2022 IEEE 7th International conference for Convergence in Technology (I2CT)
Pune, India. Apr 07-09, 2022

Speech Recognition using Weighted Finite-State Transducers

Dipshikha Biswas
Department of Computer Science and Engineering
Amrita School of Engineering, Bengaluru
Amrita Vishwa Vidyapeetham, India
dipshikha.biswas461@gmail.com,

Suneel Nadipalli
Department of Computer Science and Engineering
Amrita School of Engineering, Bengaluru
Amrita Vishwa Vidyapeetham, India
nsuneel89@gmail.com.

Sneha B.
Department of Computer Science and Engineering
Amrita School of Engineering, Bengaluru
Amrita Vishwa Vidyapeetham, India
snehabalaji74@gmail.com,

Supriya M.
Department of Computer Science and Engineering
Amrita School of Engineering, Bengaluru
Amrita Vishwa Vidyapeetham, India
m_supriya@blr.amrita.edu

2022 IEEE 7th International conference for Convergence in Technology (I2CT) | 978-1-6654-2168-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/I2CT54291.2022.9824703

***Abstract*—Speech recognition has always been a prominent field of research in NLP, due to its numerous applications such as speech-to-text conversion, voice assistants, enabling smart homes, etc. Computer algorithms are used by speech recognition systems to process, interpret, and transform spoken words into textual content. Modern statistically-based speech recognition systems use a variety of algorithms such as Dynamic Time Warping(DTW), Neural Networks, and end-to-end automatic system. In this paper, we will perform speech recognition using a type of Finite Automata called Weighted Finite-State Transducer.**

A Finite Transducer is a machine that has no final state. It only takes in some input and produces the appropriate output. Finite State Transducers can contain weights, in which case they are called Weighted Finite-State Transducers (WFST), where each transition is labeled with a weight along with their input and output labels. Weights can represent penalties, probabilities, durations, or any other type of value that gets added along the paths for the computation of the overall weight of mapping an input sequence to an output sequence. This property of weighted transducers allows it to be an essential choice for representation of the probabilistic finite-state models that are widely used in speech processing. Thus, weighted finite-state transducers elucidate a common framework for the formulation and use of models in speech recognition, with shared algorithms that provide significant algorithmic and software engineering benefits.

Keywords—Weighted Transducers, NLP, automata, WFST, speech recognition

I. INTRODUCTION

The area of Speech Recognition is vast and has a wide variety of applications. This is used as the pivotal technology when it comes to voice assistants. It also finds its place among fields like Healthcare and Banking as adding voice assistance could greatly ease the process and make it more customer-friendly.

Much of today's large-vocabulary speech recognition relies on models like tree lexicons, HMMs, or n-gram language models, and thus, can be represented by weighted finite-state transducers. A finite-state transducer is a finite automaton that simply takes in an input and produces an output. The weight here represents probabilities, penalties, durations, or any other value that accumulates along the paths to calculate the comprehensive weight of mapping an input string to an output string. Weighted transducers are therefore, an obvious choice for representing the probabilistic finite-state models common in speech processing.

In this project, we use a Hidden Markov model (HMM)-based pipeline where the probability of transitioning to another state is modeled by the weights in the WFSTs. The training and decoding phases use the Forward and Viterbi Algorithms respectively. The Free Spoken Digit Dataset is used for the process of training. This dataset has recordings of six speakers who utter the English numerals and their audio is stored in .wav files, at 8 kHz. There are a total of 3,000 recordings (50 per digit, per speaker) in this dataset.

II. RELATED WORK

NLP is a vast topic and as a result has been the subject of research for many years now, including automatic speech recognition (ASR). There have been many undertakings to construct an ASR system that could be extend to a multitude of applications – speech transcription, translation, voice assistants, etc. To achieve this, quite a few concepts have been employed, notably HMMs and the topic of this paper, Finite State automata. Hidden-Markov models and the related probabilistic automata for acoustic modelling, as well as numerous probabilistic language models, have been widely employed in voice recognition for a long time [1]. Text-based applications such as Chinese text segmentation [2] and text indexation [3] can also benefit from weighted finite-state automata.

Apart from implementations of the core concept, there have been various publications discussing the theory behind this application and sketching out the idea. Tharun et al. [4] discusses a pipeline for continuous speech recognition with the help of utterance verification. The work proposed in [4] talks about the various algorithms and steps involved in detail. Mohri et al. has also provided the necessary theoretical overview in regard to the application of weighted automata in speech recognition [5], as well as applying optimization techniques on certain automata against the DARPA evaluation dataset to reduce the number of states, transitions and time in order to perform speech recognition

[6]. N Uchat [7] and J Hui [8] detail us through all the necessary theoretical terms required for this project.

There are many other architectures that our paper has been inspired from. These involve Neural Network and Deep Learning architectures, such as those in Madansamy et al. [9], Lalitha et al. [10] and Kumar et al. [11]. The architecture in the work proposed by Shivakumar et al. [12], focuses on improving the accuracy of speech to text pipeline, while that proposed by Poorna et al. [13] and Veni et al. [14] focus on a weighted and multimodal architecture for the same pipeline.

This paper takes the core concept of the above mentioned works as a reference and implements the end to end pipeline for automatic speech recognition using weighted finite state transducers and manages to produce a fruitful result with a relatively small dataset.[15]

III. THEORETICAL OVERVIEW

A. Weighted Finite-State Transducers

Weighted Transducers are used in many applications, such as text, voice, and picture processing. They are automata in which, in addition to the standard input label, each transition has an output label from a possible new alphabet and a weight element.

Transducers can be used to create a mapping between two types of data sources, such as word and phoneme sequences. To model the uncertainty or variability of such information sources, the weights are crucial. For instance, weighted transducers can be used to assign different pronunciations to the same word with varying rankings or probability.

Weighted Finite-State Transducers are usually good at modeling the required HMM and solving state machine problems. It gives HMM phonetic models, pronunciation lexicons, context-dependent phones, and grammars a natural representation in the ASR context.

In this project, we have used three operators to manipulate the implementation of the WFSTs as a part of the Automata Theory:

- **Composition**

  Composition is a transducer operation that allows several levels of representation to be combined. For example, a phone-to-word transducer where the word sequences are limited by the grammar can be created by combining a pronunciation lexicon with a word-level grammar. Composition makes it simple and fast to use several ASR transducer combination strategies, both context-independent, and context-dependent.

- **Determinization**

  Weighted determinization applies to a weighted automaton and produces an equivalent deterministic weighted automaton, which generalizes the classical subset approach for determinizing finite automata. We can make the WFST deterministic to boost search efficiency. Determinization removes numerous pathways, resulting in a single path for each input sequence. It drastically decreases the time and space required to decode an input sequence.

- **Minimization**

  A deterministic weighted automaton is considered to be minimal if no alternative deterministic weighted automaton with fewer states and performing the same function exists. Minimization reduces the number of states in a deterministic WFA to the minimum. Again, it simplifies the model.

B. Hidden Markov Model

The Hidden Markov model is the foundation of several effective acoustic modeling approaches in speech recognition systems. HMM has a lot of mathematical structure, thus it can be used as a theoretical foundation for a lot of things. When used correctly, the HMM model works effectively in practice for a variety of essential applications, which is why WFSTs are used to model HMM in this project.

The HMM may be used to model the transition between phones and the accompanying observable.

HMMs answer three questions:

- *Evaluation* — How likely is it that something observable will happen?
- *Decoding* — What is the reason for the observation that happened?
- *Learning* — What I can learn from the observation data I have?

To answer each of these questions, HMM uses different algorithms. The evaluation, decoding, and the learning problems can be solved using the Forward Algorithm, the Viterbi Algorithm and the Baum Welch Algorithm respectively.

In this project, we deal with two main HMM algorithms:

- **Forward Algorithm**

  Essentially, the probability of a state at a particular time is calculated using a forward algorithm based on the history of evidence. After an HMM is learned, the forward algorithm is used to compute the likelihood of the observations (eq. 1)

  $$p(X) = \sum_S \quad p(X,S) = \sum_S \quad p(S)p(S) \quad (1)$$

- **Viterbi Algorithm**

  Viterbi algorithm is a dynamic programming algorithm for obtaining the maximum posterior probability estimate of the most likely sequence of hidden states called the Viterbi path (eq. 2)

  $$v_t(j) = max_{i=1}^{N} v_{t-1}(i) a_{ij} b_j(x_t) \quad (2)$$

C. Gaussian Mixture Model

Gaussian Mixture Model (GMM) demonstrates the observed probability distribution of the feature vector for a given phone. A probabilistic model in which all data points are considered to be generated by a mixture of a finite number of Gaussian distributions with unknown parameters is known as a Gaussian mixture model. Mixture models may be viewed as a generalization of k-means clustering that incorporates information on the data's covariance structure as well as the latent Gaussian distributions' centers.

Implementing GMM allows for multimodal distributions, which means that any existing feature can

have a few different values. This allows for a variety of speech variations to be used.

So, given a phone, GMM can be used to learn the feature vector of the observable. The likelihood of the taken speech segment to a phone can be computed using the resultant probability distribution.

This probability calculated in the previous step is also the emission probability of a given internal state in HMM.

D. MFCC Vectors

MFCC stands for Mel-frequency cepstral coefficients. It is a popular feature extraction method that is well suited for audio samples. It has the following objectives:

- Removing vocal fold excitation (F0) — the pitch information.
- Making the extracted features independent.
- Adjusting to the perception of loudness and frequency of sound in humans.
- Capturing the dynamics of phones (the context).

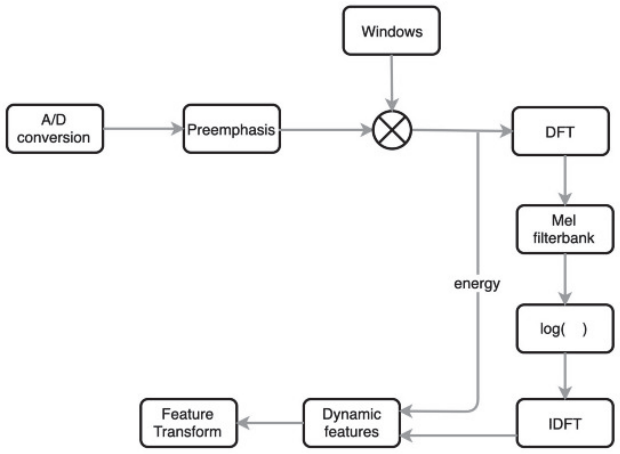


Fig 1. Conversion of signals into MFCC features
Source: Adapted from [8]

The steps involved in MFCC's feature extraction is explained below (Fig. 1):

- **A/D Conversion** involves converting the given analog signal into a digital signal for further processing
- **Pre-emphasis** aids in increasing or boosting the energy of a sound wave in higher frequencies
- **Windowing** refers to breaking the audio signal into parts for better detection and analysis
- **DFT** is used to convert a given signal in the time domain to the frequency domain
- **Mel Filterbank and Log** uses the Mel scale to map the audio of a given frequency to one that can be perceived by humans and mimics the human hearing system
- **IDFT** is performed to gain more information about the phones in the speech

IV. General Workflow

Our model follows the steps listed below (Fig. 2) to perform training (on the free-spoken digits Dataset) and testing (on an unseen audio clip)

- Input dataset is passed to the model
- Feature Extraction using MFCCs
- Creating an acoustic model by quantizing the features using the GMM model
- Word to phoneme mapping is performed using the acoustic model
- The HMM model is created along with a transition probability matrix (C, H)
- Applying the Forward algorithm for training and Viterbi algorithm for decoding

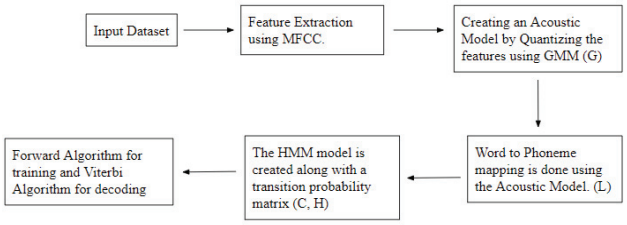


Fig 2. Diagram for general workflow

A. Step I: Passing dataset to the model

The input dataset consisting of audio files of spoken digits (in the .wav format) is passed to the model

B. Step II: Feature Extraction

From each audio sample given, twenty features are extracted. This number is used as it is the most optimal one and prevents complications in the model. These features are extracted using the MFCC method.

C. Step III: Creating the acoustic model

The features obtained from Step II are still in the continuous form and cannot be modeled to represent the states of an HMM. Each of these feature vectors must be mapped to a particular value (or phoneme) and that will get converted to an HMM.

The main goal of speech recognition is to build a statistical model that can infer text sequences W from a set of feature vectors X. To recognize a given word, we should extract phonemes from a voice sample. A GMM may be used to model the distribution of characteristics on a phone.

Thus, feature vectors have been modeled with a much denser representation and this is called quantization. We find the text sequence with the highest likelihood by integrating the acoustic and linguistic models.

D. Step IV & Step V: Phoneme mapping & creating HMMs

From the acoustic model we create a GMM-HMM representation. The GMM is modeled into 10 states, and we want the phonemes of only the ten digits.

Phones are not homogeneous. The amplitudes of frequencies change from the start to the end. Thus, we further subdivide the phone into three states: the beginning, the middle, and the ending part of a phone. This forms the states of our HMM.

For each digit in our dataset, we create one such model. Our final model comprises ten GMM-HMM models.

E. Step VI: Training and decoding

We wish to calculate P(X|W), given an observation sequence and an HMM model. This relates to the Evaluation Problem and to solve it, we use the Forward Algorithm. This algorithm is used in the training phase and it helps in finalizing our Transition and Emission Probability Matrix.

The next problem we encounter is that given an observation sequence and an HMM model we need to calculate the most probable state sequence. This represents the Decoding Problem, which can be solved using the Viterbi Algorithm. This algorithm is used in the prediction stage, where a sound wave is fed to it and the appropriate text is transcribed.

F. Automata Implementation

Step III of the Workflow described about creating an acoustic model for the given data, which involved mapping a given feature vector to the distribution of phonemes, essentially creating a vector-phoneme mapping through quantization.

The next step involves creating the automata that will help label unknown data, and it will be modeled as a GMM-HMM. The combined model can be thought of as two separate GMM and HMM models working together.

The GMM model is comprised of 10 states – each state corresponding to each of the digits present in our database. Phonemes have different frequencies from beginning to end; leading to a further subdivision of each phoneme into three states – modeled by the HMM. This forms the composition of the final GMM-HMM model and each digit in the dataset has its own GMM-HMM model.

The HMM model is trained using the Forward Algorithm, which fine tunes the elements of the transition and emission probability matrices so that they have optimal values best fit to the given data. This algorithm relates to the evaluation problem solved by HMMs and helps with the vector-phoneme mapping, or in other words, answers the question: how probable is it that something happened given the data?

HMMs usually answer 3 problems, one of which was discussed above. The other problem tackled in this project is the decoding problem and helps in the prediction stage, where an unseen voice recording is transcribed with the most appropriate digit. The Viterbi Algorithm is employed to solve this problem and answer the question: what the most probable state sequence is, given an observation sequence and an HMM model.

This is how the automata for the project is constructed and trained to best fit the data. When it comes to predicting the text/label for unknown data, the MFCC features are extracted from the given recording and converted into a feature vector with quantization. This is then run against all 10 models and the label whose model outputs the highest probability is chosen as the transcribed text. Fig. 3 depicts the automata example. The system architecture for the problem described in this section is presented in Fig. 4.

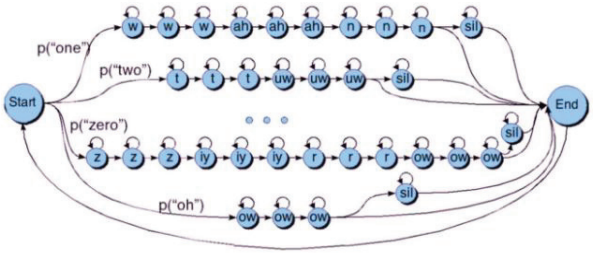


Fig 3. Example automata
Source: Adapted from [8]

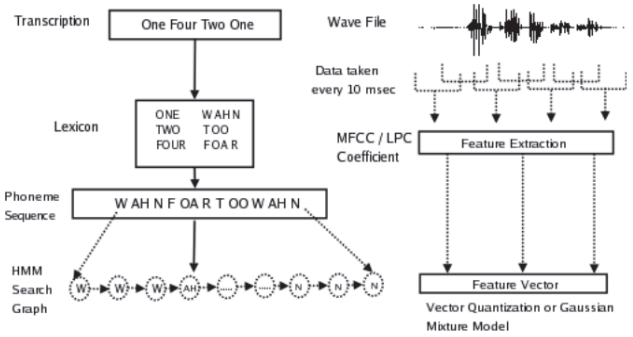


Fig 4. Final System Architecture
Source: Adapted from [7]

V. Results and Discussion

This speech to text system was implemented using Python 3.9 and was tested on a Google Colab Virtual Environment. The audio files from the free spoken digit dataset were converted to feature vectors with the help of MFCC feature extraction. More specifically, 20 MFCC features were extracted which yielded:

- A recognition rate of 81.2%
- An F1 Score of 80.7%
- Precision of 82.5%
- Recall of 80%

These rates were modeled using an automata with 5 states. These parameters were tested among other values and were empirically found to give out better metrics.

Here is the classification report for each class in our model. Each class indicates each digit which ranges from 0 to 9.

TABLE I. Classification Report

Class	F1 Score (%)	Precision (%)	Recall (%)
0	89.5	93.4	86
1	75.2	81.3	70
2	87.6	95.1	84.7
3	71.7	73.3	70.2
4	98.1	99	96.2
5	84.7	79.4	88.5
6	51.7	42.8	65.2
7	78.2	87.2	70.8
8	80	75.8	84.6
9	86.1	94.8	78.7

A. Streamlit GUI

The culmination of the project code results in a GUI that the user can interact with to:

- Record an audio clip
- Transcribe the clip to text

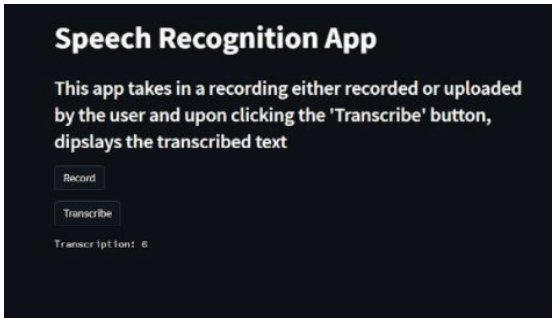
Record an audio clip

Once the user has clicked the “Record” button, the user can use their device’s microphone to record an audio clip of them saying out a number. The button will automatically record for 1 second, after which the clip will be stored until further action is taken.

Transcribe audio

After the user has recorded their audio clip, they can click the “Transcribe” button which will then run the clip through the trained model and obtain the label (digit) with the highest probability. This digit is then displayed as text under the buttons. A snapshot of the implementation is presented in Fig. 5.

The user can perform the above order of operations as many times as needed before closing the window when required.



Speech Recognition App

This app takes in a recording either recorded or uploaded by the user and upon clicking the 'Transcribe' button, dipslays the transcribed text

```
Transcription: 6
```

Fig 5. Final GUI Window

VI. CONCLUSION AND FUTURE SCOPE

An implementation of a speech recognition system using Weighted Finite-State Transducers has been carried out in this project. This speech recognition system detects single words, i.e., numbers uttered by the users. The input dataset considered for the validation of the model is .wav audio files. Feature extraction, creation of acoustic model using GMM, procurement of transition probability of the phonemes, training, and decoding using forward and Viterbi algorithm respectively, and finally and implementation of GUI has been done in this project. The results obtained were fruitful with a recognition rate of 81.2%.

However, there is still scope for improvement, that could further be the usefulness of the project and increase its applications. One such improvement would be to extend the capabilities of this speech recognizer to continuous speech, making it more fit for the real world and increasing its utility in the process. Another extension involves speech recognition for languages other than English.

With the right improvements, this project could prove to be another application of HMMs, amongst the already existing ones. It could also be helpful in exploring the uses and capabilities of transducers among new learners and enthusiasts of the field of automata, giving them valuable insight and clarity about the inner workings of transducers.

REFERENCES

[1] Lalit R. Bahl, Fred Jelinek, and Robert Mercer, ‘A maximum likelihood approach to continuous speech recognition’, IEEE Trans. PAMI, 5(2), 179–190, (March 1983).

[2] Richard Sproat, Chilin Shih, Wiliam Gale, and Nancy Chang, ‘A stochastic finite-state word-segmentation algorithm for Chinese’, in 32nd Annual Meeting of the Association for Computational Linguistics, pp. 66–73, San Francisco, California, (1994). New Mexico State University, Las Cruces, New Mexico, Morgan Kaufmann.

[3] Mehryar Mohri, ‘On some applications of finite-state automata theory to natural language processing: Representation of morphological dictionaries, compaction, and indexation’, Technical Report IGM 94-22, Institut Gaspard Monge, Noisy-le-Grand, (1994).

[4] M.THARUN PRASATH (2014) Continuous Speech Recognition Based on Deterministic Finite Automata Machine using Utterance and Pitch Verification.

[5] Mehryar Mohri, Fernando Pereira, Michael Riley (2005) Weighted Automata in Text and Speech Processing .

[6] Mehryar Mohri, Fernando Pereira, Michael Riley (2001) WEIGHTED FINITE-STATE TRANSDUCERS IN SPEECH RECOGNITION

[7] N. Uchat, *Cse.iitb.ac.in*, 2022. [Online]. Available: https://www.cse.iitb.ac.in/~nirav06/i/HMM_Report.pdf.

[8] J. Hui, "Speech Recognition Series", *Medium*, 2022. [Online]. Available: https://jonathan-hui.medium.com/speech-recognition-series-71fd6784551a.

[9] Madasamy, A. K., & Padannayil, S. K. (2021). Transfer learning based code-mixed part-of-speech tagging using character level representations for indian languages. Journal of Ambient Intelligence and Humanized Computing, doi:10.1007/s12652-021-03573-3

[10] Lalitha, S., Tripathi, S., & Gupta, D. (2019). Enhanced speech emotion detection using deep neural networks. International Journal of Speech Technology, 22(3), 497-510. doi:10.1007/s10772-018-09572-8

[11] Kumar, S., Kumar, M. A., & Soman, K. P. (2019). Deep learning based part-of-speech tagging for malayalam twitter data (special issue: Deep learning techniques for natural language processing). Journal of Intelligent Systems, 28(3), 423-435. doi:10.1515/jisys-2017-0520

[12] Shivakumar, K. M., Jain, V. V., & Priya, P. K. (2018). A study on impact of language model in improving the accuracy of speech to text conversion system. Paper presented at the Proceedings of the 2017 IEEE International Conference on Communication and Signal Processing, ICCSP 2017, , 2018-January 1148-1151. doi:10.1109/ICCSP.2017.8286557

[13] Poorna, S. S., Anuraj, K., & Nair, G. J. (2018). A weight based approach for emotion recognition from speech: An analysis using south indian languages doi:10.1007/978-981-13-1936-5_2

[14] Veni, S., & Thushara, S. (2017). Multimodal approach to emotion recognition for enhancing human machine interaction - a survey. International Journal on Advanced Science, Engineering and Information Technology, 7(4), 1428-1433. doi:10.18517/ijaseit.7.4.1662

[15] Z. Jackson, "GitHub - Jakobovski/free-spoken-digit-dataset: A free audio dataset of spoken digits. Think MNIST for audio.", GitHub, 2022. [Online].