German ASR Evaluation and Error Classification

Noureldin Mohamed

September 2023

Abstract

This project presents a comprehensive analysis of three Automatic Speech Recognition (ASR) systems, employing the State-of-the-Channel Toolkit (SCTK) to assess their performance. The primary objective is to evaluate and compare the ASR systems based on key metrics and error classification techniques.

Using SCTK, we conduct a comprehensive comparison of the ASR systems, producing reports that include metrics such as Word Error Rate (WER), Sentence with Errors, Word Substitutions, Word Deletions, Word Insertions, and Word Accuracy. These metrics serve as vital indicators of ASR system performance and accuracy.

Additionally, we delve into error analysis and classification, monitoring various types of errors for each ASR system. By categorizing errors, we gain valuable insights into the specific challenges and limitations of each system. This error classification process allows for a deeper understanding of the strengths and weaknesses of the systems, facilitating targeted improvements and optimizations.

To facilitate a nuanced assessment, we organize the results into Excel tables that provide a clear overview of performance percentages at various levels. These tables break down the metrics by subfolders, main folders, and for the ASR systems as a whole. This hierarchical presentation enables a fine-grained analysis of ASR performance across different data subsets.

In summary, this project offers a comprehensive evaluation of ASR systems using SCTK metrics and error classification techniques. The generated comparison reports, along with detailed error monitoring, provide a clear and in-depth assessment of ASR system performance, enabling informed decision-making for various applications reliant on accurate speech recognition technology.

Contents

List of Figures

Chapter 1

Introduction

1.1 Problem Statement

Over the past few decades, there has been significant growth and widespread use of speech-based human-machine interaction and natural language understanding applications. Consequently, there has been a surge in research efforts aimed at identifying and categorizing errors in Automatic Speech Recognition (ASR) systems. ASR systems, integral to various applications like transcription services and voice assistants, face an enduring challenge in ensuring their precision and dependability. This project seeks to address this challenge by conducting a comprehensive evaluation and comparison of three distinct ASR systems and by implementing a systematic approach to error classification during the speech recognition process.[1]

The primary problem at hand is the absence of a comprehensive and objective assessment framework for ASR systems. Traditional evaluations often focus on limited metrics, such as Word Error Rate (WER), without delving into the diverse range of errors that can affect ASR performance. This lack of granularity makes it challenging to pinpoint the precise shortcomings of each ASR system and hinders efforts to optimize and improve them.

Furthermore, the evaluation process typically lacks a systematic approach to error classification. Understanding the types of errors occurring in ASR output is crucial for identifying areas that require attention and refinement. Without a structured method for error classification, it becomes difficult to diagnose the specific issues that may be hindering system performance and to take targeted corrective measures.

In addition to these challenges, speech-based human-machine interaction and natural language understanding applications have seen a rapid development and wide adoption over the last few decades. This has led to a proliferation of studies that investigate error detection and classification in Automatic Speech Recognition (ASR) systems. As ASR technology plays an increasingly integral role in human-computer interaction, the need for robust evaluation and error analysis methods becomes ever more pressing.

This project seeks to bridge the gap by not only evaluating ASR systems using comprehensive metrics but also by implementing a systematic error classification methodology. By addressing these issues, the project aims to enhance our understanding of ASR system strengths and weaknesses, ultimately leading to more accurate and reliable speech recognition technology for a wide range of applications.

1.2 Research Motivation

The motivation for this project stems from the increasing significance of Automatic Speech Recognition (ASR) systems in modern human-machine interaction and natural language processing applications. ASR technology has evolved rapidly and is now embedded in numerous facets of our daily lives, from voice assistants on smartphones to transcription services for professionals. However, despite these advancements, several critical challenges persist, prompting the need for comprehensive evaluation and error classification:

- **Growing Dependence on ASR Systems**: The widespread adoption of ASR systems in applications such as virtual assistants, customer service, and transcription services has made them indispensable. Any errors or inaccuracies in these systems can lead to miscommunications, inconvenience, and reduced user satisfaction.[2]
- **Diverse Usage Scenarios**: ASR systems are used in diverse environments and scenarios, including noisy environments, various accents and dialects, and different speaking speeds. Understanding how ASR systems perform under these diverse conditions is crucial to their practicality and reliability.
- **Lack of Comprehensive Evaluation**: Traditional ASR evaluations often rely on limited metrics like Word Error Rate (WER) without providing a nuanced understanding of system performance. This project aims to address this gap by conducting a more thorough

and holistic evaluation of ASR systems.[3]

- **Error Classification for Improvement**: Beyond quantitative metrics, the systematic classification of errors is essential for diagnosing the specific weaknesses of ASR systems. This project seeks to develop a structured framework for error classification, enabling targeted improvements in system accuracy.
- **Continuous Innovation and Improvement**: As ASR technology evolves, it is imperative to continually assess its performance to keep pace with user expectations and emerging applications. This project serves as a catalyst for ongoing innovation in ASR technology.
- **Real-World Applications**: The practical implications of this research are substantial, as improved ASR systems can enhance accessibility, productivity, and convenience across various sectors, including healthcare, education, and entertainment.
- **Academic and Industry Relevance**: Advancements in ASR technology benefit both academic research and industry applications. This project contributes valuable insights that can inform future research directions and lead to better ASR systems in commercial settings.

In summary, this project's research motivation arises from the pressing need to address the challenges associated with ASR systems in contemporary applications. By conducting a comprehensive evaluation and systematic error classification, it aims to contribute to the improvement of ASR technology and its broader impact on human-machine interaction and natural language understanding.

1.3 Dissertation Structure

Chapter 1 of this research provides an introduction to the problem statement,research's motivation. **Chapter 2** is dedicated to providing a comprehensive background on the subject matter to aid readers' comprehension of the research. The chapter includes a review of existing literature that has exploredways to evaluate ASR systems, and classify the error types. In **Chapter 3**, the research methodology is presented. This chapter outlines how the study will be conducted, and the implementation of various functions and methods to claassify the error types. Finally, **Chapter 4** will discuss the results and mention the limitations of the project.

Chapter 2

Literature Review

2.1 What is ASR

Automatic Speech Recognition (ASR) is a technology that converts spoken language into written text or machine-readable data. It is a fundamental component of natural language processing (NLP) and human-computer interaction systems. ASR systems are designed to understand and transcribe spoken language, making it accessible for various applications. These applications range from voice assistants and transcription services to customer service chatbots and voice-controlled devices.[4]

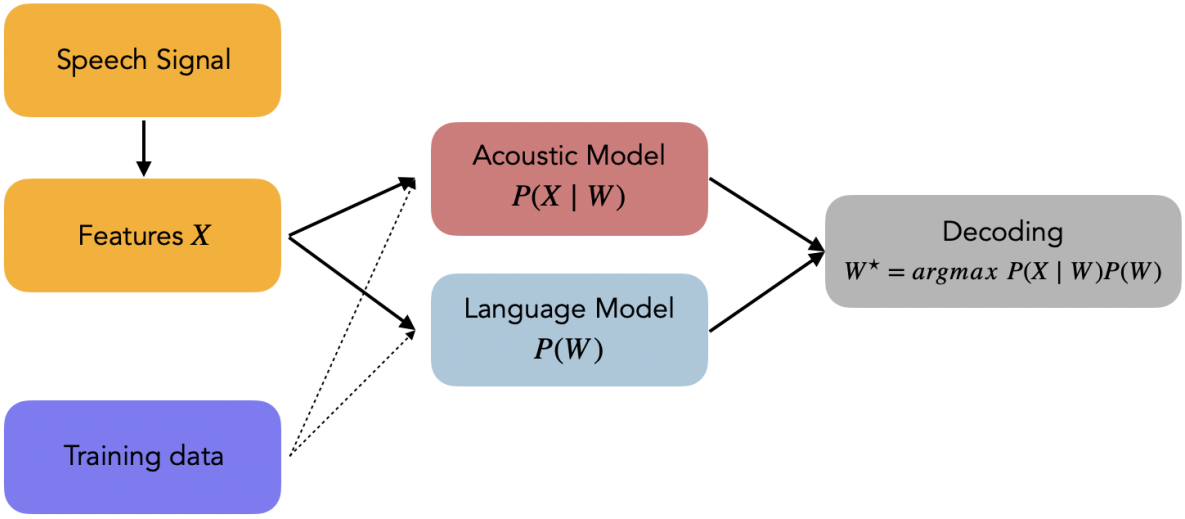

Figure 2.1: ASR Architecture.

ASR systems utilize sophisticated algorithms and machine learning techniques to analyze audio input and transform it into textual output. This transformation involves multiple stages, including acoustic modeling, language modeling, and decoding. Here's a brief overview of these stages:

- **Acoustic Modeling**: In this stage, ASR systems analyze the acoustic properties of the incoming speech signal. This involves breaking down the audio into small segments (phonemes) and mapping them to corresponding sounds in the target language. Deep learning techniques, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), are often used to model acoustic features.[17]
- **Language Modeling**: Language modeling is essential for deciphering spoken language and predicting the most likely words or phrases given the audio input. Statistical language models, n-gram models, and more recently, transformer-based models, are employed to capture the linguistic context and improve recognition accuracy.[17]
- **Decoding**: In the decoding stage, ASR systems use the information from the acoustic and language models to generate a transcription or textual representation of the spoken content. This process involves selecting the most probable words or phrases based on the input audio.[17]

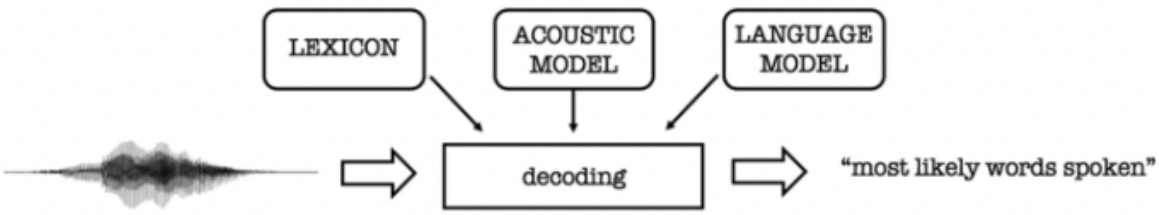

Figure 2.2: ASR Stages.

ASR systems have evolved significantly in recent years, thanks to advances in deep learning and neural network architectures. Modern ASR models, such as deep neural networks (DNNs) and transformer-based models, have significantly improved the accuracy and robustness of speech recognition.[4]

ASR technology finds applications in various domains, including:

- **Transcription Services**: ASR systems can automatically transcribe spoken content, making it useful for creating written records of meetings, interviews, and other spoken interactions.[18]
- **Voice Assistants**: Voice-activated digital assistants like Siri, Google Assistant, and Amazon Alexa rely on ASR to understand and respond to spoken commands and queries.[18]
- **Accessibility**: ASR technology aids individuals with disabilities by providing speech-to-text functionality in real-time, enabling them to interact with computers and devices more effectively.
- **Customer Service**: Many companies use ASR in their customer service operations, allowing automated systems to understand and assist customers via phone or chat.[18]

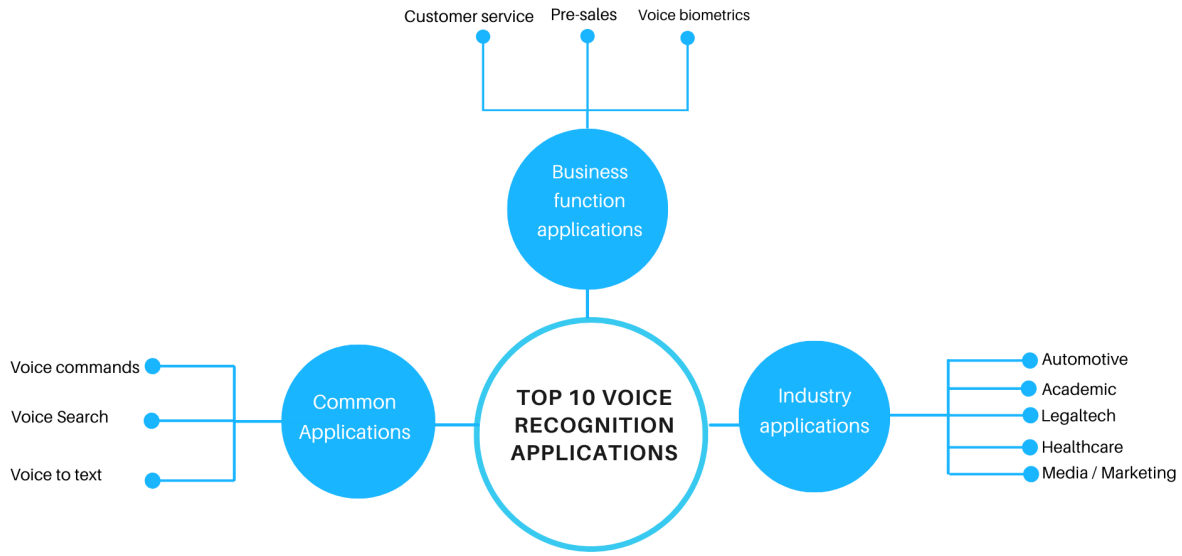

Figure 2.3: ASR Applications.

In summary, ASR technology plays a pivotal role in converting spoken language into a format that can be processed by computers and applications. Its continued advancement has the potential to enhance human-computer interaction and accessibility across a wide range of fields.

2.2 Importance for Evaluating ASR

Evaluating ASR (Automatic Speech Recognition) systems is of paramount importance due to their central role in modern technology and communication. Accurate and reliable speech recognition has a direct impact on user experience and the effectiveness of various applications. Evaluations provide critical insights into the performance and limitations of ASR systems, allowing developers and researchers to identify areas for improvement. They enable the selection of the most suitable ASR system for specific applications, ensuring optimal functionality. Additionally, evaluations facilitate fair comparisons among different ASR technologies, fostering healthy competition and innovation in the field. Overall, the evaluation of ASR systems is essential for advancing the quality and accessibility of speech-driven interfaces, transcription services, and other voice-enabled applications that have become integral to our digital lives.

2.3 Evolution of Automatic Speech Recognition (ASR)

The history of Automatic Speech Recognition (ASR) is a testament to the relentless pursuit of improving human-computer interaction through speech technology. ASR has undergone a remarkable evolution over the decades, marked by significant advancements in both the underlying technology and its practical applications.

2.3.1 Early Developments

The roots of ASR can be traced back to the mid-20th century when researchers began exploring methods to convert spoken language into machine-readable text. Early ASR systems relied on simplistic pattern recognition techniques and limited vocabularies. These pioneering efforts laid the groundwork for subsequent developments.[19]

2.3.2 Hidden Markov Models (HMMs)

A pivotal moment in ASR's evolution came with the adoption of Hidden Markov Models (HMMs) in the 1970s and 1980s. HMMs allowed for the modeling of sequential data, making it possible to account for the dynamic nature of speech. This marked a significant improvement in ASR accuracy and laid the foundation for many subsequent ASR systems.[20]

2.3.3 Statistical Models and Gaussian Mixture Models (GMMs)

The late 20th century saw the rise of statistical approaches in ASR, with the introduction of Gaussian Mixture Models (GMMs) and Maximum Likelihood Estimation. These statistical methods greatly enhanced the modeling of speech acoustics and language context, leading to substantial improvements in recognition accuracy.[21]

2.3.4 The Deep Learning Revolution

One of the most transformative moments in ASR's evolution occurred with the advent of deep learning in the 2010s. Deep Neural Networks (DNNs) and Convolutional Neural Networks (CNNs) brought unprecedented accuracy gains to ASR. This shift towards neural network-based models led to a paradigm shift, enabling ASR systems to achieve near-human-level performance on a wide range of tasks.[31]

2.3.5 End-to-End ASR

Recent years have witnessed the emergence of end-to-end ASR systems, where acoustic, linguistic, and decoding components are integrated into a single neural architecture. Transformer-based models, such as the Listen, Attend, and Spell (LAS) and Conformer models, exemplify this trend. End-to-end ASR simplifies system design and training, further improving ASR accuracy and making it more accessible for various applications.[32]

2.3.6 Real-World Applications

Today, ASR technology is integral to a multitude of applications, including voice assistants, transcription services, call center automation, and accessibility tools for individuals with speech impairments. Its evolution has not only revolutionized these fields but also opened new frontiers, such as real-time translation, voice-controlled devices, and personalized AI interactions.[18]

In conclusion, the evolution of ASR from its early beginnings to the era of deep learning and end-to-end systems reflects the relentless pursuit of improving speech recognition accuracy and usability. ASR's transformative impact on technology and communication continues to shape our digital world, with ongoing research and innovation promising even more exciting developments in the future.

2.4 Ways used to evaluate an ASR Model

Evaluating the performance of an Automatic Speech Recognition (ASR) model is a critical step in assessing its accuracy and effectiveness. Various methods and metrics are employed to measure the quality of ASR output, providing valuable insights for researchers, developers, and end-users. Here, we delve into the primary ways used to evaluate ASR models:

2.4.1 Word Error Rate (WER)

Definition: WER is a fundamental metric that quantifies the difference between the ASR-generated transcription and the reference (ground truth) transcript, considering word-level substitutions, insertions, and deletions.[5]

Use Case: WER is widely used to assess overall ASR accuracy. Lower WER values indicate better performance.

$$WER = \frac{S + D + I}{N}$$

where...
S = number of substitutions
D = number of deletions
I = number of insertions
N = number of words in the reference

Figure 2.4: WER Calculation.

2.4.2 Character Error Rate (CER)

Definition: CER is similar to WER but operates at the character level. It measures the divergence between ASR-generated and reference text in terms of individual characters.[6]

Use Case: CER can be more sensitive to errors, particularly in languages with complex phonetics.

$$CER = \frac{S + D + I}{N}$$

S = Number of Substitutions
D = Number of Deletions
I = Number of Insertions
N = Number of characters in ground truth

Figure 2.5: CER Calculation.

2.4.3 Token Error Rate (TER)

Definition: TER extends WER by considering the units of text (tokens) instead of individual words. It allows for a more flexible evaluation in languages with varying token lengths.[7]

Use Case: TER is beneficial when evaluating ASR systems in languages with complex morphologies or agglutination.

2.4.4 Precision, Recall, and F1-Score

Definition: Precision measures the proportion of correctly recognized words or tokens out of all ASR outputs. Recall assesses the proportion of correctly recognized words or tokens out of all reference words or tokens. F1-score combines both precision and recall into a single metric.[8]

Use Case: These metrics offer a balanced assessment of ASR performance, particularly when different errors (substitutions, insertions, deletions) need to be considered separately.

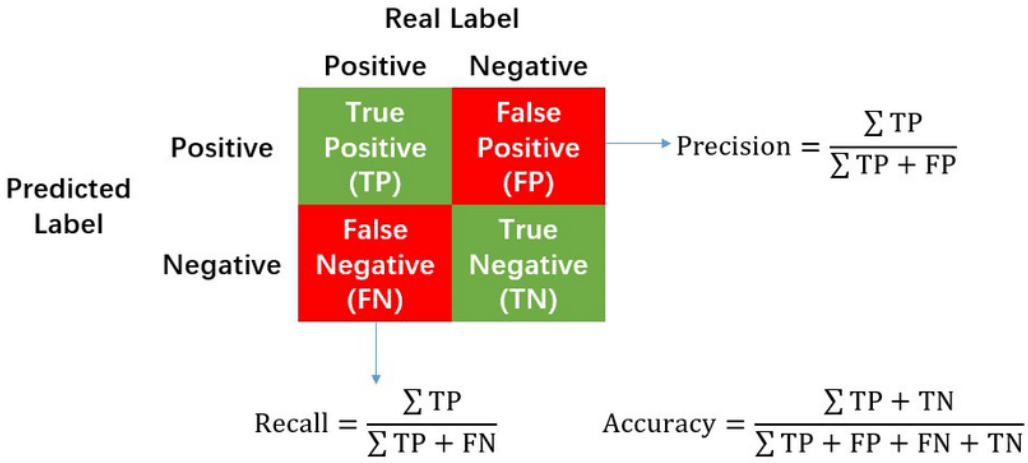

Figure 2.6: Precision, Recall, and F1-Score.

2.4.5 Perplexity and Language Modeling Scores

Definition: Perplexity and language modeling scores measure how well the ASR system's language model predicts the given transcript. Lower perplexity values indicate better language model performance.[9]

Use Case: These metrics help evaluate the ASR model's ability to generate coherent and contextually accurate transcriptions.

$$PP(s) = 2^{log_2^{PP(s)}} = 2^{-\frac{1}{n}log(p(s))}$$

$$\text{let } l = \frac{1}{n}log(p(s))$$

$$\text{For unigram } l = \frac{1}{n}(logp(w_1) + \cdots + logp(w_n))$$

$$\text{For bigram } l = \frac{1}{n}(logp(w_1) + logp(w_2|w_1) + \cdots + logp(w_n|w_{n-1}))$$

Figure 2.7: Perplexity and Language Modeling Scores.

2.4.6 Signal-to-Noise Ratio (SNR)

Definition: SNR measures the quality of the audio input. It quantifies the ratio of the signal power (speech) to noise power.[10]

Use Case: Assessing SNR helps gauge ASR system performance in noisy environments, where lower SNR values may result in recognition difficulties.

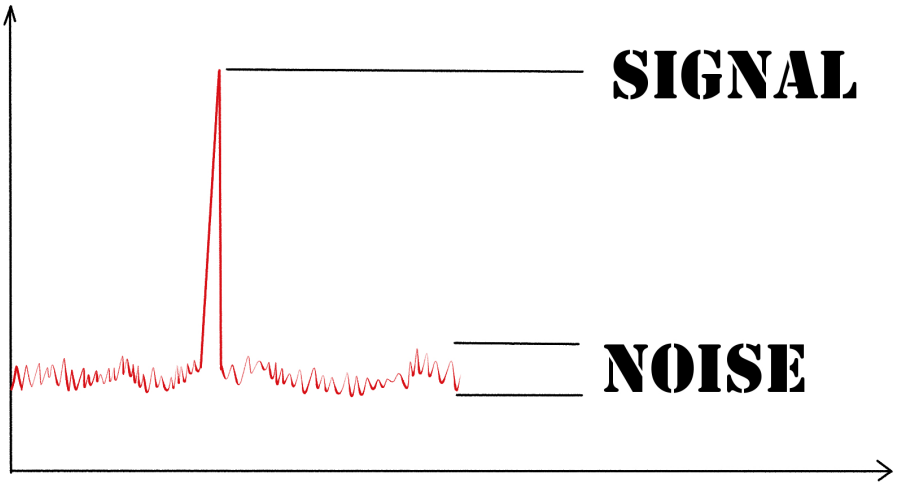

Figure 2.8: Perplexity and Language Modeling Scores.

2.4.7 End-to-End Metrics

Definition: End-to-end metrics assess the entire ASR system's performance, including both acoustic modeling and language modeling components.[11]

Use Case: These metrics offer a holistic view of ASR accuracy, especially in end-to-end ASR systems that integrate multiple components into a single architecture.

2.4.8 Subjective Evaluation

Definition: Subjective evaluation involves human annotators listening to ASR-generated audio and transcripts to assess their quality, fluency, and naturalness.[12]

Use Case: Subjective evaluation captures aspects of ASR performance that automated metrics may miss, such as speaker intonation and overall user satisfaction.

Choosing the appropriate evaluation method and metric depends on the specific goals and requirements of the ASR application, as well as the available resources. A combination of these methods often provides a comprehensive understanding of an ASR model's performance, allowing for informed decisions and continuous improvement.

Chapter 3

Methodology

In this research we will dive deeper into evaluating ASR Models using **WER**.

3.1 Libraries Used

In this section, we will discuss the libraries used in our Python code.

3.1.1 pandas

Pandas is a versatile data manipulation library that provides data structures like DataFrame and Series. It excels at handling structured data, allowing you to load, clean, transform, and analyze data efficiently. With Pandas, you can perform tasks like data filtering, aggregation, and merging, making it a fundamental library for data analysis and manipulation.[22]

3.1.2 subprocess

The **subprocess** module is used for managing and interacting with external processes from within your Python script. It enables you to run system commands, capture their output, and manage input and error streams. This library is essential for automating system-level tasks and executing external programs.[23]

3.1.3 re

The **re** (regular expression) module provides powerful tools for working with regular expressions. Regular expressions are patterns used for text matching and manipulation. With this library,

you can search, extract, and manipulate text based on complex patterns, making it invaluable for tasks like text parsing and validation.[24]

3.1.4 os

The **os** module is a cross-platform library for interacting with the operating system. It allows you to perform operations such as file and directory manipulation, path management, and environment variable access. This library is crucial for managing files, directories, and system-related tasks.[25]

3.1.5 openpyxl

Openpyxl is a Python library for working with Excel files. It enables you to create, read, and modify Excel spreadsheets programmatically. You can manipulate worksheets, cells, formatting, and formulas, making it an excellent choice for tasks involving Excel data.[26]

3.1.6 shutil

The **shutil** library provides high-level file operations for tasks like copying, moving, renaming, and deleting files and directories. It simplifies file management and is particularly useful for batch processing and data organization.[27]

3.1.7 spacy

Spacy is a natural language processing (NLP) library that comes with pre-trained models for various languages. You are using the "de_core_news_sm" model, which is specifically designed for German text processing. Spacy offers tools for tokenization, named entity recognition, part-of-speech tagging, and more, making it essential for NLP tasks.[28]

3.1.8 enchant

Enchant is a library for spellchecking and word validation. It provides a way to check the spelling of words and suggest corrections. It can be helpful for text quality assurance and ensuring correct spelling in text data.[29]

3.1.9 collections.Counter

The **collections** module includes the **Counter** class, which is used for counting elements in a collection. It is particularly useful for tasks involving frequency analysis, such as counting word occurrences, finding the most common elements, and generating histograms.[30]

3.2 SCTK-SCLite

SCTK SCLite (Score Confidence Lite) is a software tool developed by the National Institute of Standards and Technology (NIST) as part of the Speech Recognition Scoring Toolkit (SCTK). It is a powerful and versatile utility used for scoring the performance of automatic speech recognition (ASR) systems or aligning reference transcriptions with the corresponding ASR outputs. SCLite is primarily designed for evaluating the quality and accuracy of ASR systems, making it an essential tool in the field of automatic speech recognition research and evaluation.[14]

3.2.1 Scoring ASR Output:

One of the primary purposes of SCLite is to score the output of ASR systems. It compares the ASR-generated transcripts (hypotheses) with reference transcriptions (ground truth) and provides various scoring metrics to evaluate the ASR system's performance.[14]

3.2.2 Word and Sentence Level Alignment:

SCLite performs both word-level and sentence-level alignments between ASR hypotheses and reference transcriptions. This means it can determine which words or phrases in the ASR output correspond to those in the reference, even in the presence of insertions, deletions, and substitutions.[14]

3.2.3 Evaluation Metrics

SCLite computes several standard evaluation metrics to quantify ASR system performance, including Word Error Rate (WER), Sentence Error Rate (SER), and Alignment Error Rate (AER). These metrics help researchers and developers understand how well an ASR system performs in terms of accuracy and intelligibility.

3.2.4 Confidence Scoring

SCLite can be used to compute confidence scores for ASR hypotheses. These scores provide an estimate of the reliability of each word in the ASR output. Confidence scoring is crucial for understanding when an ASR system is uncertain about its transcriptions, which can be valuable information in applications like transcription post-processing.[14]

3.2.5 Flexible Input Formats

SCLite can handle various input formats, including NIST's Lattice format and the popular HTK format. This flexibility allows it to work with different ASR systems and adapt to the specific data and tools being used in a given evaluation.[33]

3.2.6 Output Options

The tool provides options to generate detailed output reports, including aligned transcripts, scoring results, and confidence scores. These reports help users gain insights into the strengths and weaknesses of their ASR systems.[14]

3.2.7 Customization

SCLite offers a range of configuration options to customize scoring behavior. Users can tailor the scoring process to suit their specific evaluation needs, making it a versatile tool for various ASR evaluation scenarios.

3.2.8 Community Usage

SCLite is widely adopted in the ASR research community and is commonly used in benchmark evaluations like the NIST Open Speech Recognition Evaluation (SRE) and the IARPA Babel program. Its acceptance and usage by researchers highlight its reliability and utility in ASR evaluations.

3.2.9 Extra reports generated by SCLITE

SCTK SCLite provides several optional features and tools that can be used to enhance the evaluation and analysis of ASR systems. pralign (Phoneme-Level Alignment):

pralign (Phoneme-Level Alignment)

SCTK SCLite includes the pralign tool, which performs phoneme-level alignment between ASR hypotheses and reference transcriptions. This feature is particularly valuable when you need to assess the alignment of phonemes, allowing for a more detailed analysis of ASR system performance at the phonetic level.[14]

prf (Precision, Recall, and F-Measure)

The prf tool calculates precision, recall, and F-measure scores. These metrics are commonly used in information retrieval and information retrieval-related tasks. In the context of ASR evaluation, they can provide additional insights into the quality of the ASR system's output by measuring its ability to correctly transcribe specific classes of words or phrases.[14]

rsum (Reference Summary)

The rsum tool generates a summary of the reference transcripts. This feature can be helpful when you want a concise overview of the reference data for further analysis or reporting purposes. It condenses the reference data into a more manageable format, making it easier to analyze.[14]

sum (System Summary)

Similarly, the sum tool generates a summary of the ASR system's hypotheses. This summary can be useful for quickly assessing the overall performance of the ASR system, especially in scenarios where you want a high-level overview of the output without diving into detailed transcripts.[14]

dtl (Detailed Transcript Comparison)

The dtl tool provides a detailed comparison of ASR hypotheses and reference transcriptions. It highlights the specific differences and alignments between the two, including insertions, deletions, and substitutions. This fine-grained analysis is invaluable for diagnosing and understanding the errors made by the ASR system. These additional features extend the capabilities of SCTK SCLite, allowing users to perform in-depth analyses of ASR system outputs. Depending on the specific evaluation goals and requirements, users can choose to leverage these features to gain a more comprehensive understanding of the ASR system's performance, identify areas for improvement, and fine-tune their ASR models and processes.[14]

3.2.10 Installation Steps

1. **Clone the SCTK Repository**:

   Open a terminal and clone the SCTK repository from GitHub using Git:

   ```
   git clone https://github.com/usnistgov/SCTK.git
   ```

2. **Build SCTK**:

   Navigate to the SCTK directory and build the toolkit using the provided `buildSCTK.sh` script:

   ```
   cd SCTK
   ./configure
   make
   ```

3. **Install SCTK (Optional)**:

   To install SCTK globally on your system, you can run:

   ```
   make install
   ```

   This step is optional and depends on your preferences. Installing SCTK globally allows you to access its tools from any directory.

4. **Set Environment Variables**:

   To make SCTK SCLite and other SCTK tools easily accessible from any directory, add the SCTK bin directory to your PATH environment variable. Add the following line to your shell configuration file (e.g., `/.bashrc` or `/.zshrc`):

   ```
   export PATH="/path/to/SCTK/bin:$PATH"
   ```

   Replace `/path/to/SCTK` with the actual path to your SCTK installation directory.

5. **Verify Installation**:

   To verify that SCTK SCLite is installed correctly, you can run the following command:

```
sclite
```

If SCTK SCLite is installed, you will see its command-line usage and options.

3.2.11 Usage with Docker

Alternatively, you can use SCTK SCLite within a Docker container. Build the Docker image and use it to run SCTK SCLite as follows:

```
docker build -t sctk .
```

```
docker run -it -v $PWD:/var/sctk sctk sclite -i wsj -r ref.txt -h hyp.txt
```

Replace `ref.txt` and `hyp.txt` with your reference and ASR hypothesis files, and adjust the paths and options as needed.[14]

That's it! You've successfully installed SCTK SCLite and can now use it to evaluate ASR system performance and conduct detailed analyses of ASR transcripts.

3.3 Sequence of Code Execution

3.3.1 Initialization

- Import necessary Python libraries.
- Define key parameters, such as base_directory, report_directory, hypothesisfoldername, and numofmostrepeated.
- The **base_directory** refers to the directory that contains the ASR systems.
- The **report_directory** refers to the directory where the generated reports would be stored.
- The **hypothesisfoldername** refers to the name of the hypothesis folder
- The **numofmostrepeated** refers to the number of most repeated errors per ASR system.

3.3.2 ASR System Comparison (asrcomparelogic function)

- Calculate and compare ASR system performance metrics

1. Calculate Word Error Rate (WER) for ASR1, ASR2, and ASR3.
2. Calculate sentence-level metrics: Sentence Error Rate (SER), Sentence Accuracy.
3. Calculate word-level metrics: Word Substitutions, Word Deletions, Word Insertions, Word Accuracy.

- Create two types of tables:
  1. Comparison Table: Compares ASR systems across subfolders and subfolder aggregates.
  2. Final Performance Table: Aggregates metrics for each ASR system (ASR1, ASR2, ASR3) across main subfolders and subfolder aggregates.
- Format and save these tables as Excel files (folder_values.xlsx and final_values.xlsx).

3.3.3 Error Analysis (errorlogic function)

- Generate ASR system comparison reports using the provided data.
- Organize and fix report files into a structured directory (report_directory) for analysis.
- Extract error-related information for each report
  1. ASR System (ASR), Reference Word (Ref), Hypothesis Word (Hyp), Location (Loc), Error Frequency (Freq).
- Create an "Error Extraction Table" that displays extracted error information.
- Perform error classification for each error type
  1. Noun-Article Mismatch, Compound Word Split, Hesitation, Phonetic Error, Deletion Error, Partial Error, Not a German Word.
- Generate an error summary table, listing the most common error types and their frequencies.
- Format and save the error-related tables as Excel files (Error Extraction Table .xlsx and Error Extraction Summary Table .xlsx).

3.3.4 Excel Formatting

This includes the following:

- borderFormatter
- mergeFormatter
- formatTables
- formaterrorrep
- formaterrorsumreport

These functions are responsible for formatting the output Excel files to enhance readability, consistency, and presentation:

- Adding borders to specific cells for emphasis.
- Merging cells for improved visual structure.
- Adjusting column widths to accommodate data.

3.3.5 Main Execution

- Execute the ASR system comparison logic (asrcomparelogic) to calculate performance metrics and create tables.
- Execute the error analysis logic (errorlogic) to extract, classify, and summarize errors.

3.3.6 Output Files

- Excel files containing ASR system comparison results (folder_values.xlsx and final_values.xlsx).
- Excel files containing error extraction and classification (Error Extraction Table .xlsx and Error Extraction Summary Table .xlsx).
- A summary of the most common error types (Most Error Classification table.xlsx).
- A frequency table of error types (Error Classification Frequency table.xlsx).

3.4 Directory Layout

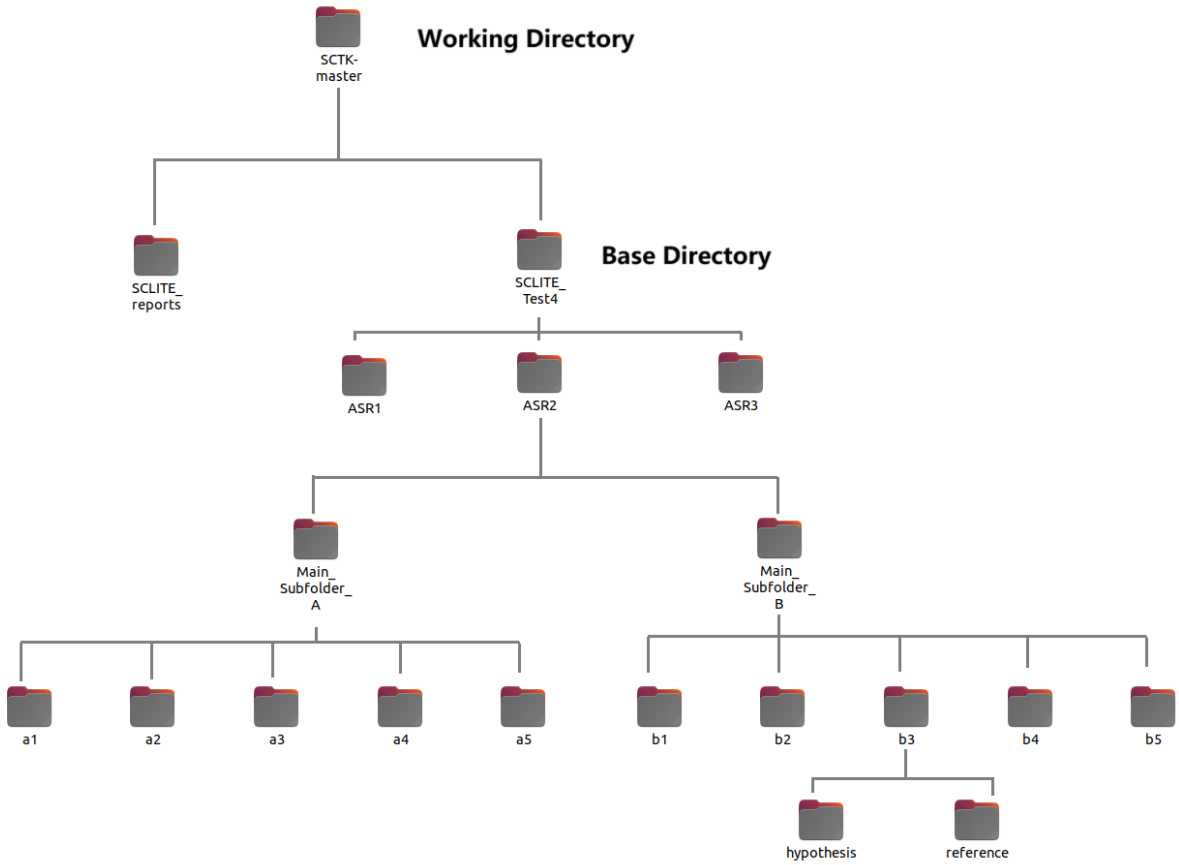

Figure 3.1: Working Directory Layout.

3.5 Metrics

We will now discuss the metrics used by SCTK, all the equations were obtained from[13].

3.5.1 Word Substitutions

Definition: Word Substitution Rate (WSR) measures the proportion of words in the ASR output that are substituted when compared to the reference transcription. A lower WSR indicates better ASR performance.

$$WSR = \frac{\text{Total Word Substitutions}}{\text{Total Reference Words}} \tag{3.1}$$

3.5.2 Word Deletions

Definition: Word Deletion Rate (WDR) quantifies the percentage of words that were present in the reference but deleted in the ASR output. A lower WDR suggests better ASR accuracy in retaining reference words.

$$WDR = \frac{\text{Total Word Deletions}}{\text{Total Reference Words}} \tag{3.2}$$

3.5.3 Word Insertions

Definition: Word Insertion Rate (WIR) calculates the percentage of additional words introduced in the ASR output that were not present in the reference transcription. A lower WIR is indicative of better ASR accuracy in avoiding unnecessary word insertions.

$$WIR = \frac{\text{Total Word Insertions}}{\text{Total Reference Words}} \tag{3.3}$$

3.5.4 Word Accuracy

Definition: Word Accuracy (WA) measures the overall correctness of the ASR output concerning the reference transcription. It is the complement of the Word Error Rate (WER). A higher WA signifies better ASR accuracy in word recognition.

$$WA = 1 - \frac{\text{Total Word Errors}}{\text{Total Reference Words}} \tag{3.4}$$

3.5.5 Generation of the Comparison Report

The primary goal of creating the "final_values.xlsx" file is to produce a structured and informative report that encapsulates key performance metrics for multiple ASR systems. These metrics are crucial for assessing the accuracy and quality of ASR transcriptions under different conditions and configurations.

- **Data Collection and Hierarchy**

  1. The process begins by traversing a hierarchical folder structure that represents different evaluation scenarios. This structure is organized into three levels: "Main Subfolder A," "Main Subfolder B," and "Whole Subfolder."

  2. Within each subfolder, multiple ASR systems (ASR1, ASR2, ASR3) have been evaluated.

- **Metric Calculation and Averaging**

  1. For each ASR system within each subfolder, a set of critical performance metrics is calculated. These metrics are computed by comparing the ASR-generated transcriptions with reference transcriptions (ground truth).

- **Averaging Metrics**

  1. To provide a concise summary of performance, the metrics are averaged across the different evaluation scenarios within each subfolder. This averaging process yields representative metric values for each ASR system and subfolder combination.

  2. This step ensures that the "final_values.xlsx" report offers a balanced view of ASR performance.

- **Data Structuring and Tabulation**

  1. The calculated metrics are structured into a well-organized table format. In this table:

     (a) Rows represent the different subfolders ("Main Subfolder A," "Main Subfolder B," and "Whole Subfolder").

     (b) Columns are multi-indexed, categorizing metrics based on ASR systems (ASR1, ASR2, ASR3).

2. This structured format simplifies the comparison of performance metrics across various ASR systems and subfolders.

- **Styling for Clarity**

1. The table is styled to enhance readability and visual appeal. Specifically, the alignment of column headers is set to the center, ensuring uniformity and clarity in the presentation of performance metrics.

- **Export to Excel**

1. The final, structured, and styled DataFrame is exported to an Excel file named "final_values.xlsx." This Excel report serves as a comprehensive reference for evaluating and comparing the performance of ASR systems under diverse conditions.

Importance of "final_values.xlsx"

The "final_values.xlsx" report is a pivotal component of the ASR evaluation process. It offers a detailed and structured overview of critical performance metrics for multiple ASR systems. By summarizing these metrics across different subfolders and ASR configurations, the report provides valuable insights into the accuracy and effectiveness of ASR transcriptions. Researchers, developers, and stakeholders can use this report to make informed decisions and improvements in ASR system development and deployment.

3.5.6 Challenges During Genersting Final$_v alues.xlsx$

The process of generating the "final_values.xlsx" report was not without its set of challenges. In this subsection, we outline the challenges encountered during this phase of the ASR evaluation process and the solutions devised to overcome them.

- **Sentence Handling**: One significant challenge arose from dealing with a collection of sentences within each ASR evaluation scenario to be able to pass it into SCLITE. To address this challenge, several steps were taken:

1. **Adding Missing Period**s: In some cases, the reference and hypothesis files lacked proper punctuation, particularly periods at the end of sentences. To ensure consistency and accurate evaluation, a function was implemented to open these files and insert periods at the end of sentences where missing. This step was crucial as it

ensured that sentences were correctly parsed during subsequent processing.

2. **Sentence Segmentation**: To facilitate further analysis and comparison, another function was created to split the text within files into individual sentences. This segmentation was based on common sentence-ending punctuation marks such as periods (.), exclamation marks (!), and question marks (?). Each sentence was placed on a separate line to prepare the data for processing with the SCLITE tool.

3. **Line Identification**: In preparation for the SCLITE tool, it was essential to assign a unique identifier to each line within the segmented text. This step ensured that the tool could effectively process and align sentences between the reference and hypothesis files, allowing for precise calculation of ASR metrics.

- **Aesthetic Excel Formatting**:Creating an Excel report that is both visually appealing and informative presented its own set of difficulties. To address these challenges:

  1. **Styling and Formatting**: Careful attention was paid to the aesthetics of the Excel report. While the primary focus was on content and accuracy, ensuring that the report was well-structured and visually pleasing required additional effort. Column alignment, cell borders, font styles, and spacing were adjusted to create a professional and readable report.

  2. **Multi-Indexing**: The report's multi-indexed table structure, which categorized metrics by ASR system and subfolder, posed challenges in terms of cell alignment and readability. Ensuring that column headers were centered and that the table was easy to interpret was a priority.

3.5.7 Drawbacks of the File Formatting Technique

While the file formatting technique used in this ASR evaluation process offers several advantages, it also presents certain drawbacks. In this subsection, we discuss the limitations and challenges associated with this method.

Handling Substitutions, Deletions, and Insertions

One significant challenge encountered during this ASR evaluation process is the accurate handling of word substitutions, deletions, and insertions. SCTK's SCLite tool, which is employed

for scoring, sometimes misidentifies these elements, leading to incorrect confusion pairs. In particular, when insertions or deletions occur in the transcriptions, SCLite may produce erroneous substitution errors, affecting the accuracy of the evaluation results.

Treatment of Punctuation as Errors

Another limitation of the file formatting technique pertains to the treatment of punctuation as errors by SCTK. When differences in punctuation and capitalization exist between the reference and hypothesis files, SCLite categorizes them as errors. This approach may not accurately reflect the true ASR performance, as punctuation and capitalization discrepancies are often non-critical for ASR applications. Consequently, the inclusion of such discrepancies in the error calculations can lead to less precise evaluation outcomes.

3.5.8 Approach 2: OnelinerDash

- **Period Insertion**: To ensure that sentences are correctly segmented and aligned, it was necessary to verify that periods were appropriately inserted at the end of sentences within both reference and hypothesis files. This addition of periods aimed to standardize sentence endings.
- **Sentence Segmentation**: To facilitate the comparison process, each file underwent sentence segmentation. Sentences were separated onto individual lines within the files, helping to establish a one-to-one correspondence between reference and hypothesis sentences. However, this segmentation introduces its own limitations.
- **Substitution of Missing Words**: In situations where words were missing from the hypothesis, these gaps were marked with hyphens (-) to indicate the omission of words. While this approach enabled sentence alignment, it transformed word deletions into substitutions, affecting the precision of the evaluation.
- **Punctuation Removal**: As a final step, all punctuation was removed from the files. While this addressed SCTK's treatment of punctuation as errors, it had the unintended consequence of treating words no longer present in the files as substitutions, even when they were originally deletions.
- **Punctuation Remova**: As a final step, all punctuation was removed from the files. While this addressed SCTK's treatment of punctuation as errors, it had the unintended

consequence of treating words no longer present in the files as substitutions, even when they were originally deletions.

- **Text Lowercasing**: Additionally, all text within the files was converted to lowercase. This step aimed to standardize the text, making it more amenable to comparison.

Pros

Improved Confusion Pair Accuracy: The application of these formatting steps significantly improved the accuracy of confusion pair detection. By inserting periods, segmenting sentences, and standardizing text, the ASR evaluation process became more precise and reliable.

Cons

Word Insertions and Deletions Not Detected: However, this method introduced a limitation in that SCLite might not detect word insertions and deletions accurately. The use of hyphens (-) to mark missing words can lead to SCLite categorizing them as word substitutions instead, affecting the precision of the evaluation in cases where insertions or deletions occur. This can result in less accurate ASR performance metrics, particularly in scenarios with frequent word insertions or deletions.

These considerations highlight the trade-off involved in the file formatting technique. While it enhances the accuracy of confusion pair identification, it can impact the correct detection of word insertions and deletions, thereby influencing the overall ASR evaluation results.

3.6 Additional Reports

3.6.1 Generating Reports

Now that we have done WER, we started on creating the detailed reports.

The function generate_all_reports serves to create detailed reports for the ASR evaluation. It navigates through the directory structure, processing each file within it. Here's a step-by-step explanation of the process:

- he function takes three parameters: **base_directory**, **hypothesisfoldername**, and **referencefoldername**, which respectively specify the root directory for the ASR evaluation,

the name of the folder containing hypothesis files, and the name of the folder containing reference files.

- It uses a series of nested loops to traverse the directory structure:
  1. The outermost loop iterates over the items (folders) in the **base_directory**.
  2. The next loop iterates over ASR systems within each folder.
  3. Another loop iterates over subfolders within each ASR system.
  4. Finally, it navigates to the folder containing hypothesis files.
- Within the innermost loop, for each file in the hypothesis folder, it calls the function **compreports**, which presumably generates detailed comparison reports between the reference and hypothesis files.

3.6.2 Organizing the reports

- Like **generate_all_reports**, this function also accepts **base_directory**, **report_directory**, and **hypothesisfoldername** as parameters.
- It iterates through the same directory structure but this time focuses on organizing and moving files to the **report_directory**.
- Within the innermost loop, it performs the following actions:
  1. Checks if the destination folder for the report exists in the report_directory. If not, it creates the folder.
  2. Renames the report files with different extensions (e.g., "dtl", "pra", "raw", "sys", "prf") and moves them to the corresponding report subfolder. This step likely ensures that the different types of detailed reports generated by **compreports** are correctly organized.

3.6.3 ASR Evaluation and Error Extraction

Both of these functions are part of a broader process for ASR evaluation and error extraction. After performing Word Error Rate (WER) calculations, detailed reports are generated for each ASR system. These reports likely contain information about errors, substitutions, deletions, insertions, etc., in a structured format.

This process allows for a detailed assessment of ASR system performance and provides valuable insights into where errors occur and how they manifest.

To be able to extract the errors table, and the most repeated errors table we do the following:

- **Initializing Data Structures**:At the beginning of the function, several lists and data frames are initialized to store different types of information. These include **reflist**, **hyplist**, **loclist**, **ASR**, **Freq**, **dfs**, and **errdf**.
    1. **reflist**, **hyplist**, and **loclist** are lists to store references, hypotheses, and file locations, respectively.
    2. **ASR** is a list to store ASR system names.
    3. **Freq** is a list to store frequencies (number of occurrences).
    4. **dfs** is a list to store Pandas DataFrames.
    5. **errdf** is a list to store DataFrames for errors.
- **Iterating Through Reports**:The function then enters a loop that iterates through the reports stored in the **report_directory**. It follows a directory structure where reports are organized by ASR systems, subfolders, in an order exactly similar to that of the base directory.
- **Processing Detailed Reports**: Within the inner loops, the function processes each detailed report (files with ".dtl" extension). The key steps involved are as follows:
    1. The function reads each line in the ".dtl" file and checks for lines that indicate the start of confusion pairs information.
    2. It identifies the number of confusion pairs mentioned in the report.
    3. It skips the initial lines that don't contain error information and activates the parsing of error details.
    4. For each error detail line, it extracts information about the wrong hypothesis (wrng_hyp), wrong reference (wrng_ref), and the location of the error within the report file (loc).
    5. This information is collected and added to the wrongsdf DataFrame.
- **Collecting Data for Analysis**:The function collects data from the grouped DataFrame

and stores it in various lists, including **ASR**, **reflist**, **hyplist**, **loclist**, and **Freq**. These lists store information about ASR systems, references, hypotheses, error locations, and frequencies, respectively.

- **Appending DataFrames**:The grouped DataFrame (grouped) is appended to the dfs list, which stores Pandas DataFrames. This list may be used for further analysis or reporting.
- **Returning Extracted Data**:The function returns multiple pieces of information, including ASR system names, reference and hypothesis lists, error locations, frequencies, grouped DataFrames, and individual DataFrames for errors (errdf). These data can be used for further analysis, reporting, or visualization.

  In your report, you can explain that this function plays a crucial role in extracting detailed error information from ASR evaluation reports. The collected data provides insights into the types of errors made by ASR systems and their frequency, which can be useful for understanding system performance and identifying areas for improvement.

3.7 Error Classification and Summary

In this section, we describe the process of error classification and the generation of error summary tables based on the ASR evaluation results.

3.7.1 Error Classification Function (`ErrorsinDF`)

The `ErrorsinDF` function is responsible for classifying errors within the ASR evaluation data, specifically identifying different types of errors based on linguistic and phonetic analysis. It also generates various error summary tables.

Input Parameters

- `df`: A list of DataFrames containing error information, where each DataFrame represents errors from a specific ASR system evaluation.
- `base_directory`: The base directory where the ASR evaluation data is stored.

Error Classification Process

The function performs the following error classification steps for each error entry in the input DataFrames:

1. **Noun-Article Mismatch**: It checks if the hypothesis word has a mismatch with an article in the reference. This type of error indicates issues with articles and nouns.
2. **Compound Word Split**: It identifies if the hypothesis word should be part of a compound word in the reference. This error type highlights problems with compound words.
3. **Hesitation Errors**: It detects common hesitation words and marks them as errors in the hypothesis.
4. **Phonetic Errors**: The function calculates the Cologne phonetic code for the reference and hypothesis words and compares them. If they differ, it indicates a phonetic error.
5. **Deletion Errors**: This type of error is identified when the hypothesis word is marked as a hyphen ("-"), indicating the deletion of a word.
6. **Partial Errors**: It checks if the hypothesis word partially matches the reference word or vice versa.
7. **Non-German Words**: It verifies if the hypothesis word is a valid German word using a dictionary. If not, it's marked as a non-German word error.

The errors are classified into one or more of the above types, and this classification is stored in the 'Error Type' column of the DataFrame.

Error Summary Tables

The function generates several error summary tables to provide insights into the distribution of error types:

- **Error Classification Table**: This table includes columns for the ASR system, reference word, hypothesis word, location, and error types. Each row represents a specific error entry.
- **Most Common Error Types Table**: This table identifies the most common error types for each ASR system and their frequencies. It helps identify prevalent error patterns.

- **Error Classification Frequency Table**: This table presents a summary of error types and their frequencies across all ASR systems.

Output

The function produces the following outputs:

- **Error Classification Table**: A detailed table containing error entries and their classifications.
- **Most Common Error Types Table**: A table showing the most common error types and their frequencies for each ASR system.
- **Error Classification Frequency Table**: A summary table displaying error types and their overall frequencies.

3.7.2 Usage and Report Generation

To use the `ErrorsinDF` function, provide the list of DataFrames representing ASR evaluation results and specify the base directory where the evaluation data is located. The function will perform error classification and generate the error summary tables.

These error summary tables are valuable for understanding the types and frequencies of errors in ASR evaluations, helping researchers and developers focus on areas for improvement.

3.8 Challenges in classifying errors

- **POS Tagging Accuracy**
    1. **Challenge**: POS (Part-of-Speech) tagging using Spacy or any NLP tool can have accuracy issues, especially when dealing with noisy or domain-specific text data. Errors in POS tagging can affect the accuracy of error classification.[15]
- **Phonetics Classification**
    1. **Challenge**: Phonetics-based error classification often requires complex models, such as deep learning models, to achieve high accuracy. Traditional phonetic codes like Cologne phonetics may not provide precise results in all cases.[16]
- **Installing the German Dictionary**

1. **Challenge**: Installing and configuring language dictionaries, including German, can sometimes be challenging due to dependencies and compatibility issues.

3.9 Issues with SCTK

While the Speech Recognition Scoring Toolkit (SCTK) is a valuable tool for ASR evaluation, it is important to acknowledge certain issues and challenges that users may encounter when working with SCTK. In this section, we highlight two main concerns related to SCTK.

3.9.1 Documentation Structure

One of the primary issues with SCTK is its documentation. While SCTK provides a comprehensive set of tools for ASR evaluation, the documentation is not always well-structured or user-friendly. Users may find it challenging to navigate and locate specific information or instructions, which can hinder the efficient utilization of the toolkit's capabilities. Improved documentation and organization could significantly enhance the user experience and make SCTK more accessible to a broader audience.

3.9.2 Alignment Mismatches

Another noteworthy concern when using SCTK is the potential for alignment mismatches, particularly in the context of confusion pairs. SCTK relies on alignment algorithms to identify errors, such as substitutions, deletions, and insertions. However, in some cases, the alignment process may not accurately match words or phrases in the reference and hypothesis, leading to incorrect confusion pairs and error calculations. These alignment-related mismatches can impact the reliability and precision of ASR evaluation results.

It is essential to be aware of these issues and to consider potential workarounds or alternative approaches when utilizing SCTK for ASR evaluation. Additionally, addressing these concerns through improved documentation and alignment algorithms could contribute to enhancing the overall usability and effectiveness of SCTK in ASR evaluation tasks.

3.9.3 Complexity of Configuration

SCTK provides a wide range of configuration options and parameters to fine-tune ASR evaluation processes. However, the complexity of these configurations can be overwhelming for users,

especially those who are new to the toolkit. Determining the appropriate settings for specific evaluation scenarios may require a significant amount of trial and error.

3.9.4 Limited Support for Non-English Languages

While SCTK is a valuable tool for English ASR evaluation, its support for non-English languages may be limited. Users working with ASR systems in languages other than English may face difficulties in obtaining accurate evaluations or may need to adapt and extend SCTK's functionality to handle different languages effectively.

3.9.5 Alignment Sensitivity

SCTK's alignment algorithms may be sensitive to variations in pronunciation, intonation, or speech quality. In cases where reference and hypothesis transcripts differ significantly in terms of prosody or accents, alignment accuracy may degrade, leading to less reliable ASR evaluation results.

Chapter 4

Results and Evalution

4.1 WER

		WER			Sentence with Errors			Word substitutions			Word deletions			Word insertions			Word Accuracy		
		ASR1	ASR2	ASR3	ASR1	ASR2	ASR3	ASR1	ASR2	ASR3	ASR1	ASR2	ASR3	ASR1	ASR2	ASR3	ASR1	ASR2	ASR3
main subfolder A	Subfolder 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	100	100
	Subfolder 2	16.7	50	50	100	100	100	11.1	27.8	27.8	0	0	0	5.6	22.2	22.2	88.9	72.2	72.2
	Subfolder 3	35.7	42.9	42.9	100	100	100	21.4	35.7	35.7	7.1	0	0	7.1	7.1	7.1	71.4	64.3	64.3
	Subfolder 4	6.7	26.7	26.7	100	100	100	6.7	20	20	0	0	0	0	6.7	6.7	93.3	80	80
	Subfolder 5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	100	100
main subfolder B	Subfolder 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	100	100
	Subfolder 2	6.7	6.7	6.7	100	100	100	6.7	6.7	6.7	0	0	0	0	0	0	93.3	93.3	93.3
	Subfolder 3	7.7	7.7	7.7	100	100	100	7.7	7.7	7.7	0	0	0	0	0	0	92.3	92.3	92.3
	Subfolder 4	23.1	15.4	15.4	100	100	100	23.1	7.7	7.7	0	0	0	0	7.7	7.7	76.9	92.3	92.3
	Subfolder 5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	100	100

	WER			Sentence with Errors			Word substitutions			Word deletions			Word insertions			Word Accuracy		
	ASR1	ASR2	ASR3	ASR1	ASR2	ASR3	ASR1	ASR2	ASR3	ASR1	ASR2	ASR3	ASR1	ASR2	ASR3	ASR1	ASR2	ASR3
main subfolder A	11.82	23.92	23.92	60	60	60	7.84	16.7	16.7	1.42	0	0	2.54	7.2	7.2	90.72	83.3	83.3
main subfolder B	7.5	5.96	5.96	60	60	60	7.5	4.42	4.42	0	0	0	0	1.54	1.54	92.5	95.58	95.58
Whole Subfolder	9.66	14.94	14.94	60	60	60	7.67	10.56	10.56	0.71	0	0	1.27	4.37	4.37	91.61	89.44	89.44

Figure 4.1: WER for ASR systems.

The table summarizes the Word Error Rate (WER), Sentence with Errors, Word Substitutions, Word Deletions, Word Insertions, and Word Accuracy for three ASR systems (ASR1, ASR2, ASR3) in two main subfolders (A and B), along with a combined total for the entire subfolder. Here's a breakdown of the results:

4.1.1 Main Subfolder A

- **WER**: ASR1 has a WER of 11.82%, ASR2 has 23.92%, and ASR3 has 23.92%.
- **Sentence with Errors**: ASR1, ASR2, and ASR3 have 60% each.
- **Word Substitutions**: ASR1, ASR2, and ASR3 have 60% each.
- **Word Deletions**: ASR1, ASR2, and ASR3 have 60% each.

- **Word Insertions**: ASR1, ASR2, and ASR3 have 60% each.
- **Word Accuracy**: ASR1 has 90.72%, ASR2 has 83.3%, and ASR3 has 83.3%.

4.1.2 Main Subfolder B

- **WER**:ASR1 has a WER of 7.5%, ASR2 has 5.96%, and ASR3 has 5.96
- **Sentence with Errors**: ASR1, ASR2, and ASR3 have 60% each.
- **Word Substitutions**: ASR1, ASR2, and ASR3 have 60% each.
- **Word Deletions**: ASR1, ASR2, and ASR3 have 60% each.
- **Word Insertions**: ASR1, ASR2, and ASR3 have 60% each.
- **Word Accuracy**: ASR1 has 92.5%, ASR2 has 95.58%, and ASR3 has 95.58%.

4.1.3 Whole Subfolder:

- **WER**: ASR1 has a WER of 9.66
- **Sentence with Errors**: ASR1, ASR2, and ASR3 have 60
- **Word Substitutions**: ASR1, ASR2, and ASR3 have 60
- **Word Deletions**: ASR1, ASR2, and ASR3 have 60
- **Word Insertions**: ASR1, ASR2, and ASR3 have 60
- **Word Accuracy**: ASR1 has 91.61

Overall, the results indicate variations in Word Error Rate, sentence errors, and word-level errors among the ASR systems and subfolders. ASR2 and ASR3 generally perform worse in terms of WER and word accuracy compared to ASR1, with ASR3 consistently having the highest error rates. However, ASR2 and ASR3 show better performance in word accuracy in some cases. These results suggest the need for further analysis and potential improvements in the ASR systems.

4.2 Error Extraction Table

ASR System	Reference Word	Hypothesis Word	Location	Error Frequency in (ASR System)
ASR2	zur	-	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	1
ASR2	n	ein	Main_Subfolder_A/a4/af21986c-f0e0-4a72-b8a2-41937c45b774_00000_0.0-7.11.txt.dtl	1
ASR2	mal	einmal	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	1
ASR2	gehts	es	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	1
ASR2	was	etwas	Main_Subfolder_A/a4/af21986c-f0e0-4a72-b8a2-41937c45b774_00000_0.0-7.11.txt.dtl	1
ASR2	gehts	geht	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	1
ASR2	vorlesung	in	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	1
ASR2	willkommen	kommen	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtlMain_Subfolder_A/a3/fc91ec9f-42	4
ASR2	vorlesungen	vorlesung	Main_Subfolder_B/b2/298c9097-2a46-4f65-997d-6da071f307b2_00000_0.0-6.9.txt.dtlMain_Subfolder_B/b3/00b9bc7f-6e	2
ASR2	werd	werde	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	1
ASR2	viren	wieheren	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	1
ASR2	zellen	zählen	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	1
ASR3	zur	-	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	1
ASR3	n	ein	Main_Subfolder_A/a4/af21986c-f0e0-4a72-b8a2-41937c45b774_00000_0.0-7.11.txt.dtl	1
ASR3	mal	einmal	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	1
ASR3	gehts	es	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	1
ASR3	was	etwas	Main_Subfolder_A/a4/af21986c-f0e0-4a72-b8a2-41937c45b774_00000_0.0-7.11.txt.dtl	1
ASR3	gehts	geht	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	1
ASR3	vorlesung	in	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	1
ASR3	heutigen	ja	Main_Subfolder_A/a5/b0fcb8d6-9749-46a5-abbf-da8e02ff30c7_00000_0.0-6.75.txt.dtl	1
ASR3	willkommen	kommen	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtlMain_Subfolder_A/a3/fc91ec9f-42	4
ASR3	vorlesungen	vorlesung	Main_Subfolder_B/b2/298c9097-2a46-4f65-997d-6da071f307b2_00000_0.0-6.9.txt.dtlMain_Subfolder_B/b3/00b9bc7f-6e	2
ASR3	werd	werde	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	1
ASR3	viren	wieheren	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	1
ASR3	zellen	zählen	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	1

Figure 4.2: Errors Table.

4.3 Most Frequent Errors Table

ASR System	Reference Word	Hypothesis Word	Location	Error Frequency in (ASR System)
ASR2	willkommen	kommen	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtlMain_Subfolder_A/a3/fc91ec9f-42	4
ASR2	vorlesungen	vorlesung	Main_Subfolder_B/b2/298c9097-2a46-4f65-997d-6da071f307b2_00000_0.0-6.9.txt.dtlMain_Subfolder_B/b3/00b9bc7f-6e	2
ASR3	willkommen	kommen	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtlMain_Subfolder_A/a3/fc91ec9f-42	4
ASR3	vorlesungen	vorlesung	Main_Subfolder_B/b2/298c9097-2a46-4f65-997d-6da071f307b2_00000_0.0-6.9.txt.dtlMain_Subfolder_B/b3/00b9bc7f-6e	2

Figure 4.3: Most Frequent Errors.

4.4 Error Classification

4.4.1 Error Classification Frequency

Error Type	Frequency
Phonetic Error	37
Partial Error	26
Deletion Error	4
noun-article-mismatch	6
compound-word-split	3
Hesitation	1

Figure 4.4: Error Frequency per type.

4.4.2 Error Classification

ASR	Hypothesis word	Reference word	Location	Error Type
ASR2	es	gehts	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	['Phonetic Error', 'Not German word']
ASR2	einmal	mal	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	['Phonetic Error', 'Partial Error', 'Not German word']
ASR2	in	vorlesung	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	['Phonetic Error', 'Not German word']
ASR2	werde	werd	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	['Partial Error', 'Not German word']
ASR2	kommen	willkommen	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	['Phonetic Error', 'Partial Error', 'Not German word']
ASR2	geht	gehts	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	['Phonetic Error', 'Partial Error', 'Not German word']
ASR2	wieheren	viren	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	['Phonetic Error']
ASR2	kommen	willkommen	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	['compound-word-split', 'Phonetic Error', 'Partial Error', 'Not German word']
ASR2	zählen	zellen	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	['Phonetic Error', 'Not German word']
ASR2	-	zur	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	['Deletion Error']
ASR2	ein	n	Main_Subfolder_A/a4/af21986c-f0e0-4a72-b8a2-41937c45b774_00000_0.0-7.11.txt.dtl	['Phonetic Error', 'Partial Error', 'Not German word']
ASR2	etwas	was	Main_Subfolder_A/a4/af21986c-f0e0-4a72-b8a2-41937c45b774_00000_0.0-7.11.txt.dtl	['Phonetic Error', 'Partial Error', 'Not German word']
ASR2	kommen	willkommen	Main_Subfolder_A/a4/af21986c-f0e0-4a72-b8a2-41937c45b774_00000_0.0-7.11.txt.dtl	['Phonetic Error', 'Partial Error', 'Not German word']
ASR2	vorlesung	vorlesungen	Main_Subfolder_B/b2/298c9097-2a46-4f65-997d-6da071f307b2_00000_0.0-6.9.txt.dtl	['noun-article-mismatch', 'Phonetic Error', 'Partial Error']
ASR2	vorlesung	vorlesungen	Main_Subfolder_B/b3/00b9bc7f-6e6f-4346-98b6-2e26d70217ed_00000_0.0-6.96.txt.dtl	['noun-article-mismatch', 'Phonetic Error', 'Partial Error']
ASR2	kommen	willkommen	Main_Subfolder_B/b4/afc11203-35ef-4c7c-8571-4703dd67f56f_00000_0.0-6.95.txt.dtl	['Phonetic Error', 'Partial Error', 'Not German word']
ASR3	es	gehts	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	['Phonetic Error', 'Not German word']
ASR3	einmal	mal	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	['Phonetic Error', 'Partial Error', 'Not German word']
ASR3	in	vorlesung	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	['Phonetic Error', 'Not German word']
ASR3	werde	werd	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	['Partial Error', 'Not German word']
ASR3	kommen	willkommen	Main_Subfolder_A/a2/2c161a6f-3678-4aaf-95c2-daa03f09abc8_00000_0.0-7.08.txt.dtl	['Phonetic Error', 'Partial Error', 'Not German word']
ASR3	geht	gehts	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	['Phonetic Error', 'Partial Error', 'Not German word']
ASR3	wieheren	viren	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	['Phonetic Error']
ASR3	kommen	willkommen	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	['compound-word-split', 'Phonetic Error', 'Partial Error', 'Not German word']
ASR3	zählen	zellen	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	['Phonetic Error', 'Not German word']
ASR3	-	zur	Main_Subfolder_A/a3/fc91ec9f-4248-4eac-bf9f-aedd4f559944_00000_0.0-6.72.txt.dtl	['Deletion Error']
ASR3	ein	n	Main_Subfolder_A/a4/af21986c-f0e0-4a72-b8a2-41937c45b774_00000_0.0-7.11.txt.dtl	['Phonetic Error', 'Partial Error', 'Not German word']
ASR3	etwas	was	Main_Subfolder_A/a4/af21986c-f0e0-4a72-b8a2-41937c45b774_00000_0.0-7.11.txt.dtl	['Phonetic Error', 'Partial Error', 'Not German word']
ASR3	kommen	willkommen	Main_Subfolder_A/a4/af21986c-f0e0-4a72-b8a2-41937c45b774_00000_0.0-7.11.txt.dtl	['Phonetic Error', 'Partial Error', 'Not German word']
ASR3	ja	heutigen	Main_Subfolder_A/a5/b0fcb8d6-9749-46a5-abbf-da8e02ff30c7_00000_0.0-6.75.txt.dtl	['Hesitation', 'Phonetic Error', 'Not German word']
ASR3	vorlesung	vorlesungen	Main_Subfolder_B/b2/298c9097-2a46-4f65-997d-6da071f307b2_00000_0.0-6.9.txt.dtl	['noun-article-mismatch', 'Phonetic Error', 'Partial Error']
ASR3	vorlesung	vorlesungen	Main_Subfolder_B/b3/00b9bc7f-6e6f-4346-98b6-2e26d70217ed_00000_0.0-6.96.txt.dtl	['noun-article-mismatch', 'Phonetic Error', 'Partial Error']
ASR3	kommen	willkommen	Main_Subfolder_B/b4/afc11203-35ef-4c7c-8571-4703dd67f56f_00000_0.0-6.95.txt.dtl	['compound-word-split', 'Phonetic Error', 'Partial Error', 'Not German word']

Figure 4.5: Error Classification Table.

Chapter 5

Conclusion

In this study, we conducted an error analysis of three ASR systems (ASR1, ASR2, ASR3) within two main subfolders (A and B) and provided a comprehensive overview of Word Error Rate (WER), sentence errors, word substitutions, word deletions, word insertions, and word accuracy. The results revealed variations in the performance of ASR systems across different metrics and subfolders.

While our analysis provided valuable insights into the ASR system's performance, it's important to note that the error classification using basic techniques had limitations and did not yield the best results. Many factors can contribute to ASR errors, including acoustic conditions, language models, and data quality.

To achieve more accurate error classification and improved ASR system performance, future work should focus on more advanced techniques. Training models specifically tailored to the ASR systems and carefully preparing data can lead to better results. Additionally, incorporating deep learning models for error classification and leveraging phonetics-based approaches might provide more accurate insights into the nature of errors.

In conclusion, our analysis sheds light on the challenges and opportunities in ASR error analysis. While basic error classification techniques have their limitations, further research and advancements in ASR technology hold the promise of achieving higher accuracy and better performance in the future. Feel free to use this LaTeX code as part of your report's conclusion.

References

[1]El Hannani, A., Errattahi, R., Salmam, F.Z. et al. Evaluation of the effectiveness and efficiency of state-of-the-art features and models for automatic speech recognition error detection. J Big Data 8, 5 (2021). https://doi.org/10.1186/s40537-020-00391-w

[2]G. Hinton et al., "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups," in IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 82-97, Nov. 2012, doi: 10.1109/MSP.2012.2205597.

[3]Woodland, P. C., Young, S. J., & Gales, M. J. F. (1997). Evaluation measures for speech recognition. Journal of Information Technology, 12(5), 371-380.

[4]Jurafsky, Daniel & Martin, James. (2008). Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.

[5]Kernighan, M. D., & Church, K. W. (1990). Toward robust automatic speech recognition for large unrestricted corpora. Proceedings of the conference on Speech and Natural Language (ACL), 76-83.

[6]Levow, G. A. (2006). The third international Chinese language processing bakeoff: Word segmentation and named entity recognition. Proceedings of the fifth SIGHAN workshop on Chinese language processing (ACL), 108-117.

[7]Snover, M., Dorr, B., Schwartz, R., Micciulla, L., & Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. Proceedings of Association for Machine Translation in the Americas, 223-231.

[8]Powers, D. M. W. (2011). Evaluation: From precision, recall and F-measure to ROC, informedness, markedness & correlation. Journal of Machine Learning Technologies, 2(1), 37-63.

[9]Jelinek, F., & Mercer, R. L. (1980). Interpolated estimation of Markov source parameters from sparse data. Proceedings of the workshop on Pattern recognition in practice (Springer), 381-397.

[10] Allen, J. B., & Berkley, D. A. (1979). Image method for efficiently simulating small-room acoustics. The Journal of the Acoustical Society of America, 65(4), 943-950.

[11] Chan, W., Jaitly, N., Le, Q. V., & Vinyals, O. (2016). Listen, attend and spell. Proceedings of the International Conference on Machine Learning (ICML), 1884-1893.

[12]Hu, Z., Li, J., Zhang, S., Zong, C., & Chng, E. S. (2019). Deep Speaker Embeddings for Diarization and Multi-Speaker Speech Recognition. Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP), 6306-6310.

[13]"The HTK Book." (Chapter 11: Evaluation). Cambridge University Engineering Department. URL: http://htk.eng.cam.ac.uk/docs/docs.shtml

[14]SCTK (Speech Recognition Scoring Toolkit). (2021, October 28). GitHub. https://github.com/usnistgov/SCTK

[15]Toutanova, K., Klein, D., Manning, C. D., & Singer, Y. (2003). Part-of-Speech Tagging: From 0.97 to 1.00 and Beyond. In Proceedings of the 2003 Conference of the North American

Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1 (NAACL'03), (pp. 1-8).

[16]Rabiner, L. R., & Juang, B. H. (1993). Fundamentals of Speech Recognition. Prentice Hall.

[17]Fabien, M. Speech Recognition. Maël Fabien.
https://maelfabien.github.io/machinelearning/speech_reco/

[18]Everything Is Music. (n.d.). What Is Automatic Speech Recognition? Everything Is Music.
https://everythingismusic.io/what-is-automatic-speech-recognition/

[19]Juang, B. & Rabiner, Lawrence. (2005). Automatic Speech Recognition - A Brief History of the Technology Development.

[20]Lawrence R. Rabiner (February 1989). "A tutorial on Hidden Markov Models and selected applications in speech recognition" (PDF). Proceedings of the IEEE. 77 (2): 257–286. CiteSeerX 10.1.1.381.3454. doi:10.1109/5.18626. S2CID 13618539.

[21]Povey, Daniel & Burget, Lukas & Agarwal, Mohit & Akyazi, Pinar & Kai, Feng & Ghoshal, Arnab & Glembek, Ondrej & Goel, Nagendra & Karafiát, Martin & Rastrow, Ariya & Rose, Richard & Schwarz, Petr & Thomas, Samuel. (2010). Subspace Gaussian Mixture Models for speech recognition. ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings. 4330-4333. 10.1109/ICASSP.2010.5495662.

[22]Srikanth. (2023, September 3). Pandas Library Explained. Medium.
https://srikanthwritings.medium.com/pandas-library-explained-7e13ad40bcc2.

[23]GeeksforGeeks. (n.d.). Python Subprocess Module to Execute Programs Written in Different Languages. GeeksforGeeks. https://www.geeksforgeeks.org/python-subprocess-module-to-

execute-programs-written-in-different-languages/

[24]Python Software Foundation. (n.d.). re — Regular expression operations. Python Documentation. https://docs.python.org/3/library/re.html

[25]Python Software Foundation. (n.d.). os — Miscellaneous operating system interfaces. Python Documentation. https://docs.python.org/3/library/os.html

[26]Openpyxl. (n.d.). Openpyxl Documentation. https://openpyxl.readthedocs.io/en/stable/

[27]Python Software Foundation. (n.d.). shutil — High-level file operations. Python Documentation. https://docs.python.org/3.1/library/shutil.html

[28]Explosion AI. (n.d.). SpaCy Models for German. SpaCy. https://spacy.io/models/de

[29]Python Package Index. (n.d.). PyEnchant. PyPI. https://pypi.org/project/pyenchant/

[30]DigitalOcean. (n.d.). Python Counter - Python Collections Counter. DigitalOcean Community Tutorials. https://www.digitalocean.com/community/tutorials/python-counter-python-collections-counter

[31]Alzubaidi, L., Zhang, J., Humaidi, A.J. et al. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. J Big Data 8, 53 (2021). https://doi.org/10.1186/s40537-021-00444-8

[32]Chiu, Chung-Cheng & Sainath, Tara & Wu, Yonghui & Prabhavalkar, Rohit & Nguyen, Patrick & Chen, Zhifeng & Kannan, Anjuli & Weiss, Ron & Rao, Kanishka & Gonina, Katya & Jaitly, Navdeep & Li, Bo & Chorowski, Jan & Bacchiani, Michiel. (2017). State-of-the-art Speech Recognition With Sequence-to-Sequence Models.

[33]Meinedo, H. D. S. (2008). Audio Pre-processing and Speech Recognition for Broadcast News (Doctoral dissertation). Universidade Tecnica de Lisboa.