Keyword Spotting with Deep Neural Network and Hidden Markov Model

Jiayan Li and Emin Ozyoruk

Abstract

This report addresses the problem of keyword spotting (KWS) in audio files, specifically targeting the detection of the keyword "never." The goal is to create a system that operates efficiently on a low-power processor, maintains a small memory footprint, and achieves high precision. We implemented a system that combines a Deep Neural Network (DNN) for feature extraction and a Hidden Markov Model (HMM) for keyword detection. This report details our approach, including data collection, feature engineering, model training, and experimental results. Our system demonstrates promising results with small training data size and low false positive rate.

1 Introduction

Keyword Spotting (KWS) is a critical area in speech recognition technology focused on identifying specific words or phrases within continuous streams of audio. This technology is fundamental to voice-activated systems such as virtual assistants, where it enables devices to respond to predefined commands such as "Hey Siri" or "OK Google" without requiring manual activation.

The primary goal of KWS systems is to enable hands-free interaction with devices, making them essential for applications where users need to operate devices without physical contact. For instance, while driving, cooking, or in situations where quick access is necessary, such as during emergencies. These systems must achieve a high level of accuracy and reliability while maintaining low computational and power overheads, especially for deployment on battery-powered devices.

KWS is crucial for voice-activated systems like virtual assistants. The primary challenge is to detect specific keywords accurately while minimizing power consumption and computational costs. This involves designing a system that can run on a low-power processor, even when the main processor is asleep, ensuring a small memory footprint and low computational cost without compromising precision.

The development of KWS can be traced back to the early days of speech recognition research. In the 1980s and 1990s, initial approaches to KWS relied on template matching and dynamic time warping (DTW) techniques. These methods were effective for small vocabulary tasks but struggled with larger and more diverse sets of keywords. The introduction of hidden Markov models (HMMs) in the 1990s marked a significant advancement, allowing for better modeling of temporal variations in speech [1,2]. The HMMs were coupled with acoustic models such as Gaussian Mixture Models (GMM) [3]. The true breakthrough in KWS came with the advent of deep learning in the 2010s [4]. Neural network-based approaches, particularly those using convolutional neural networks and recurrent neural networks, have dramatically improved the accuracy and robustness of KWS systems.

While the earlier works focused on using KWS systems in telecommunications and security systems, recent approaches aim to improve voice activated assistants such as Amazon Alexa, Google's assistant, and Apple's Siri. In [5], the authors combines DNN and HMM to detect "Hey Siri" in low resource hardware such as iPhone and Apple watch. The study in [6] builds on [5] by joinly optimizing DNN and HMM instead of training the models separately. In [7], the authors use DNN and a new posterior handling method to detect "okay google".

In our report, we are inspired by the studies in [5] and [6]. Our combined DNN and HMM model closely resembles the one in [5]. Due to the dataset available to us, we decided to detect the keyword "never". In addition to low computational complexity, our model was trained on an extremely small dataset and demonstrated good performance, especially in terms of recall.

2 Data and Method

2.1 Data

We utilized the TIMIT Acoustic-Phonetic Continuous Speech Corpus, a comprehensive dataset comprising recordings from 630 speakers, each articulating ten distinct sentences. For our keyword detection task, we selected the keyword "never," which appeared in 57 out of the 6300 audio files. This posed a significant challenge due to the limited occurrence of the keyword within the dataset. Specifically, we allocated 28 audio files to the training set and 29 to the test set, highlighting the constraints of working with a small dataset.

For feature extraction, we employed the `librosa` library, which facilitated the extraction of Mel-frequency cepstral coefficients (MFCCs). The audio data was sampled at a rate of 16000 Hz, with each frame having a duration of 25 milliseconds and an overlap (hop length) of 5 milliseconds. This configuration resulted in the generation of a 20-dimensional MFCC vector for each frame.

The keyword "never" included four distinct phonemes: /n/, /eh/, /v/, and /axr/. Each phoneme was further segmented into three states: beginning, middle, and end. Additionally, we included one state for silence and one state for background noise, culminating in a total of 14 states. For each frame (as illustrated by the green box in Figure 1), the vector representing the percentage of the frame occupied by each phoneme state was calculated. The elements of this 14-dimensional vector summed to one, providing a probabilistic representation of phoneme presence within each frame. This 14-dimensional vector was subsequently used as the ground truth for training our deep neural network (DNN).

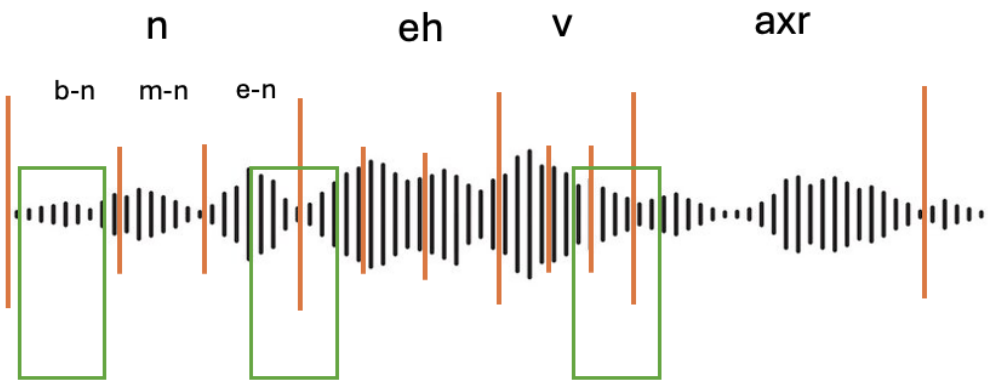

Figure 1: Feature engineering. Green boxes represent each frame, and orange vertical lines represent the beginning and end of each phoneme state. We divide the total duration of each phoneme into three parts: Beginning, middle, and end. Frames are labeled based on the percentage of intersection with each phoneme state window. The labels of frames are vectors representing the probability distributions.

2.2 Methods

Our KWS system used DNN as encoder and HMM as decoder. The DNN processes MFCC vectors and outputs probabilities for different phoneme states. The HMM uses these probabilities to detect the keyword using the Viterbi algorithm. A single threshold was used for the path probabilities of all layers of HMM in determining whether to trigger the system.

We implemented two types of DNNs as encoder, both using softmax as activation layer and cross entropy loss.

- Feed-forward Neural Network (FF-NN): Consisting of 4 linear layers.
- Long Short-Term Memory Network (LSTM): Consisting of 2 LSTM layers followed by 3 linear layers with ReLU activation.

Transition probabilities between phoneme states (e.g., from "beginning-n" to "middle-n") were calculated solely from the training data, ensuring that the test data remained untouched. The HMM leverages the DNN outputs as emission probabilities. The Viterbi algorithm was used to determine the highest probability path, and the final detection decision was made based on a threshold defined based on training sets.

3 Experimental Results and Discussion

We first evaluated the DNN encoder alone using accuracy of the highest probable phoneme state. The LSTM achieved an accuracy of 61.78% on the test set. The FF-NN achieved an accuracy of 74.28% on the test set.

For the experiment on the entire KWS system, we used three sets of emission probabilities—true emission labels (Figure 2), labels from the LSTM model (Figure 3), and labels from the FF-NN model (Figure 4)—while keeping the transition probability and HMM structure fixed. The threshold for triggering the system was determined after examining the boxplots on the left side in the plots and evaluated on the test set. The comparison of results are shown in Table 1.

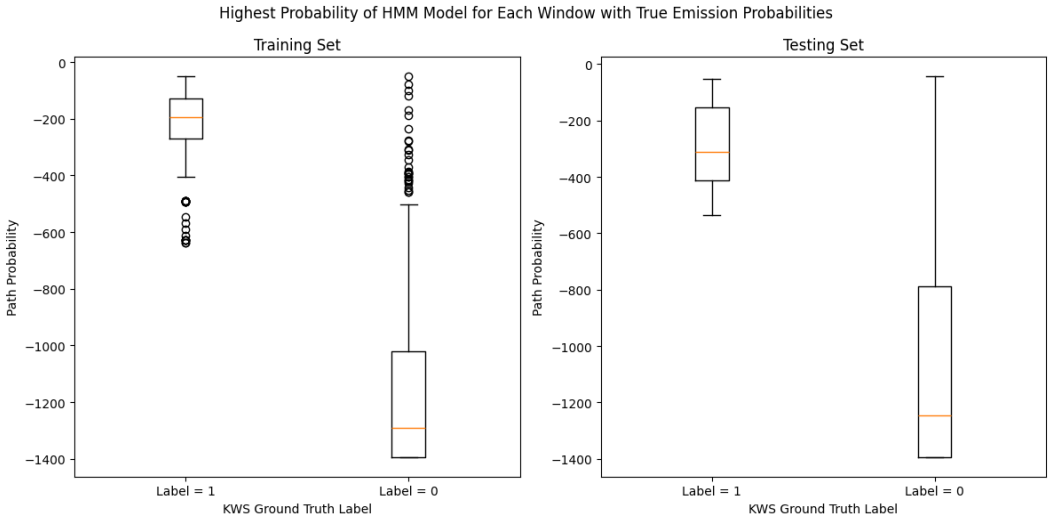

Figure 2: HMM Path Probability with True Emission Probability.

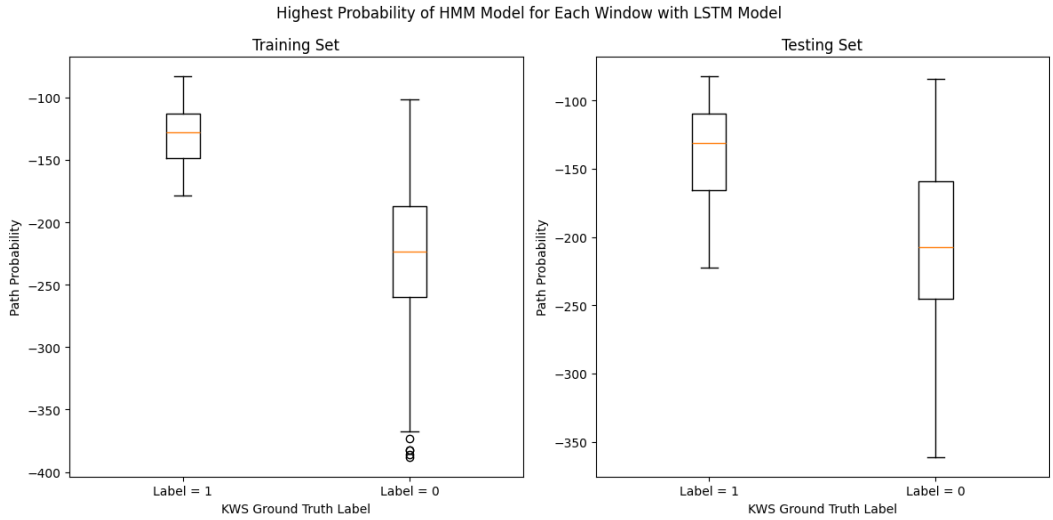

Figure 3: HMM Path Probability with LSTM Model.

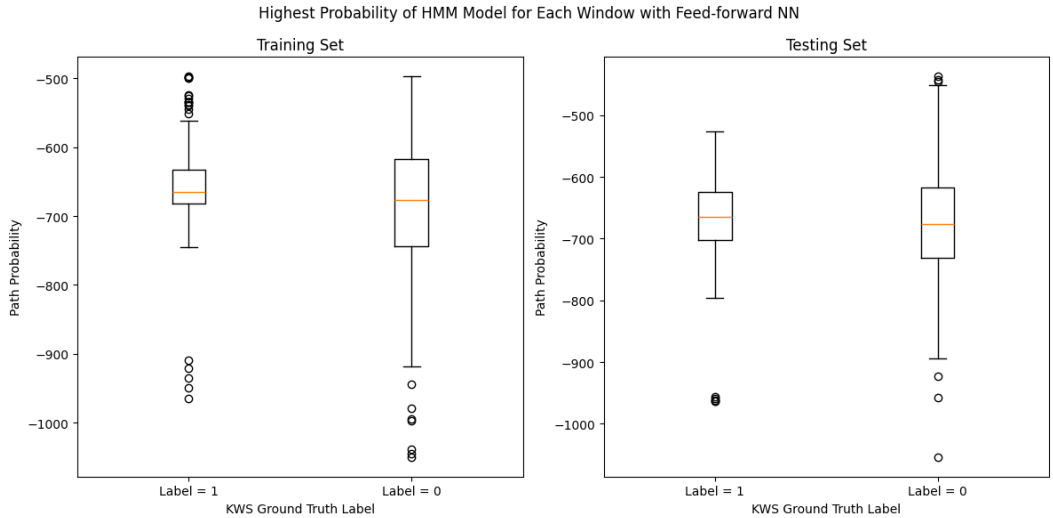

Figure 4: HMM Path Probability with Feed-forward Neural Network.

Table 1: Performance Metrics with Emission Probabilities from Different Sources

	Train Set	Test Set
DNN True Labels		
Recall	0.94	1.0
F1	0.86	0.76
Precision	0.79	0.61
LSTM Model		
Recall	0.98	0.92
F1	0.73	0.57
Precision	0.58	0.42
Feed-forward Model		
Recall	0.86	0.78
F1	0.39	0.37
Precision	0.25	0.25

Note that while the accuracy of the FF-NN was higher than that of the LSTM, the emission probability performed worse in the overall system. This discrepancy arises because the accuracy of the DNN encoder serves merely as a sanity check for how closely the encoder's results match the ground truth. It does not, however, reflect how accurately the emission probability aligns with the true vector.

The experiment results shown in Table 1 reveal several directions for improvement. First, improving the performance of the DNN and the accuracy of the emission probability is crucial. In the results, the HMM with true emission probabilities clearly outperforms the KWS with emission probabilities from the LSTM and FF-NN, proving that enhancing the accuracy of the DNN encoder would significantly improve the system's performance (0.76 F1 score on the test set). To help the DNN increase accuracy, collecting more audio samples containing the target phonemes would address data imbalance issues and improve model training.

Second, improving the transition probability is another avenue. There is a discrepancy between the train set and test set with true emission probabilities, with the only difference being that the transition probabilities are calculated with the training set alone. Again, with more data, the validity of transition probabilities could greatly improve, thereby enhancing the performance of the HMM.

Furthermore, integrating end-to-end training for the DNN and HMM components could streamline the process and potentially increase accuracy. Finally, optimizing detection thresholds, perhaps by setting multiple thresholds to balance false negative and positive rates, would further refine the system's performance. One threshold could be set to ensure high recall and low false negatives, while another could ensure high precision and low false positives. If between the two thresholds, another more sophisticated model could be triggered.

These directions hold promise for achieving even higher accuracy and efficiency in keyword spotting applications.

4 Conclusion

Our keyword spotting system effectively detects the keyword "never" with extremely small training size, moderate accuracy, and high efficiency. The combination of DNN for feature extraction and HMM for keyword detection proved to be successful, demonstrating the potential for deployment in low-power devices and with limited training data.

References

[1] Gales, M., & Young, S. (2008). The application of hidden Markov models in speech recognition. Foundations and Trends® in Signal Processing, 1(3), 195-304.

[2] Rose, R. C., & Paul, D. B. (1990, April). A hidden Markov model based keyword recognition system. In International conference on acoustics, speech, and signal processing (pp. 129-132). IEEE.

[3] Wilpon, J. G., Rabiner, L. R., Lee, C. H., & Goldman, E. R. (1990). Automatic recognition of keywords in unconstrained speech using hidden Markov models. IEEE Transactions on Acoustics, Speech, and Signal Processing, 38(11), 1870-1878.

[4] Graves, A., Mohamed, A. R., & Hinton, G. (2013, May). Speech recognition with deep recurrent neural networks. In 2013 IEEE international conference on acoustics, speech and signal processing (pp. 6645-6649). Ieee.

[5] Chen, G., Parada, C., & Heigold, G. (2014, May). Small-footprint keyword spotting using deep neural networks. In 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP) (pp. 4087-4091). IEEE.

[6] Sigtia, S., Haynes, R., Richards, H., Marchi, E., & Bridle, J. (2018, October). Efficient Voice Trigger Detection for Low Resource Hardware. In Interspeech (pp. 2092-2096).

[7] Shrivastava, A., Kundu, A., Dhir, C., Naik, D., & Tuzel, O. (2021, June). Optimize what matters: Training dnn-hmm keyword spotting model using end metric. In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 4000-4004). IEEE.

[8] Chen, G., Parada, C., & Heigold, G. (2014, May). Small-footprint keyword spotting using deep neural networks. In 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP) (pp. 4087-4091). IEEE.