UNIVERSITY OF SCIENCE AND TECHNOLOGY OF HANOI

Faculty of Information and Communication Technology

ASSIGNMENT REPORT
for
Natural Languages Processing

Automatic Speech Recognition

No.	Name	Student ID
1	Lương Ngọc Phúc	22BI13358
2	Nguyễn Quốc Thành	22BI13410
3	Lê Quang Minh	22BI13286
4	Nguyễn Minh Quân	22BI13374
5	Đỗ Hoàng Sơn	22BI13393
6	Lý Trần Gia Minh	22BI13288

I. The project goals.

- The goal of this project is to develop an Automatic Speech Recognition (ASR) system that converts spoken language into written text.
- This system aims to enhance accessibility, enable real-time transcription, and assist in speech-driven applications such as voice assistants, customer support automation, and transcription services.
- We aim to train and optimize an ASR model using deep learning-based approaches like Wav2Vec 2.0, Whisper, or DeepSpeech to improve accuracy.

II. The target audience or customers for the product

1. Business & Accessibility

A. Business & Enterprise Solutions

How ASR Benefits Businesses:

- Automates transcription for meetings, customer support, and corporate documentation, saving time and reducing costs.
- Enhances AI-driven customer support chatbots, virtual assistants, and call center analytics.
- Helps industries like legal, healthcare, and finance comply with documentation and regulatory requirements.

Key Applications:

- Call Centers & Customer Service – Converts voice interactions into text for analysis and chatbot automation.
- Corporate Meetings & Documentation – Integrates with platforms like Zoom, Teams for real-time transcription.
- Healthcare & Legal – Automates medical record-keeping and legal transcription for better workflow.

B. Accessibility & Inclusion

How ASR Improves Accessibility:

- Provides real-time captions for people with hearing impairments.

- Enhances learning with AI-driven transcriptions and multilingual subtitles.

Key Applications:

- Live Captioning for Deaf & Hard-of-Hearing Users – Instant speech-to-text conversion in classrooms and workplaces.
- Education & Learning Support – Improves online learning with AI-generated subtitles and lecture notes

2. Media & Smart Technology

A. Media, Journalism & Content Creation

How ASR Revolutionizes Media:

- Automates subtitles for videos, podcasts, and news broadcasts.
- Improves SEO & content discoverability by converting voice into searchable text.

Key Applications:

- Podcast & Video Transcription – Converts spoken content into articles, blogs, or summaries.
- Broadcast Captioning – Automates real-time subtitles for live events and multilingual content.
- Voice-Based Search – Helps media companies index voice content for keyword-based retrieval.

B. Smart Technology & AI-Powered Devices

How ASR Powers Smart Tech:

- Enables voice control in smart homes, virtual assistants, and automotive applications.
- Enhances user experience with AI-driven personalization and speech commands.

Key Applications:

- Voice Assistants (Alexa, Google Assistant, Siri) – Enables hands-free interaction with AI-powered smart assistants.

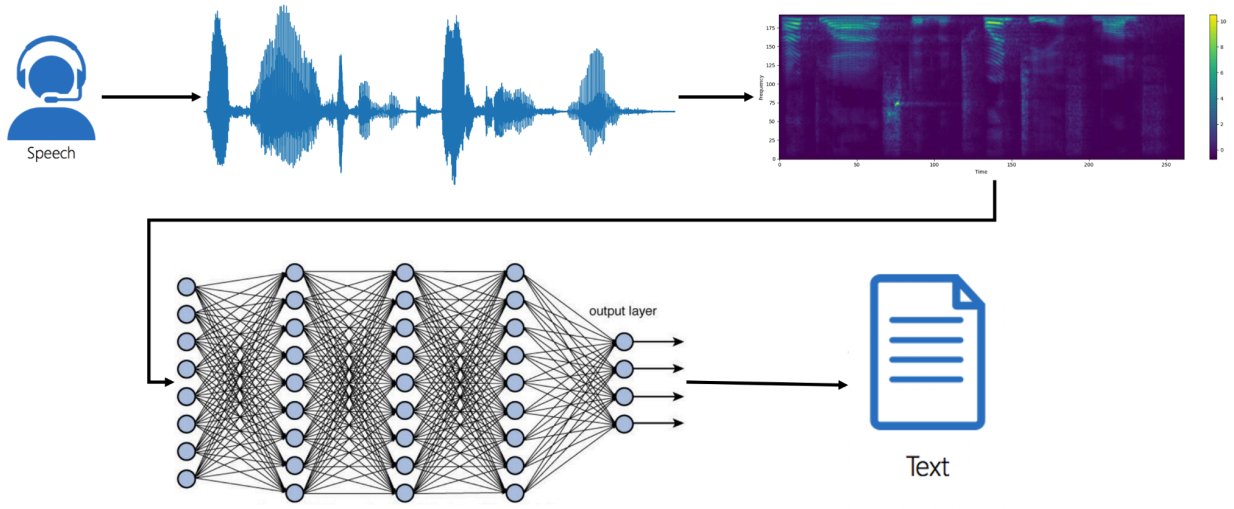
- Smart Home & IoT Devices – Controls devices with voice, improving accessibility.
- Automotive Speech Recognition – Reduces driver distraction with hands-free navigation and voice commands.

III. The methodology

1. Challenges in Speech Recognition

- Words that sound similar: For example, "to" and "too" sound identical but have different meanings and must be distinguished accurately.
- Words with multiple meanings: Words that have multiple meanings can make it challenging for the model to interpret correctly.
- Audio signal quality: Poor quality recordings can affect the model's ability to recognize speech accurately.
- Large training dataset required: The model needs to be trained on a large dataset of audio samples to achieve high accuracy.
- Multilingual recognition: The model must recognize and handle the phonetic and grammatical differences between languages.

2. Techniques in ASR



Machine learning and speech recognition: Machine learning has improved speech recognition accuracy by recognizing complex patterns, understanding natural language, and distinguishing languages.

Deep learning: Artificial neural networks are used to recognize complex patterns and are used in devices like Google Home, Amazon Alexa, and speech-to-text services.

Other techniques:

- Hidden Markov Models (HMM): A statistical approach for speech pattern recognition.
- Dynamic Time Warping (DTW): Compares temporal sequences to recognize similar speech patterns.
- Phonetic-based approaches: Recognize speech based on phonetic similarity.

Improvement techniques:

- Beamforming: Reduces background noise by focusing on the sound source.
- Noise cancellation: Subtracts background noise to improve accuracy.

Transformers: Released in 2018, transformers revolutionized speech recognition by improving accuracy, modeling long-term dependencies, and enhancing language recognition.

Applications: Improved models enable virtual assistants, voice-controlled interfaces, and speech-to-text transcription.

3. Building an ASR Model with TensorFlow

An Automatic Speech Recognition Tool converts spoken words into text using deep learning models. To construct an ASR model, we use TensorFlow. It provides powerful tools for training speech recognition models.

A. Dataset Preparation

The LJSpeech Dataset:

This dataset contains 13000 audio files in a ".wav" format. All the actual labels are also given to us in the annotation file. Some common datasets include:

- LJSpeech: English
- Common voice: Mozilla
- Google speech Commands: key recognition
- AISHELL-1: Vietnamese speech recognition

Preprocess audio:

- Convert “.wav” files to spectrograms using “tensorflow.signal” or “librosa”.
- Normalize and standardize input features.
- Process transcriptions.

B. The model architecture:

- CNN Layers:

Convolutional Neural Networks are a type of machine learning architecture that is mostly used for analyzing visual datasets. For speech recognition, CNNs take a spectrogram of the speech signal, represented as an image, and use these features to recognize speech.

- RNN (LSTM/GRU/BiLSTM):

RNNs are the preferred deep learning architecture for speech recognition because they are good at modeling sequential data. They can capture the long-term dependencies between the features in the input dataset and produce outputs based on past observations.

```python
import tensorflow as tf
from keras import layers
from keras.models import Model
from mltu.model_utils import residual_block, activation_layer

def train_model(input_dim, output_dim, activation='leaky_relu', dropout=0.2):
    inputs = layers.Input(shape=input_dim, name="input")
    input = layers.Lambda(lambda x: tf.expand_dims(x, axis=-1))(inputs)
    x = layers.Conv2D(filters=32, kernel_size=[11, 41], strides=[2, 2], padding="same", use_bias=False)(input)
    x = layers.BatchNormalization()(x)
    x = activation_layer(x, activation='leaky_relu')
    x = layers.Conv2D(filters=32, kernel_size=[11, 21], strides=[1, 2], padding="same", use_bias=False)(x)
    x = layers.BatchNormalization()(x)
    x = activation_layer(x, activation='leaky_relu')
    x = layers.Reshape((-1, x.shape[-2] * x.shape[-1]))(x)
    x = layers.Bidirectional(layers.LSTM(128, return_sequences=True))(x)
    x = layers.Dropout(dropout)(x)
    x = layers.Bidirectional(layers.LSTM(128, return_sequences=True))(x)
    x = layers.Dropout(dropout)(x)
    x = layers.Bidirectional(layers.LSTM(128, return_sequences=True))(x)
    x = layers.Dropout(dropout)(x)
    x = layers.Bidirectional(layers.LSTM(128, return_sequences=True))(x)
    x = layers.Dropout(dropout)(x)
    x = layers.Bidirectional(layers.LSTM(128, return_sequences=True))(x)
    x = layers.Dense(256)(x)
    x = activation_layer(x, activation='leaky_relu')
    x = layers.Dropout(dropout)(x)
    output = layers.Dense(output_dim + 1, activation="softmax")(x)

    model = Model(inputs=inputs, outputs=output)
    return model
```

C. Training the models:

- Use CTC loss for training ASR models with varying-length sequences.

- Train on a large dataset using data augmentation techniques to improve robustness.
- Consider using Transfer Learning with models like DeepSpeech, Wav2 Vec2.0 or Whisper.

```python
import typing
import numpy as np
from mltu.inferenceModel import OnnxInferenceModel
from mltu.preprocessors import WavReader
from mltu.utils.text_utils import ctc_decoder, get_cer, get_wer

class WavToTextModel(OnnxInferenceModel):
    def __init__(self, char_list: typing.Union[str, list], *args, **kwargs):
        super().__init__(*args, **kwargs)
        self.char_list = char_list
    def predict(self, data: np.ndarray):
        data_pred = np.expand_dims(data, axis=0)
        preds = self.model.run(None, {self.input_name: data_pred})[0]
        text = ctc_decoder(preds, self.char_list)[0]
        return text

if __name__ == "__main__":
    import pandas as pd
    from tqdm import tqdm
    from mltu.configs import BaseModelConfigs
    configs = BaseModelConfigs.load("Models/05_sound_to_text/202302051936/configs.yaml")
    model = WavToTextModel(model_path=configs.model_path, char_list=configs.vocab, force_cpu=False)
    df = pd.read_csv("Models/05_sound_to_text/202302051936/val.csv").values.tolist()

    accum_cer, accum_wer = [], []
    for wav_path, label in tqdm(df):
        spectrogram = WavReader.get_spectrogram(wav_path, frame_length=configs.frame_length, frame_step=configs.frame_step, fft_length=configs.fft_length)
        padded_spectrogram = np.pad(spectrogram, (0, (configs.max_spectrogram_length - spectrogram.shape[0]),(0,0)), mode='constant', constant_values=0)
        text = model.predict(padded_spectrogram)
        true_label = "".join([l for l in label.lower() if l in configs.vocab])
        cer = get_cer(text, true_label)
        wer = get_wer(text, true_label)
        accum_cer.append(cer)
        accum_wer.append(wer)
    print(f"Average CER: {np.average(accum_cer)}, Average WER: {np.average(accum_wer)}")
```

D. Deployment

- Convert the trained model to TensorFlow Lite (TFLite) for mobile applications.
- Use ONNX for deployment in various environments.
- Create an API using FastAPI or Flask to integrate ASRT into applications.

4. Applications of ASR

A. Virtual Assistants

- Personalization through ASR: Virtual assistants are becoming increasingly personalized, adapting to users' preferences and speech patterns over time. ASR helps these devices learn and improve responses based on context, making interactions smoother. For example, virtual assistants can remember previous interactions, suggest reminders, or even offer location-based advice based on voice recognition.

- Multilingual Support: ASR is enabling virtual assistants to understand and respond in multiple languages. For example, Google Assistant can seamlessly switch between languages in a conversation. This is especially beneficial in multicultural and multilingual environments.
- Integration with IoT: ASR plays a key role in the growing Internet of Things (IoT). Devices like smart thermostats, refrigerators, and lights are controlled by voice commands. As more devices integrate with virtual assistants, ASR will be critical for managing smart homes and businesses efficiently.

B. Customer Service

ASR is widely used in call centers to automate transcription, sentiment analysis, and call routing, improving efficiency and customer satisfaction. It allows companies to offer 24/7 support while minimizing human intervention.

C. Healthcare

In healthcare, ASR helps professionals dictate medical notes and reports, streamlining documentation processes. This reduces administrative workload and ensures accurate, real-time medical records.
These applications demonstrate how ASR technology is transforming various industries by improving efficiency and accessibility.

5. Future Trends & Developments

A. Enhanced Accuracy and Contextual Understanding through Advanced Deep Learning

- Why it's likely: As AI and machine learning technologies continue to evolve, deep learning models—particularly transformers like GPT and BERT—are significantly improving the accuracy of ASR systems. These models don't just recognize words, but also understand the context in which they are spoken. This means ASR systems will become better at distinguishing accents, dialects, and speech in noisy environments. Such advancements are not just theoretical; they're happening now, with continuous improvements expected in the near future.
- Impact: The improved accuracy and contextual awareness will make ASR systems more reliable across a variety of applications. For example, they will enhance customer service experiences, enable more precise healthcare documentation, and

provide real-time communication solutions with fewer misunderstandings, even in challenging environments..

B. Real-Time Multilingual Translation

- Why it's likely: In an increasingly interconnected world, the demand for real-time multilingual communication is growing. ASR is increasingly being integrated with machine translation models to create systems that can seamlessly translate spoken language in real-time. This development is already underway with applications like Google Translate, and as ASR and translation technologies improve, the ability to communicate fluently in multiple languages will become more refined and accessible.
- Impact: Real-time multilingual translation will have a profound impact on global business, travel, and diplomacy, removing language barriers and facilitating smoother communication across cultures. It will also make information more accessible to people worldwide, especially in regions where language diversity has previously hindered access to key services or resources. This trend will help bridge the gap between different languages and foster better cross-cultural understanding.

IV. The achieved results.

- **Model Performance**: The speech recognition model achieved a Character Error Rate (CER) of 1.7% and a Word Error Rate (WER) of 7%. This indicates that the model is highly accurate in recognizing spoken words and characters from audio input.
- **Real-Time Inference**: The model is capable of real-time speech recognition, which means it can convert spoken words into text almost instantly. This makes it suitable for applications like virtual assistants and transcription services.
- **Training Process**: The model underwent training with a dropout rate of 0.5 to prevent overfitting. The training process was monitored using TensorBoard, and the performance was evaluated using CER and WER metrics.
- **Inference Testing**: After training, the model was tested on validation data, where it demonstrated satisfactory results. The validation data showed a CER of 1.7% and a WER of 7%, indicating the model's effectiveness in recognizing speech.

Overall, the implemented model is highly effective in recognizing speech with a low error rate, making it suitable for various applications that require accurate and real-time speech recognition.