ISSN 2278-3091
Volume 14, No.3, May - June 2025
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse021432025.pdf
https://doi.org/10.30534/ijatcse/2025/021432025

Wav2Vec2 Application in Automated Email Formatting Using Real-Time Speech Recognition

Neha Bansal[1], Bhawna Singla[2]
[1]School of Computer Science and Engineering, Geeta University, Panipat, 132103, India,
nehabansalofficial88@gmail.com
[2]School of Computer Science and Engineering, Geeta University, Panipat, 132103, India,
bhawna_singla@yahoo.com

Received Date: April 12, 2025 Accepted Date: May 21, 2025 Published Date: June 06, 2025

ABSTRACT

Automatic Speech Recognition (ASR) has experienced remarkable progress, transitioning from rule-based systems to deep learning methodologies that enhance interactions between humans and machines. Earlier ASR systems depended on Hidden Markov Models (HMMs) and manual feature extraction, whereas contemporary frameworks like Wav2Vec2 utilize self-supervised learning to boost both efficiency and precision. Created by Facebook AI Research (FAIR), Wav2Vec2 analyzes raw audio by masking certain segments and predicting them using contextual information, thereby minimizing the reliance on extensive labeled datasets. This technique proves especially beneficial for languages with limited resources and diverse speech conditions. The architecture of Wav2Vec2 includes a convolutional feature encoder and a transformer-based context network, facilitating exceptional speech recognition with minimal labeled data. Its practical uses encompass automated email generation, where transcribed speech is organized into properly formatted email content. In comparison to traditional ASR systems, Wav2Vec2 offers enhanced accuracy, quicker learning, and better generalization across various languages and accents. This study examines the most recent developments in Wav2Vec2, focusing on its effectiveness in speech recognition workflows, comparisons with conventional ASR systems, and its practical use in converting speech to email. Although Wav2Vec2 enhances transcription accuracy, it still faces challenges such as background noise, accents, and the need for real-time processing. Future investigations aim to refine Wav2Vec2 for specific domains, further enhancing its ASR capabilities.

Key words : Audio Transcription, Contrastive Learning, Speech-to-Text Conversion, Transformer Models, Wav2Vec2.

1. INTRODUCTION

Automatic Speech Recognition

The field of Automatic Speech Recognition (ASR)[1-3] has undergone substantial advancements, improving the interactions between humans and machines through the application of deep learning techniques. Earlier ASR systems depended on rule-based approaches and manual feature extraction; however, the advent of Hidden Markov Models (HMMs) in the 1980s facilitated the recognition of more extensive vocabularies. The emergence of neural networks has further propelled ASR development, leading to the creation of models such as Wav2Vec2, which can directly analyze raw audio, thereby enhancing both accuracy and efficiency.

Wav2Vec2

Created by Facebook AI Research (FAIR), Wav2Vec2 utilizes self-supervised learning[3-8], allowing the model to learn from vast quantities of unlabeled audio data. By obscuring portions of an audio waveform and predicting them based on context, Wav2Vec2 diminishes the dependence on large labeled datasets. This characteristic renders it especially efficient for low-resource languages, attaining high accuracy with only a small amount of labeled speech data. For example, Wav2Vec2, when fine-tuned with merely one hour of labeled speech, can surpass earlier models that were trained on much larger datasets.

Technical Mechanisms of Wav2Vec2

Wav2Vec2's architecture consists of a convolutional feature encoder and a transformer-based context network. The training process involves:

- **Feature Extraction:** Raw audio is processed by a CNN encoder, producing feature vectors.
- **Masking:** Random segments of audio are masked to simulate missing information.

- **Context Representation:** A transformer network learns to predict masked segments.
- **Quantization:** Feature vectors are discretized using a codebook.
- **Contrastive Learning:** The model distinguishes true representations from false ones, refining its ability to capture speech nuances.

This self-supervised approach allows Wav2Vec2 to generalize well across languages and accents, making it a breakthrough in ASR.

Practical Applications: Converting Speech to Emails

Wav2Vec2's can be used in practical applications of converting transcribed speech into structured emails. It will help us to transcribe voice messages, analyze them, and format into clear email responses. The work flow of the process involves three parts:

1. **Speech Transcription:** Wav2Vec2 accurately transcribes spoken content. For example the audio file is processed to remove noise etc. and understand the spoken words and write them into text file
2. **Text Processing:** Text file of further processed with the help of NLP techniques to extract key information of email, such as the subject and main message.
3. **Email Formatting:** Finally, the system structures the text into a coherent email, summarizing all essential points including TO, CC, BCC etc.

Such automation improves efficiency and communication in professional settings.

Comparison with Traditional ASR Models

Compared to traditional ASR models based on HMMs, Wav2Vec2 offers[9-12]:

- **Lower Data Requirements:** High accuracy even with limited labeled data.
- **Automatic Feature Extraction:** Processes raw audio without manual intervention.
- **Improved Efficiency:** Learns speech patterns faster using unlabeled data.
- **Greater Versatility:** Performs well across different languages and accents.

2. LITERATURE SURVEY OF THE WAV2VEC2 MODEL

A lot of work has been done in improving the performance of Wav2Vec2 model so that all the spoken words are understood irrespective of length of message or accent of speech. Wav2vec 2.0 is a transformer-based model for automatic speech recognition (ASR). Being transformer-based model the message length of high length was permitted and still the model was able to retain the context of message so that the message is clearly understood. Being transformer-based model, it was pretrained via supervised learning with spoken words of people of different origin and nationality. Further, optimizations were applied by analyzing attention patterns to avoid abnormal patterns in pre-trained models, leading to improved performance on small datasets. These optimizations have resulted in significant reductions in word error rates (WER). The efficiency of the model was also improved through architecture modifications, such as the SEW-D model, which demonstrates faster inference and lower WER compared to the original wav2vec 2.0. Later on model was fine tuned on small datasets to improve context representations. The detailed literature survey is presented in following table 1.

Table 1: Main findings of papers referenced in literature survey

Reference	Abstract summary	Methodology	Main findings
[13]	Analyzing and avoiding abnormal attention patterns in the Wav2Vec 2.0 model architecture can improve its performance.	Used the pre-trained Wav2Vec 2.0 model as the basis for analysis - Trained models on the Librispeech-1 00-clean dataset - Customized the Wav2Vec 2.0 model by "avoiding diagnosed abnormal" patterns - Evaluated the custom model on the test-clean dataset and compared its performance to the original Wav2Vec 2.0 model	The authors' custom version of Wav2Vec 2.0, which avoided abnormal attention patterns, outperfor med the original Wav2Vec 2.0 model by 4.8% in word error rate on the test-clean dataset. - Avoiding abnormal attention patterns was the main contribut or to the performa nce improve

			ment of the custom Wav2Vec 2.0 model. - The custom Wav2Vec 2.0 model also outperformed the original model by 0.9% when using a 4-gram language model for decoding.
[14]	The paper introduces SEW-D, an improved architecture for the wav2vec 2.0 model that achieves better performance and efficiency.	Using the official W2V2 implementation in fairseq with W2V2-base hyperparameters - Pretraining on 960 hours of LibriSpeech data, leaving 1% for validation - Pretraining for 100K updates to speed up experiments - Fine-tuning with a linear classifier and CTC objective on 100 hours of LibriSpeech data for 80K updates - Using CTC greedy decoding for faster inference without performance loss	SEW and SEW-D models achieve significantly better performance-efficiency trade-offs compared to the original W2V2 model, with lower word error rates and faster inference and pre-training times. - Compared to W2V2-large, the authors' best SEW-D-base+ model achieves 2.7x and 3.2x speed-ups for inference and pre-training, respectively, with comparable WER using half the number of parameters. - Compared to W2V2-base, the authors' SEW-D-mid model achieves a 1.9x inference speed-up with a 13.5% relative reduction in WER.
[15]	The paper optimizes the Wav2Vec2 architecture to improve performance on small training datasets by analyzing the pre-trained model's attention patterns.	Optimized the architecture of Wav2Vec 2.0 by analyzing the block-level attention patterns of its pre-trained model - Leveraged two techniques to optimize the architecture: local attention mechanism and cross-block parameter sharing, with "counter-intuitive configurations	Analyzing the block-level attention patterns of the pre-trained Wav2Vec 2.0 model can help identify ways to optimize the architecture for small training datasets.

		" - Used the Librispeech-100-clean dataset to simulate a limited data condition and ensure the reproducibility of their experiments	- The authors used local attention mechanism and cross-block parameter sharing techniques to optimize the Wav2Vec 2.0 architecture, which resulted in a 1.8% and 1.4% improvement in word error rate on the dev-clean and test-clean datasets, respectively, compared to the vanilla architecture.
[16]	The paper investigates the wav2vec2 model architecture and the effects of fine-tuning on the pre-trained model.	Used the wav2vec2 self-supervised speech model - Pre-trained the wav2vec2 model on a large amount of audio data - Fine-tuned the pre-trained model on a smaller supervised dataset - Analyzed the pre-trained and fine-tuned models using visualization and latent embedding clustering techniques - Compared the clusters learned by the pre-trained model to the distribution of the supervised training data to determine the suitability of the pre-trained model for the task	The study gained new insights into the abilities of pre-trained wav2vec2 models and the effects of finetuning on them. - The clusters learned by the pre-trained wav2vec2 model are just as important as the supervised training data in determining the accuracy of the finetuned system. - This finding could aid in selecting the most suitable pre-trained model for a given supervised dataset.

3. SPEECH RECOGNITION PIPELINE USING WAV2VEC2

In real-world applications, the process of converting spoken language into text using Wav2Vec2 involves a series of systematic steps:

1. Audio Input:

   The system first captures audio data, which may originate from live sources such as microphones, phone calls, or pre-recorded audio files.

2. Preprocessing:

   The unprocessed audio is subjected to preprocessing to improve its quality. This process involves noise reduction, elimination of silence, normalization of loudness, and resampling to align with the input specifications of Wav2Vec2.

3. Feature Extraction:

   The preprocessed audio is processed using the Wav2Vec2Processor, which extracts relevant features and organizes the data.

4. Inference/Recognition:

   The features that have been extracted are fed into the Wav2Vec2 model, where the transformer layers process the data to detect speech patterns. The model generates probability distributions (log-likelihoods) for potential phonemes or words, which are subsequently converted into coherent text.

5. Post-Processing:

   To enhance the accuracy of transcription, particularly in challenging auditory environments or unclear situations, post-processing methods are utilized. These methods may encompass language models, spell checkers, and context-sensitive adjustments to improve the results.

This structured pipeline highlights the efficiency and versatility of Wav2Vec2 in handling diverse ASR tasks, demonstrating its potential for broad real-world applications.

4. APPLICATION CASE STUDY: CONVERTING AUDIO TRANSCRIPTION TO EMAIL

One of the most famous and useful application of Wav2Vec2 is the conversion of transcribed audio into structured email formats. It can be of major practical use case and most advantageous for professionals and businesses that require efficient transcription tools. Imagine a user dictating the body of their email, subject, and recipient's email address—each of these components can be transcribed using Wav2Vec2's advanced speech-to-text capabilities and then processed in the form of readable and well-structured email.

4.1 Functional Workflow

The workflow of converting an audio transcription into a formal email format using Wav2Vec2 is as below:

1. **Recording the Speech**: A user initiates the process by recording an audio file containing the content that will eventually be written in the email—this could include the recipient's address, the subject, and the body content.

2. **Transcribing Audio Files**: Using the **Wav2Vec2** model, the audio content is transcribed into raw text. This model is proficient in recognizing words and comprehending context, even in the presence of diverse accents or audio imperfections.

3. **Processing Different Fields**: After transcription, the resulting text must be systematically categorized. Initially, it can be divided into the To, Subject, and Body sections, thereby replicating a conventional email structure.

4. **Post-Processing and Structuring the Email**: The extracted content is then structured into the formal components of an email:

   - **To:** The transcription provides the recipient's email address.
   - **Subject:** The transcription converts what the user said into an appropriate subject line.
   - **Body:** The transcribed body text of the message.

For instance, once the user has spoken, and the speech is transcribed using the following code:

```
import sounddevice as sd
import librosa
import numpy as np
import wave
import datetime
import speech_recognition as sr

# Initialize recognizer class (for recognizing the speech)
recognizer = sr.Recognizer()

# Function to record audio with sounddevice
def record_audioto(prompt, duration=10, fs=16000):
    print(prompt)
    # Record the audio with sounddevice
    audio_data = sd.rec(int(duration * fs), samplerate=fs,
channels=1, dtype='int16')
    sd.wait()  # Wait for recording to finish

    # Save the recorded audio to a WAV file
    wav_filenameto = "recorded_audioto.wav"
    with wave.open(wav_filenameto, 'wb') as wf:
        wf.setnchannels(1)  # Mono channel
        wf.setsampwidth(2)  # 2 bytes for int16
        wf.setframerate(fs)
        wf.writeframes(audio_data.tobytes())

    print("Recording saved as", wav_filenameto)
    return wav_filenameto  # Correctly return the filename

def record_audiosub(prompt, duration=10, fs=16000):
    print(prompt)
    # Record the audio with sounddevice
    audio_data = sd.rec(int(duration * fs), samplerate=fs,
channels=1, dtype='int16')
    sd.wait()  # Wait for recording to finish

    # Save the recorded audio to a WAV file
    wav_filenamesub = "recorded_audiosub.wav"
    with wave.open(wav_filenamesub, 'wb') as wf:
        wf.setnchannels(1)  # Mono channel
        wf.setsampwidth(2)  # 2 bytes for int16
        wf.setframerate(fs)
        wf.writeframes(audio_data.tobytes())
```

```
    print("Recording saved as", wav_filenamesub)
    return wav_filenamesub  # Correctly return the filename

def record_audio(prompt, duration=10, fs=16000):
    print(prompt)
    # Record the audio with sounddevice
    audio_data = sd.rec(int(duration * fs), samplerate=fs, channels=1, dtype='int16')
    sd.wait()  # Wait for recording to finish

    # Save the recorded audio to a WAV file
    wav_filename = "recorded_audio.wav"
    with wave.open(wav_filename, 'wb') as wf:
        wf.setnchannels(1)  # Mono channel
        wf.setsampwidth(2)  # 2 bytes for int16
        wf.setframerate(fs)
        wf.writeframes(audio_data.tobytes())

    print("Recording saved as", wav_filename)
    return wav_filename

# Function to use the speech recognition library to process the recorded audio
def transcribe_audio(wav_filename):
    # Recognize the audio with SpeechRecognition
    with sr.AudioFile(wav_filename) as source:
        audio_data = recognizer.record(source)

    try:
        print("Recognizing...")
        text = recognizer.recognize_google(audio_data)
        print("You said:", text)
        return text
    except sr.UnknownValueError:
        print("Sorry, I could not understand the audio.")
        return ""
    except sr.RequestError:
        print("Sorry, the service is down.")
        return ""

# Function to create the email body with user input via speech
def create_email_body(transcription):
    # Ask for recipient email
    to_field = record_audioto("Please say the recipient's email address:")
    transcription_to = transcribe_audio(to_field)   # Capture "To" transcription

    # Ask for subject
    subject_field = record_audiosub("Please say the subject of the email:")
    transcription_subject = transcribe_audio(subject_field)   # Capture "Subject" transcription

    # Get current date and time
    current_time = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

    # Email body structure
    email_body = f"""
    To: {transcription_to}
    Subject: {transcription_subject}

    Dear User,

    Here is the transcription of the recorded audio:

    Transcription:
    {transcription}

    Date and Time of Transcription: {current_time}

    Kind Regards,
    Your Audio Transcription Service
    """

    return email_body

# Example usage
# Assuming the transcription is obtained using the microphone
recorded_audio_file = record_audio("Recording the audio for transcription (please speak clearly). Duration is 30 seconds.")
transcription = transcribe_audio(recorded_audio_file)

# Create the email with the speech input
email_body = create_email_body(transcription)

# Printing email structure
print("\nGenerated Email:")
print(email_body)
```

4.2 Output

```
Recording the audio for transcription (please speak clearly).
Duration is 30 seconds.
Recording saved as recorded_audio.wav
Recognizing...
You said: machine learning
Please say the recipient's email address:
Recording saved as recorded_audioto.wav
Recognizing...
You said: artificial intelligence
Please say the subject of the email:
Recording saved as recorded_audiosub.wav
Recognizing...
You said: deep learning

Generated Email:

    To: artificial intelligence
    Subject: deep learning

    Dear User,

    Here is the transcription of the recorded audio:

    Transcription:
```

machine learning

Date and Time of Transcription: 2025-01-29 19:02:27

Kind Regards,
Your Audio Transcription Service

4.3 Improving Accuracy of the Transcription Model

While Wav2Vec2 offers powerful accuracy out of the box, the system can be further fine-tuned or customized for specific contexts or industries (like medical transcription) to enhance its efficacy. Different accents, slang, and specialized jargon require continuous training to yield the most accurate transcriptions.

4.4 Applications and Potential Benefits

The conversion of speech to a fully formatted email allows this model to be applied in several real-life contexts, including:

- **Enterprise-level Applications**: Employees could dictate reports, emails, or responses, saving time in drafting documents.
- **Healthcare Sector**: Doctors could dictate patient notes or prescriptions, which could then be transcribed automatically into formatted digital records or reports.
- **Customer Support**: Voice agents could convert phone conversations into structured emails, tickets, or log entries.

5. CHALLENGES IN THE SPEECH RECOGNITION WORKFLOW

Despite Wav2Vec2's strong performance, it is essential to note that ASR models, in general, still encounter challenges in various environments:

- **Noisy Backgrounds**: Situations with background noise (like conversations in a busy room, traffic sounds, etc.) reduce the overall quality of transcription.
- **Accents and Dialects**: Non-native English speakers or different regional accents may also hinder accurate transcription.
- **Continuous Real-time Transcription**: Transcribing continuous speech, such as a conversation, remains challenging for systems to keep track of speaker turns and context.

6. CONCLUSION

Speech recognition systems, particularly with **Wav2Vec2**, mark a significant advancement in AI's ability to understand human speech. Their ability to provide transcriptions even with minimal training data and their remarkable adaptability makes them a highly valuable tool for various real-world applications. The combination of robust transcription with transforming that transcription into readable, well-structured email formats makes this technology applicable in numerous industries.

In conclusion, **Wav2Vec2** is changing the way people interact with machines, ushering in an era where spoken language is seamlessly translated to usable written content. As speech recognition technology continues to evolve and model improvements continue to emerge, we can expect future systems to make even better contextual decisions, leading to more accurate and natural language interactions across multiple domains.

The potential for innovation through applications such as real-time transcription to email generation continues to grow, cementing Wav2Vec2's role at the forefront of speech-to-text technologies.

Disclosure of interest

This article does not has any conflict of interest with anyone.

Data Availability Statement

No data is gathered from any online/offline account/organization.

Funding

No funding was received.

Acknowledgements

We would like to express our sincere gratitude to OpenAI for the development of ChatGPT, an advanced language model that has significantly supported the writing, structuring, and refinement of this research. The model's capabilities in generating academic content, rephrasing complex ideas, and providing constructive feedback have greatly contributed to the clarity and coherence of this work.

REFERENCES

1. Baevski, W. Zhou, A. Mohamed, and M. Auli, "Wav2Vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," *Advances in Neural Information Processing Systems*, vol. 33, pp. 12449–12460, 2020.
2. Q. Liu, S. Yoon, and C. Kim, "Applications of ASR in Different Industries: An Overview," *Journal of Voice Technology*, vol. 14, no. 3, pp. 567–585, 2022.
3. A. Hannun *et al*., "Deep Speech: Scaling up end-to-end speech recognition," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2014, vol. 4, pp. 1895–1899.
4. J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proc. NAACL-HLT*, 2019.
5. N. Chen, Z. Xu, C. Lee, Y. Yu, and L. Lee, "A Deep Self-Supervised Model for Noisy Speech Recognition," in *Proc. ICASSP*, 2018.
6. A. Radford *et al*., "CLIP: Connecting Vision and Language with Transformers," in *Proc. Int. Conf. Machine Learning (ICML)*, 2021.
7. Facebook AI Research, "Wav2Vec2 Model: Code and Results," 2020. [Online]. Available: https://github.com/pytorch/fairseq

8. H. Yin *et al*., "Large-Scale End-to-End Speech Recognition with Wav2Vec 2.0," in *Proc. ICASSP*, 2021.
9. Y. Chen, Y. Weng, Y. Yang, and D. Batra, "Measuring the Language Understanding of AI Models," in *AI in Action: Deep Learning, NLP, and Beyond*, 2019.
10. Y. Liu *et al*., "RoBERTa: A Robustly Optimized BERT Pretraining Approach," in *Proc. ACL*, 2019.
11. J. Gehring *et al*., "Convolutional Sequence to Sequence Learning," in *Proc. ICML*, 2017.
12. Google Cloud, "Cloud Speech-to-Text Documentation." [Online]. Available: https://cloud.google.com/speech-to-text/docs
13. L. Chen and M. Asgari, "Interpreting a pre-trained model is a key for model architecture optimization: A case study on Wav2Vec 2.0," *arXiv preprint arXiv:2104.02851*, 2021.
14. F. Wu, K. Kim, J. Pan, K. J. Han, K. Q. Weinberger, and Y. Artzi, "Performance-efficiency trade-offs in unsupervised pre-training for speech recognition," in *Proc. ICASSP*, pp. 7667–7671, 2022.
15. L. Chen, M. Asgari, and H. H. Dodge, "Optimize Wav2vec2's architecture for small training set through analyzing its pre-trained models attention pattern," in *Proc. ICASSP*, pp. 7112–7116, 2022.
16. 161616 T. Grosz, Y. Getman, R. Al-Ghezi, A. Rouhe, and M. Kurimo, "Investigating wav2vec2 context representations and the effects of fine-tuning, a case-study of a Finnish model," in *Proc. Interspeech*, pp. 196–200, Aug. 2023.