker. Verification systems confirm the identity of the speaker and identification systems determine his

identity among a pre-limited number of people. They can be text-dependent or text-independent and are assessed by the probability of not recognizing “one’s own” speaker and the probability of mistaking a “foreign” speaker for one’s own. Speech synthesis includes playback of recorded messages and synthesis of speech from text. Speech synthesizers are characterized by intelligibility, natural sound and noise immunity. The main criterion for classifying speech compression systems is the degree of compression, which can vary from low to high. The performance of such systems is usually assessed by the intelligibility of compressed speech.

These systems can be characterized by a number of parameters. First of all, this is the volume of the dictionary: small volumes are up to 20 words, large ones are thousands and tens of thousands. Number of speakers: from one to arbitrary. Pronunciation style: from isolated commands to continuous speech and from reading to spontaneous speech. Branching factor, i.e. a value that determines the number of hypotheses at each recognition step: from small values (<10-15) to large ones (~100-200). Signal-to-noise ratio from high (>30 dB) to low (<10 dB). Quality of communication channels: from a high-quality microphone to a telephone channel. The quality of speech recognition systems is usually characterized by the reliability of word recognition or, what is the same, by the percentage of errors.

2. Methodology

The process of speech recognition in an automatic speech recognition (ASP) system can be divided into three main stages: feature extraction, training and recognition.

At the first stage, feature extraction, the main task is to obtain a compact and informative representation of the speech signal. Various methods are used for this, such as mel-cepstral coefficients, linear prediction coefficients (LPC) and others. These methods make it possible to extract important speech characteristics, such as formants and spectral features and represent them as a feature vector [11].

The algorithm for obtaining MFCC (Mel-frequency cepstral coefficients) for speech analysis includes the following steps:

i. Splitting an audio signal into small fragments (usually 20-30 ms long) called windows.
ii. Overlay each window with a window function to reduce the effect of overlapping neighbouring windows.
iii. Applying a Fast Fourier Transform (FFT) to each window to obtain an amplitude spectrum.
iv. Applying the Mel scale to the amplitude spectrum, which is a set of filters spaced along the Mel scale.
v. Calculating the logarithm of the energy of each filter.
vi. Applying the inverse cosine transform to the logarithms of the energy to get the MFCC set.
vii. Normalizing an MFCC set by removing the mean and scaling by standard deviation.

As a result, a set of MFCC coefficients is obtained, which can be used for training and speech recognition using hidden Markov models [12].

To obtain the MFCC, the discrete cosine transform is used according to the following algorithm:

i. The discrete Fourier transform for the frame is calculated as in Eqs. (1) - (4):

$$X_t(m) = \sum_{k=0}^{K-1} x_t(k) \cdot e^{\frac{-j2\pi mk}{K}}. \quad (1)$$

Where,

$$x_t(k) = s'(k + t \cdot K) \cdot w(k),\ 0 \le k \le K-1, \quad (2)$$

$$s'(n) = s(n) - 0.95 \cdot s(n-1), \quad (3)$$

$$w(k) = 0.54 - 0.46 \cdot \cos\left(\frac{2\pi k}{K-1}\right). \quad (4)$$

ii. Select the central frequencies F_q of Mel-filters in Eq. (5):

$$F_{mel} = 2095 \cdot log_{10}\left(1 + \frac{F_{Hz}}{700}\right). \quad (5)$$

iii. Transition from Mel scale to traditional scale applying Eq. (6):

$$F_{Hz} = 700 \cdot \left(10^{\frac{F_{mel}}{2595}} - 1\right). \quad (6)$$

iv. Construction of amplitude-frequency characteristics of filters (Figure 1) using Eq. (7):

$$U_q(k) = \begin{cases} 1 - \frac{|k - F_q|}{\Delta_q},\ |k - F_q| < \Delta_q, \\ 0,\ |k - F_q| \ge \Delta_q. \end{cases} \quad (7)$$

v. The frame energy value Q is calculated by applying Mel-filters and referring to Eq. (8):

$$Y_t(q) = \sum_{k=0}^{K-1} |x_t(k)| \cdot U_q(k) \quad (8)$$

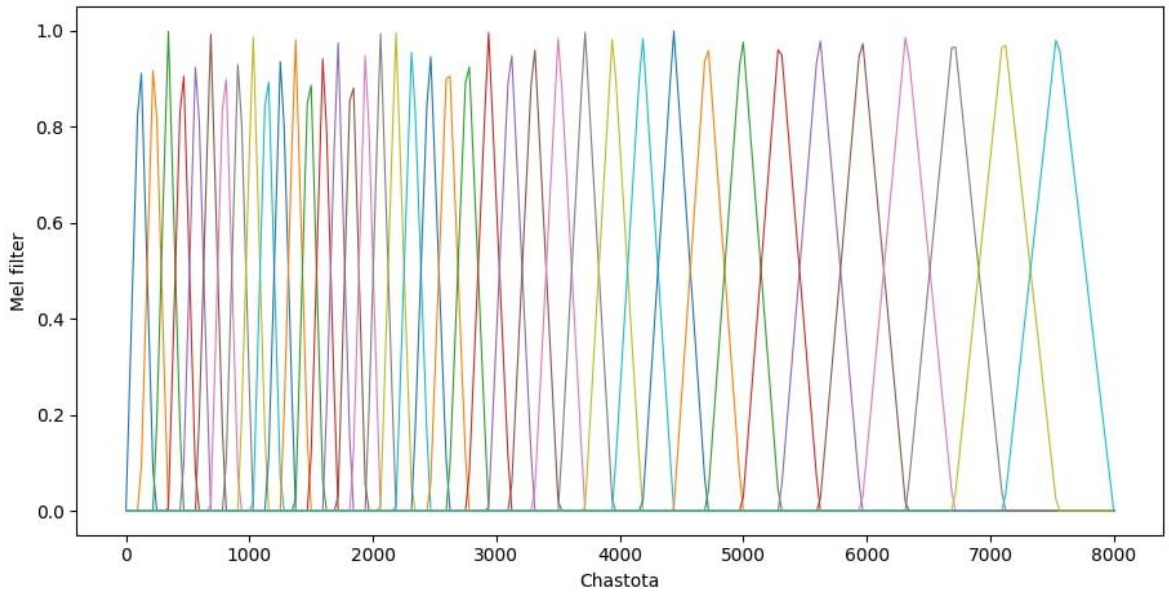

Fig. 1. Amplitude-frequency characteristics of Mel filters q=24

vi. The logarithm of the modulus is calculated, $Y_t(q)$ applying Eq. (9):

$$L_t(q) = log|Y_t(q)|. \quad (9)$$

vii. An inverse discrete Fourier transform is performed, which is equivalent to the discrete cosine transform in Eq. (10):

$$o_t(p) = \sum_{q=1}^{Q} L_t(q) \cdot \cos\left(p \cdot \left(q - \frac{1}{2} \right) \cdot \frac{\pi}{Q} \right), p = 0, \ldots P \quad (10)$$

Thus, the recognizer receives a p-dimensional vector $o_t(p)$ containing cepstral coefficients for the t-[th] frame (Figure 2).

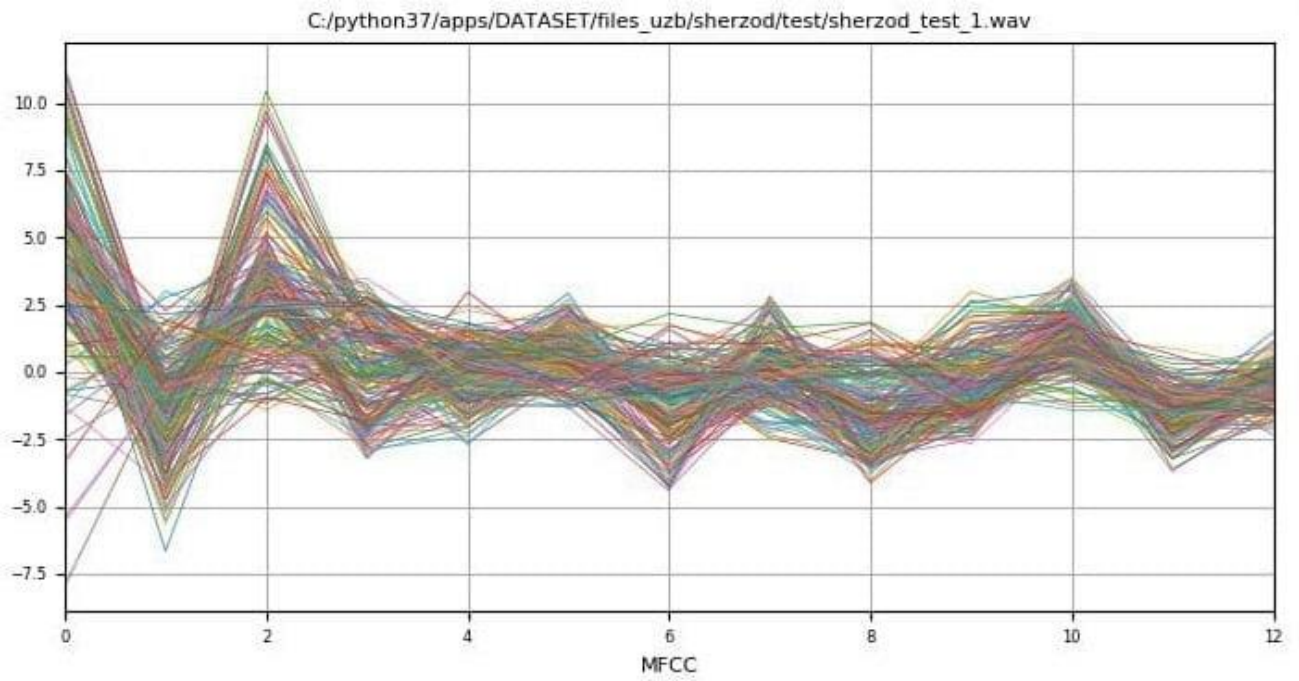

Fig. 2. MFCC of one frame

MFCCs act as characteristics of the input data for HMMs in speech recognition systems. HMMs use the information provided by the MFCC to estimate the likelihood of different sequences of phonemes or words, making them important components in the architecture of automatic speech recognition systems. After the windowed Fourier transform and the logarithm of the spectral coefficients, a discrete cosine transform is performed, which allows you to compress the information by highlighting the most significant coefficients. Usually only the first 12-13 coefficients are used; the rest are discarded. The result is the MFCC vector, which is used as features for speech recognition based on CMM [13].

Let us consider a system that can be in one of N different states $S_1, S_2, \ldots, S_N$ an arbitrary moment in time. HMM are used to model the output of some hidden process. In this case, the hidden process is modelled as a sequence of states that change over time in accordance with the probabilistic transitions between them. Each state of the hidden process generates some observation, which

becomes visible to the observer. The probabilities of state transitions and the probabilities of generating observations are determined by the model parameters, which can be estimated from the observed data using training methods. In such models, the observed events are probabilistic functions of the real state of the system. In HMM, the observed events are called "observable variables" or "outputs" and the states of the system that are not directly observed are called "hidden variables" or "internal states". The output is the result of some process that depends on the internal state of the system. The probability of observing each specific sequence of output data for a given sequence of internal states is determined using conditional probabilities [14].

Markov models can be described by the following set of parameters:

i. The set of system states $S_1, S_2, \ldots, S_N$ is a finite set of states that the system can take.
ii. The set of possible observations $V = \{V_1, V_2, \ldots V_N\}$ is the finite set of observable events that can occur in each state of the system.
iii. The matrix of transition probabilities $A = \{p_{ij}\}$ is a matrix, where each element $p_{ij} = p\left[q_{t+1} = S_j \middle| q_t = S_i\right], 1 \le i, j \le N$ determines the probability of transition from state i to state j.

Where,
$p_{ij} \geq 0$,
$\sum_{j=1}^{N} p_{ij} = 1$

iv. The observation probability matrix $G = \{g_j(k)\}$ is a matrix where each element $g_j(k) = p[v_k \middle| q_t = S_j], 1 \le j \le N, 1 \le k \le M$ determines the probability of observing an event k in state j.
v. The initial probability distribution $\Pi_i = P[q_1 = S_i], 1 \le i \le N$ is a vector, where each element determines the probability of the system initially being in state *i*.

To generate a sequence with a given number of elements, you must specify the initial state of the system (that is, one of the possible sequences of observable data, which are states that the hidden process can take) and sequentially generate new states and corresponding observable events. This can be done like this:

i. Set the initial state of the system by selecting one of the possible states of the hidden process.
ii. Using the matrix of transition probabilities A, select the next state of the system based on the current state.
iii. Using the probability matrix G, generate an observable event corresponding to the current state.
iv. Repeat steps 2 and 3 the specified number of times (until the desired sequence length is reached).

Thus, it is possible to generate a sequence of observed events that corresponds to a given statistical model. This sequence can be used to test speech recognition algorithms.

Three problems that need to be solved when using HMM for pattern recognition:

i. The task of training the model: it is necessary to determine the parameters of the HMM based on the training set. For this, the EM-algorithm (Expectation-Maximization algorithm) is used.
ii. The task of estimating the probability of observation: given a HMM and an observed sequence. It is necessary to determine the probability of observing this sequence in the given model. For this, the Forward-Backward algorithm is used.
iii. Decoding task: given a HMM and an observed sequence. It is necessary to determine the most probable sequence of hidden states. For this, the Viterbi algorithm is used.

To calculate the probability of observations for given model parameters, the forward algorithm is used. This algorithm allows you to recursively calculate the probability of observations for each possible state of the system at time t, taking into account all previous observations.

The forward propagation algorithm can be written as follows:

i. Initialization:
- calculate the initial probabilities of the distribution of the system states at the moment of time;
- calculate the probabilities of the observations in the initial state.

ii. Recursion:
- calculate the probability of observation for each time t > 0 and each state j;
- calculate the probability of observing in state j at time t;
- calculate the probability of a path that ends in state j at time t.

iii. Calculation of the probability of observation:
- calculate the probability of observation given the model parameters.

Thus, the forward propagation algorithm allows one to calculate the probability of observations for given model parameters in linear time from the length of the sequence of observations.

To determine the corresponding sequence of internal states in HMM, there is a forward-backward algorithm.

The algorithm consists of two phases: forward (forward pass) and backward (backward pass). On the forward run, the probabilities are calculated that the model has passed into a certain hidden state, taking into account all the observed values up to this point. On the backward pass, the probabilities are calculated that the model has passed into a certain hidden state, taking into account all the observed values after this moment.

The output is a sequence of probabilities for each hidden state at each point in time, which can be used to determine the corresponding sequence of internal states.

The forward-backward algorithm is efficient and widely used in speech recognition, natural language processing, computer vision and other fields related to sequence analysis.

To determine the parameters of the model, a learning algorithm is used, which is based on the criterion of maximizing the likelihood (maximum likelihood estimation, MLE).

The idea is to find the values of the model parameters at which the probability of obtaining the observed data is maximum. Formally, it is necessary to maximize the likelihood function L, which is expressed in terms of the probability of observing a given sequence X for given model parameters ϑ:

$$L(\theta) = P(X|\theta)$$

In the case of HMM, the calculation of the likelihood function L can be performed using the forward-backward algorithm, which solves the problem of determining the sequence of internal states of the model corresponding to the observed sequence.

After finding the likelihood function L, the problem is reduced to finding the values of the model parameters at which it is maximum. For this, various optimization methods are used, such as gradient descent, Newton's method, etc.

As a result of the execution of the learning algorithm, the values of the model parameters are obtained, which maximize the probability of observing a given sequence with the given model parameters. These parameters can then be used to recognize new data sequences.

In the second stage, training, the ASR system is trained to recognize speech using training data. Training involves creating models for each phoneme or word that the system needs to recognize. Training can be done using various methods, including maximum likelihood methods and deep learning methods.

The Baum-Welch method and the EM algorithm are standard methods for solving the learning problem for HMM.

The Baum-Welch algorithm is used to estimate the parameters of a HMM given observations. It is one of the standard methods for solving the HMM learning problem. The main idea of the Baum-Welch method is to calculate the posterior probabilities of states of a HMM given observations using the forward-backward algorithm. Then, based on these probabilities, estimate the parameters of the model using the maximum likelihood estimate (maximum likelihood estimation).

The Baum-Welch algorithm consists of several steps:

i. Initialization of the model parameters, for example, randomly.
ii. Calculation of a posteriori probabilities of states of a HMM under given observations using the forward-backward pass algorithm.
iii. Calculate the expectation of numerators and denominators for the model parameters, using the computed posterior probabilities of states and observations.
iv. Update the model parameters based on the expected numerators and denominators.
v. Repeat steps 2-4 until convergence.

However, the Baum-Welch method does not always guarantee convergence to the global maximum of the likelihood function, but can stop at the local maximum. Therefore, to improve the efficiency of HMM training, various modifications and combinations of methods, as well as various strategies for initializing model parameters, can be applied.

The Expectation-Maximization algorithm is also a standard method for solving the problem of learning HMMs. This algorithm allows estimation of HMM parameters based on incomplete data, that is, data in which some variables are unknown.

The Expectation-Maximization algorithm consists of two steps: the E-step (Expectation step) and the M-step (Maximization step).

At the E-step, the posterior probabilities of hidden variables (states) are calculated for given observations. Then, at the M-step, using the calculated posterior probabilities, the HMM likelihood function is maximized with respect to the model parameters.

Step E: Using the current HMM parameters, compute the probabilities of the hidden variables given the observations. For this, the forward-backward algorithm is used.

Step M: Using the latent variable posteriors computed in the E-step, maximize the HMM likelihood function over the model parameters using standard optimization techniques such as gradient descent.

Steps E and M are repeated until convergence.

The Expectation-Maximization algorithm also has its drawbacks, including the possibility of getting stuck in local optima and dependence on the initial parameters of the model. Therefore, to improve the efficiency of HMM training, various modifications and combinations of methods, as well as various strategies for initializing model parameters, can be applied.

However, these methods may fall into the local optimum and depend on the starting parameters. Therefore, to improve the efficiency of HMM training, various modifications and combinations of methods are used, as well as various strategies for initializing model parameters.

One of the main problems in HMM training is the problem of getting into the local optimum in the process of model optimization. This can cause the model optimization to stop at a loss function value that is not the global optimum and therefore the model will not have the best recognition accuracy.

To overcome this problem, there are several approaches. One of them is the use of stochastic optimization methods such as Markov chain Monte Carlo methods (MCMC) or stochastic gradient descent (SGD). These methods make it possible to get out of the local optimum and provide a wider search for the parameter space.

In addition, global optimization methods such as genetic algorithms or ant colony based global optimization methods or other evolutionary algorithms can be used. These methods also avoid local optima. In our case, genetic algorithms were used to avoid local optima.

In the third stage, recognition, the ASR system uses the trained models to recognize the speech of the input signal. Recognition is carried out by comparing the feature vector of the input signal with the models created at the previous stage. As a result of recognition, the system produces the most probable text corresponding to the input speech signal.

Thus, each of the three stages in the speech recognition process is important and requires certain methods and algorithms for its implementation.

To solve a speech recognition problem, the ASR system must determine which words were spoken in an acoustic sequence, that is, find the corresponding chain of words. This problem is solved by modelling the speech process and calculating the probability of occurrence of each chain of words for a given acoustic sequence.

3. Results

In speech recognition systems, input words are usually converted into a sequence of vectors of cepstral coefficients. This is done using a feature extraction process such as MFCC from an acoustic signal.

After extracting the cepstral coefficients, the system can be trained using HMMs for the selected words. The HMM is a probabilistic model that models the statistical properties of a sequence of sounds and associated character (word) sequences.

During training, HMM can be used to evaluate the statistical properties of each word, including the sequence of cepstral coefficients, as well as the transition probabilities between model states. As the model trains, it seeks to improve the fit between the input cepstral coefficients and the target words in order to improve the accuracy of speech recognition.

Once trained, the system can be used to recognize new speech inputs and identify the corresponding words. This is done by applying trained HMM to new sequences of cepstral coefficients. Genetic algorithms can be useful for optimizing some aspects of speech recognition systems, such as selecting optimal parameters for processing speech signals and choosing the most appropriate models for specific tasks.

Optimizing the transition probability matrix of a HMM is one of the tasks that can be solved using optimization methods.

The transition probability matrix determines the transition probabilities between the states of the model and this matrix must be optimally tuned in order for the model to correctly recognize speech. In particular, the transition probability matrix can be optimized to improve speech recognition accuracy, speed up recognition speed and reduce recognition errors.

One approach to optimizing the HMM transition probability matrix is to use an evolutionary algorithm to determine the best transition probability matrix based on speech recognition quality scores. Within this approach, random transition probability matrices are generated and used for speech recognition and then the recognition quality is evaluated. This process is repeated many times in order to find the best transition probability matrix.

The choice of optimization method for optimizing the HMM transition probability matrix will depend on the specific task and the required accuracy and speed of speech recognition. Each method has its advantages and disadvantages, which may influence the choice of optimization method.

For example, genetic algorithms can be efficient in finding the optimal transition probability matrix, but may require more time and resources than other optimization methods. Gradient descent optimization can be faster and more efficient for small matrix sizes, but can run into local minimum issues.

To write a speech recognition program based on HMM using a genetic algorithm, you will need to use the genetic algorithm to optimize the model parameters.

The following is a sample Python code that demonstrates how to use a genetic algorithm to train HMM for speech recognition:

- import librosa
- import pyaudio
- import numpy as np
- from hmmlearn import hmm
- from deap import algorithms, base, creator, tools

3.1 Defining a Constant to Set Model Parameters

```
n_mfcc = 12            # Number of MFCC coefficients
n_iterations = 1000    # Number of training iterations
```

3.2 Creating a Hidden Markov Model and Determining the Function We Will Optimize

- defcreate_model(n_states):
  model = hmm.GaussianHMM(n_components=n_states,
- covariance_type='diag', n_iter=n_iterations)
  return model
- deffitness_function(individual):
- n_states = individual[0]
  model = create_model(n_states)
- model.fit(mfcc.T)
- logprob, predicted_labels = model.decode(mfcc.T)
- returnlogprob,

3.3 Determining the Parameters of the Genetic Algorithm

- creator.create("FitnessMax", base.Fitness, weights=(1.0,))
- creator.create("Individual", list, fitness=creator.FitnessMax)
- toolbox = base.Toolbox()
- toolbox.register("attr_n_states", np.random.randint, low=2, high=10)
- toolbox.register("individual", tools.initRepeat, creator.Individual, toolbox.attr_n_states, n=1)
- toolbox.register("population", tools.initRepeat, list, toolbox.individual)
- toolbox.register("evaluate", fitness_function)
- toolbox.register("mate", tools.cxUniform, indpb=0.5)
- toolbox.register("mutate", tools.mutUniformInt, low=2, high=10, indpb=0.1)
- toolbox.register("select", tools.selTournament, tournsize=3)

3.4 Loading Training Data from an Audio File

- audio_data, sample_rate = librosa.load('training.wav', sr=None)

3.5 Extracting MFCC Coefficients from Audio Data

- mfcc = librosa.feature.mfcc(audio_data, sr=sample_rate, n_mfcc=n_mfcc)

3.6 Determining the Parameters of the Genetic Algorithm

- population_size = 20
- num_generations = 10

3.7 Running a Genetic Algorithm to Train Hidden Markov Models

- population = toolbox.population(n=population_size)
- for generation in range(num_generations):
  offspring = algorithms.varAnd(population, toolbox, cxpb=0.5, mutpb=0.1)
  fits = toolbox.map(toolbox.evaluate, offspring)
  for fit, ind in zip(fits, offspring):
- ind.fitness.values = fit
  population = toolbox.select(offspring, k=len(population))

3.8 Creation of the Initial Population and Launch of the Genetic Algorithm

- population = toolbox.population(n=population_size)
  for generation in range(num_generations): offspring = algorithms.varAnd(population, toolbox, cxpb=0.5, mutpb=0.1) fits = toolbox.map(toolbox.evaluate, offspring) for fit, ind in zip(fits, offspring): ind.fitness.values = fit population = toolbox.select(offspring, k=len(population))

3.9 Obtaining the Best Model from the Population and Deriving Its Parameters

- best_model = None best_fitness = -np.inf for individual in population: fitness = fitness_function(individual) if fitness >best_fitness: best_fitness = fitness n_states = individual[0] best_model = create_model(n_states) best_model.fit(mfcc.T) print("Best fitness:", best_fitness) print("Best number of states:", n_states).

3.10 Loading Test Data from an Audio File

- audio_data, sample_rate = librosa.load('testing.wav', sr=None)
- Extract MFCC coefficients from audio data
- test_mfcc = librosa.feature.mfcc(audio_data, sr=sample_rate, n_mfcc=n_mfcc)

3.11 Speech Recognition with a Better Model

- logprob, predicted_labels = best_model.decode(test_mfcc.T) print("Predicted labels:", predicted_labels)

The results of the genetic algorithm were compared with the traditional method of training HMM using the Baum-Welch method. Here are examples of two transition matrices for the HMM model. Number of model states N=7):

N:7

F:9,48188122599803E-174

0,94555553 0,01518084 0,03714327 0,00000000 0,00000000 0,00000000 0,00000000
0,00000000 0,93317462 0,03346314 0,01532132 0,00000000 0,00000000 0,00000000
0,00000000 0,00000000 0,93555553 0,02777766 0,02777555 0,00000000 0,00000000
0,00000000 0,00000000 0,00000000 0,92413284 0,02632237 0,02753482 0,00000000
0,00000000 0,00000000 0,00000000 0,00000000 0,93934543 0,02434543 0,01999234
0,00000000 0,00000000 0,00000000 0,00000000 0,00000000 0,95121212 0,02233232
0,00000000 0,00000000 0,00000000 0,00000000 0,00000000 0,00000000 0,96232445

N:7

F:8,16437583955457E-171

0,92211335 0,07346445 0,00000000 0,00000000 0,00000000 0,00000000 0,00000000
0,00000000 0,91956776 0,08154565 0,00000000 0,00000000 0,00000000 0,00000000
0,00000000 0,00000000 0,92342444 0,07123454 0,00000000 0,00000000 0,00000000
0,00000000 0,00000000 0,00000000 0,93123445 0,0662312344,00000000 0,00000000
0,00000000 0,00000000 0,00000000 0,00000000 0,9293456435,07786386 0,00000000
0,00000000 0,00000000 0,00000000 0,00000000 0,00000000 0,94234265 0,05823164
0,00000000 0,00000000 0,00000000 0,00000000 0,00000000 0,00000000 1,00000000

This displays the contents of the transition matrix before and after the genetic algorithm runs. After the genetic algorithm works, we observe changes in the transition matrix of the model. The original model, characterized by a limited number of transitions, has become even more strict: now transitions are made only to the next state. This led to the creation of a model that was optimized for a specific word instance and the probability of generating it became significantly higher than that of the original model, achieving a significant increase. As a result, we received a model that was highly optimized for a specific instance of a word, the probability of generating which is higher than that of the original model by several orders of magnitude: F: 8.16437583955457E-171 versus F: 9.48188122599803E-174. This result is very favourable and indicates the success of model optimization using a genetic algorithm.

4. Conclusions

Genetic algorithms work well when starting with a randomly generated population. This is because the genetic algorithm iteratively improves the solution by combining and mutating the best candidates from the previous population. A randomly generated population provides diversity in initial candidates, which can lead to a broader search and better results. However, more iterations may be required to reach the best solution. When choosing an initial population, the limitations and requirements of a particular task should be taken into account. For example, for the problem of optimizing the HMM transition probability matrix in speech recognition systems, the initial population can be generated randomly, but it is necessary to take into account the restrictions associated with the transition probabilities and the general properties of the HMM. As a result of comparing the results of the genetic algorithm with the traditional method of training HMM using the Baum-Welch method, the most typical examples are given that demonstrate the high efficiency of the genetic algorithm in solving the problem. Certain advantages of the genetic algorithm over the Baum-Welch method are noted, such as independence from initial conditions and the ability to escape from local extrema. This allows you to increase the efficiency of the learning process and improve the quality of the speech recognition model. Thus, the results of the study indicate the promise of using a genetic algorithm in the field of training HMM for speech recognition. The use of a genetic algorithm to optimize the speech recognition model led to a significant improvement in its efficiency. The initially limited model, after the algorithm worked, became more accurate and adapted to a specific instance of the word. This manifested itself in a significant increase in the probability of generating a given word compared to the original model. Such positive changes indicate the effectiveness of the genetic algorithm in optimizing speech recognition models. This approach promises significant improvements in the accuracy and reliability of speech recognition systems, which is an important step in the development of this technology.

References

[1] Al-Fraihat, Dimah, Yousef Sharrab, Faisal Alzyoud, Ayman Qahmash, Monther Tarawneh, and Adi Maaita. "Speech recognition utilizing deep learning: A systematic review of the latest developments." *Human-centric computing and information sciences* 14 (2024).

[2] Sivakumar, Balakrishnan and Biligirirangaiah, Praveen Kadakola. “Autoregressive Based Vocal Tract Shape Modelling of Vowels in Speech Processing.” *Journal of Advanced Research in Applied Sciences and Engineering Technology 32*, no. 1 (2023): 32–45. https://doi:10.37934/araset.32.1.3245

[3] Shohan, Mehedi Hasan, Kazi Rifat Ahmed, Nur Farhan Kahar, Nusrat Jahan and Md Maruf. "Use of Natural Language Processing for the Detection of Hate Speech on Social Media." *Journal of Advanced Research in Applied Sciences and Engineering Technology* 51, no. 2 (2025): 86-96. https://doi.org/10.37934/araset.51.2.8696

[4] Mamatov, N. S., N. A. Niyozmatova, A. N. Samijonov and B. N. Samijonov. "Construction of language models for Uzbek language." In *2022 International Conference on Information Science and Communications Technologies (ICISCT)*, pp. 1-4. IEEE, 2022. https://doi.org/10.1109/ICISCT55600.2022.10146788

[5] Mamatov, N. S., N. A. Niyozmatova, Sh Sh Abdullaev, A. N. Samijonov and K. K. Erejepov. "Speech recognition based on transformer neural networks." In *2021 International Conference on Information Science and Communications Technologies (ICISCT)*, pp. 1-5. IEEE, 2021. https://doi.org/10.1109/ICISCT52966.2021.9670093

[6] Mamatov, Narzillo S., N. A. Niyozmatova, Yu Sh Yuldoshev, Sh Sh Abdullaev and Abdurashid N. Samijonov. "Automatic speech recognition on the neutral network based on attention mechanism." In *International Conference on Intelligent Human Computer Interaction*, pp. 100-108. Cham: Springer Nature Switzerland, 2022. https://doi.org/10.1007/978-3-031-27199-1_11

[7] Niyozmatova, N. A., N. S. Mamatov, Sh A. Tulyaganova, A. N. Samijonov and B. N. Samijonov. "Methods for determining speech activity of uzbek speech in recognition systems." In *AIP Conference Proceedings*, vol. 2789, no. 1. AIP Publishing, 2023. https://doi.org/10.1063/5.0145438

[8] Alimuradov, A. K. and P. P. Churakov. "Review and classification methods for processing speech signals in the speech recognition systems." *Izmerenie. Monitoring. Upravlenie. Kontrol* (2015): 27-35.

[9] Todkar, Satyam P., Snehal S. Babar, Rudrendra U. Ambike, Prasad B. Suryakar and J. R. Prasad. "Speaker recognition techniques: A review." In *2018 3rd International Conference for Convergence in Technology (I2CT)*, pp. 1-5. IEEE, 2018. https://doi.org/10.1109/I2CT.2018.8529519

[10] Subhashree, R. and G. N. Rathna. "Speech emotion recognition: performance analysis based on fused algorithms and GMM modelling." *Indian Journal of Science and Technology* 9, no. 11 (2016): 1-8. https://doi.org/10.17485/ijst/2016/v9i11/88460

[11] Sithara, A., Abraham Thomas and Dominic Mathew. "Study of MFCC and IHC feature extraction methods with probabilistic acoustic models for speaker biometric applications." *Procedia computer science* 143 (2018): 267-276. https://doi.org/10.1016/j.procs.2018.10.395

[12] Barai, Bidhan, Debayan Das, Nibaran Das, Subhadip Basu and Mita Nasipuri. "An ASR system using MFCC and VQ/GMM with emphasis on environmental dependency." In *2017 IEEE Calcutta Conference (CALCON)*, pp. 362-366. IEEE, 2017. https://doi.org/10.1109/CALCON.2017.8280756

[13] Kenny, Patrick, Pierre Ouellet, Najim Dehak, Vishwa Gupta and Pierre Dumouchel. "A study of interspeaker variability in speaker verification." *IEEE Transactions on Audio, Speech and Language Processing* 16, no. 5 (2008): 980-988. https://doi.org/10.1109/TASL.2008.925147

[14] Aida-Zade, K. R., Cemal Ardil and S. S. Rustamov. "Investigation of combined use of MFCC and LPC features in speech recognition systems." *World Academy of Science, Engineering and Technology* 19 (2006): 74-80.