IWSLT 2024 Indic Track system description paper: Speech-to-Text Translation from English to multiple Low-Resource Indian Languages

Deepanjali Singh, Ayush Anand, Abhyuday Chaturvedi and **Niyati Baliyan***
Department of Computer Engineering
National Institute of Technology Kurukshetra
Haryana, India, 136118
⋆niyatibaliyan@nitkkr.ac.in

Abstract

Multi-Language Speech-to-Text Translation (ST) plays a pivotal role in bridging linguistic barriers by converting spoken language into written text across different languages. This project aims to develop a robust ST model tailored for low-resource Indian languages, specifically targeting the Indo-Aryan and Dravidian language families. The dataset used consists of speeches from conferences and TED Talks, along with their corresponding transcriptions in English (source language) and translations in Hindi, Bengali, and Tamil (target languages). By tackling the lack of data and disparities in attention within low-resource languages, the paper strives to create an efficient ST system capable of real-world deployment. Additionally, existing resources in related languages are leveraged and word-level translation resources are explored to enhance translation accuracy.

1 Introduction

Multi-Language Speech-to-Text Translation (ST) is indispensable for facilitating communication across diverse linguistic contexts. While recent advancements have shown remarkable progress, many dialects and low-resource languages still lack sufficient parallel data for effective supervised learning. Creative approaches are essential to overcome this challenge, such as leveraging resources from related languages or utilizing word-level translation resources and raw audio. This work aims to address these gaps by developing an End-to-End (E2E) or Cascaded ST model for low-resource Indian languages, including Hindi, Bengali, and Tamil.

2 Motivation

The scarcity of translators proficient in multiple languages, especially in low-resource settings, highlights the urgent need for ST systems supporting multiple languages. In regions like India, characterized by a multitude of languages, the development of dedicated models for Indian languages is essential for effective communication. This task aims to advance ST technology for a wide range of languages. Our ultimate goal is to foster inclusively and accessibility through the creation of robust ST models. This research work is fueled by a strong commitment to address significant challenges in speech translation, with a particular focus on languages spoken in India. In modern interconnected society, the capacity to communicate across various languages is crucial. However, the shortage of translators who can handle multiple languages in resource-constrained areas, presents a major obstacle.

3 Related Work

Prior research in ST has primarily focused on high-resource languages, leaving many dialects and low-resource languages underserved. The lack of parallel data poses a significant challenge in training supervised learning models for these languages. However, recent efforts have demonstrated the effectiveness of leveraging existing resources from related languages and employing innovative approaches to enhance translation accuracy[5]. The 20^{th} International Conference on Spoken Language Translation (IWSLT) organized shared tasks targeting nine scientific challenges in spoken language translation (SLT). These tasks covered a wide spectrum. This encompasses simultaneous and offline translation, automatic subtitling and dubbing, speech-to-speech translation, multilingual translation, translation of dialects and low-resource languages, and formality control. The conference witnessed substantial interest with a total of 38 submissions from 31 teams, evenly distributed between academia and industry [1]. The focal point of the 2023 IWSLT Evaluation Campaign was offline SLT, which involved translating audio speech from one language to text in another language without time constraints. It com-

Proceedings of the 21st International Conference on Spoken Language Translation (IWSLT 2024), pages 359–364

prised three sub-tasks for translating English into German, Japanese, and Chinese. Participants were given the flexibility to utilize either cascade architectures, which combine automatic speech recognition (ASR) and machine translation (MT) systems, or E2E approaches that directly translate input speech [1]. Principal objectives were twofold: firstly, to gauge the performance disparity between cascade and end-to-end systems, and secondly, to evaluate SLT technology's competence in handling intricate scenarios like simultaneous overlapping or concurrent speakers. The introduction of new test sets, encompassing ACL presentations and press conferences/interviews, aimed at a comprehensive assessment of system efficacy [1]. Training data conditions spanned from constrained to unconstrained, offering varying levels of access to training resources. Development data encompassed TED talks, ACL presentations, and interviews from the European Parliament Multimedia Centre. System evaluations were conducted employing BLEU and COMET metrics, supplemented by human assessment of the top-performing entries [4]. Ten teams partook in the offline task, collectively submitting 37 runs. A plethora of techniques were employed across these submissions, including cascade and direct models, leveraging large language models, multimodal representations, data augmentation, ensemble methods, and advanced training strategies. Evaluation criteria emphasized the attainment of high translation quality across diverse language pairs and challenging scenarios [1].

4 System Overview

4.1 Key Components of the App

4.1.1 Audio Processor and Transcription Module

- Responsible for cleaning audio file
- Uses ResembleAI for Noise reduction, Restoring distortion, enhancing speech bandwidth
- Uses OpenAI's Whisper [1] model for transcription [3].

4.1.2 Input Module

- Responsible for receiving audio files
- Validates and preprocesses the input data for further processing.

[1]https://openai.com/index/whisper/

4.1.3 Translation Module - English to Hindi

- Integrates the Helsinki model for achieving translation of the transcribed text
- Fine tuning of pretrained translator model to enhance the result quality [2].

4.1.4 Translation Module - English to Tamil

- Integrates Facebook's mBART model for achieving translation of the transcribed text
- Fine tuning of pretrained translator model to enhance the result quality [2].

4.1.5 Translation Module - English to Bengali

- Integrates Facebook's mBART model for achieving translation of the transcribed text
- Fine tuning of pretrained translator model to enhance the result quality [2].

4.1.6 Output Module

- Performs syntax correction and eliminates any detectable hallucination by the model
- Delivers the translated text to users in their desired format, such as text files

4.2 SacreBLEU scores

Table 1 contains self assessment `SacreBLEU` scores of different model tested. Models selected are:

Whisper and Helsenki [3] for English-to-Hindi.

Whisper and mBART for English-to-Tamil

Whisper and mBART for English-to-Bengali

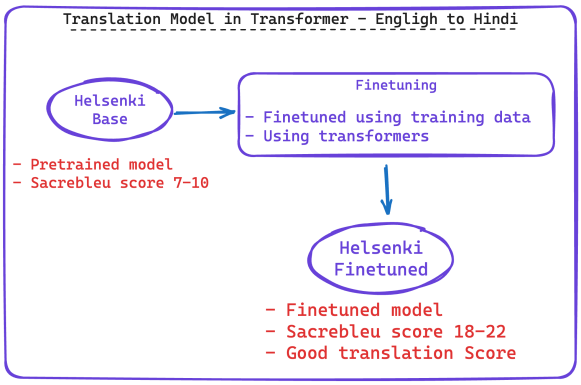
4.3 Implementation Pillars

4.3.1 translate.py

- It imports various functions from different modules to perform tasks like transcribing audio, translating text, breaking lines, saving files, and post-processing text.
- It sets up the starting time to measure how long the code takes to execute.

Language Pair	Model Used	Score
en-hi	whisper/helsenki	24.21
en-bn	whisper/helsenki	14.18
en-bn	whisper/mBART	16.18
en-ta	whisper/helsenki	7.1
en-ta	whisper/mBART	10.79

Table 1: SacreBLEU Scores



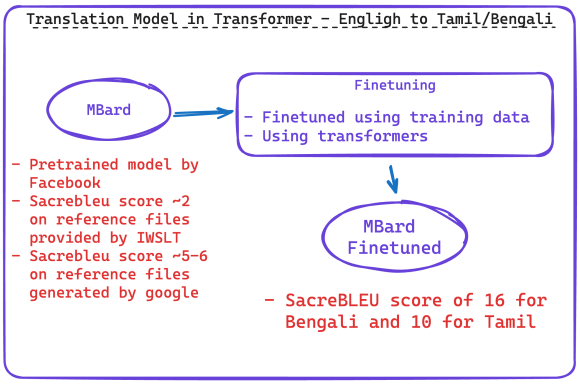


Figure 1: (a):English-to-Hindi by Helsenki (b):English-to-Tamil/Bengali by mBART

- It transcribes audio files present in the specified folder in English text, optionally based on a YAML file that specifies line changes.
- It translates the transcribed English text into Hindi and saves it.
- It translates the transcribed English text into Tamil and Bengali using Facebook's mBART translation service, then saves them.
- Finally, it prints the time it takes for execution.

4.3.2 transcriber.py

- It imports necessary libraries/modules such as `os`, `yaml`, `pydub`, `logging`, and a pipeline from the `transformers` library.
- It sets the logging level for the `transformers` library to `ERROR` to suppress unnecessary output, except for any errors.
- It defines a function `transcribe_audio(filePath)` that takes the path of an audio file, uses the OpenAI Whisper model via the Hugging Face Transformers library to transcribe the audio, and returns the transcribed text [3].
- It defines another function `transcriber(audios_dir, yaml_file_path)` that takes the directory containing audio files and the path to a YAML file as inputs. This function loads audio segments from the YAML file, iterates through audio files in the specified directory, extracts segments based on the information in the YAML file, transcribes each segment using the `transcribe_audio` function, and returns a list of transcribed texts.
- Enables selective use of YAML-based chunks to force line changes in the result.

4.3.3 translator.py

- Figure 1(a) shows English-to-Tamil translation workflow of `translator.py` module, it imports necessary functions from the `transformers` library to utilize pretrained translation models.
- It defines a function called `translatorModel`, which takes two arguments: `lines`, representing the text to be translated, and `target`, indicating the target language for translation.
- Inside the function, it loads a pretrained translation model and tokenizer specific to the target language using the Helsinki-NLP library.
- It iterates through each line in the input lines.
- For each non-empty line, it tokenizes the text using the tokenizer, prepares the input for the model, generates the translation, and decodes the translated output.
- It appends the translated text to a result array.
- It returns an array of translated text lines.

4.3.4 fbtranslate.py

- Figure 1(b) shows English-to-Tamil and English-to-Bengali translation workflow of `fbtranslate.py` module ,it defines a function called `fbtranslate(lines)`, which takes a list of input text lines as its argument.
- Inside the function, it initializes a translation pipeline using the `pipeline` function. This pipeline is configured to use the model named `"facebook/mbart-large-50-many-to-many-mmt"` for translation tasks.

- It initializes an empty list named `result` to store the translated text lines.
- It iterates through each line in the input lines.
- For each non-empty line, it translates the text from English (source language: `"en_XX"`) to Tamil (target language: `"ta_IN"`) using the translation pipeline.
- It extracts the translated text from the output of the translation pipeline and appends it to the `result` list.
- Finally, it returns the list containing the translated text lines.

4.4 Fine Tuning Logic Overview

4.4.1 Importing Libraries

We import necessary libraries including `Dataset`, `DatasetDict`, `AutoTokenizer`, `AutoModelForSeq2SeqLM`, `DataCollatorForSeq2Seq`, `Seq2SeqTrainingArguments`, `Seq2SeqTrainer`, and `load_metric`.

4.4.2 Loading Metric

We load the `SacreBLEU` metric for evaluating translation quality.

4.4.3 Model Checkpoint and File Paths

- The pretrained model checkpoint `"Helsinki-NLP/opus-mt-en-hi"` is specified.
- Paths for English (`train.en`) and Hindi (`train.hi`) training data files are defined.

4.4.4 Reading Data

English and Hindi sentences are read from their respective files.

4.4.5 Creating Dataset

- The English and Hindi sentence pairs are organized into a dictionary format.
- A `Dataset` object is created from this dictionary.

4.4.6 Creating DatasetDict

A `DatasetDict` object is created containing the train dataset.

4.4.7 Initializing Tokenizer

The tokenizer is instantiated using the specified model checkpoint.

4.4.8 Defining Preprocessing Function

- A function `preprocess_function` is defined to prepare input data for training.
- Inputs and targets are tokenized, and input IDs and labels are generated.

4.4.9 Mapping Preprocessing Function

The `preprocess_function` is applied to the train dataset using the `map` function.

4.4.10 Model Initialization

The pretrained model for sequence-to-sequence learning is instantiated.

4.4.11 Defining Training Arguments

- Evaluation strategy, learning rate, batch size, etc., are defined using `Seq2SeqTrainingArguments`.
- A data aggregator is developed for sequence-to-sequence assignments.

4.4.12 Defining Post-processing Function

A post-processing function for predictions and computing metrics is defined.

4.4.13 Training Configuration

A `Seq2SeqTrainer` is initialized with the model, training arguments, datasets, data collator, tokenizer, and compute metrics function.

4.4.14 Training Loop

The `train` method of the trainer object is called to initiate training.

4.4.15 Saving the Model

After successful execution of above logic, we have a fine-tuned model saved as a `.safetensors` file.

5 Workflow

Figure 2 shows basic workflow of the application. At first, there is Input Processing, where users upload audio files or provide input through supported channels. This serves as the gateway for input data, ensuring its integrity and validity. The Input Module undertakes the crucial task of verifying and preprocessing the audio data, preparing it for subsequent processing stages by addressing any inconsistencies.

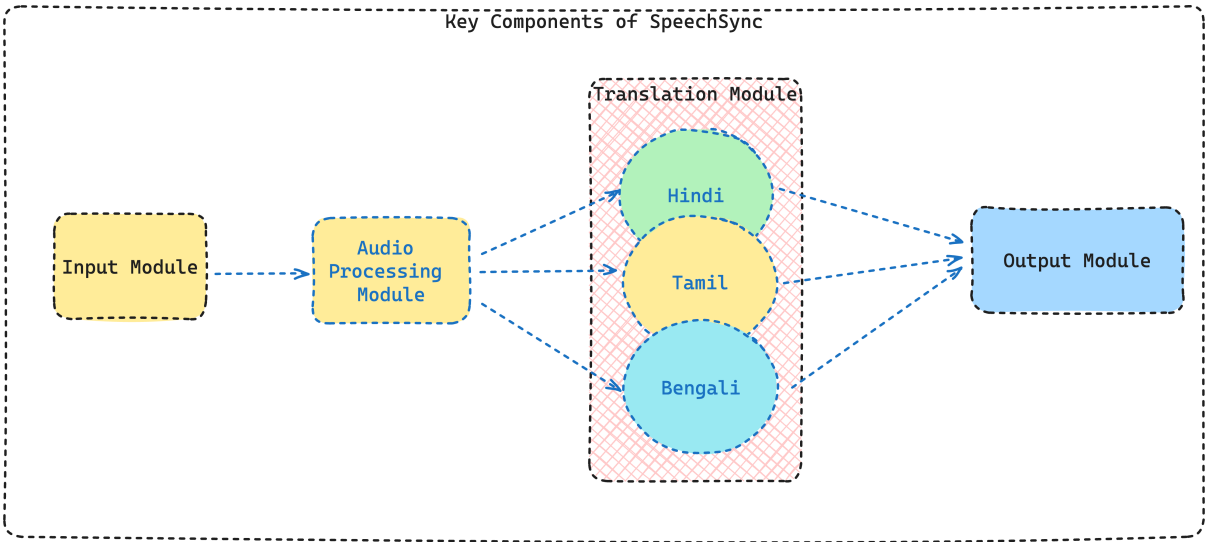


Figure 2: Basic flow of SpeechSync System

Next, Transcription and Translation enable transformation of audio content into translated text. The Transcription Module uses advanced algorithms to convert audio files into text, maintaining high accuracy and reliability. Meanwhile, the Translation Module leverages Helsinki model to translate transcribed text into specific languages, ensuring linguistic precision and preserving contextual nuances to facilitate communication across languages.

Once the transcription and translation processes are complete, the Output Delivery stage takes over, presenting the translated text to users through the Output Module. This enables seamless access and utilization of the translated content, offering users the flexibility to download the text or integrate it directly into their workflows. By providing a user-friendly interface and facilitating easy dissemination of translated content, the application empowers users to overcome language barriers and engage in effective cross-cultural communication.

5.1 Environment Settings

5.1.1 Prerequisites

- Python 3.11
- `ffmpeg` (command-line tool)

5.1.2 Installing ffmpeg

- **Ubuntu:** `sudo apt update && sudo apt install ffmpeg`
- **MacOS:** `brew install ffmpeg`
- **Windows:** `choco install ffmpeg`

5.1.3 App Installation

1. **Clone the repository**[2]

```
git clone git@github.com:
ayushannand/SpeechSync.git
```

2. **Create a virtual environment**

```
python3 -m venv env
```

3. **Activate the virtual environment**

```
source env/bin/activate
```

5.1.4 Install Rust

```
curl --proto '=https' --tlsv1.2
-sSf https://sh.rustup.rs | sh
```

6 Baseline vs. Results

The baseline `SacreBLEU` scores is provided by INDIC Track. For each language pair we have a different baselines.

6.0.1 English-to-Hindi

For language pair `en-hi` baseline is `5.23` and we get a score of 24.

6.0.2 English-to-Bengali

For language pair `en-bn` baseline is `5.86` and we get a score of 16.

[2] https://github.com/ayushannand/SpeechSync

6.0.3 English-to-Tamil

For language pair `en-ta` baseline is `1.9` and we get a score of `10`.

7 Limitations

While we acknowledge the significant challenges ahead, such as the shortage of multilingual individuals and insufficient data for certain languages, we are determined to find innovative solutions. Our input module currently supports only one language, so if the audio file contains multiple languages, the application ignores languages other than primary language. Currently, the other limitation is the time taken by the models to produce output. We may try out various optimisations and configurations to achieve faster results. For language pair - English to Bengali, we are barely crossing the baseline, so our primary goal is to achieve better score for Bengali language.

8 Conclusion

In summary, our key contributions lie in rigorous experimentation conducted to identify effective models for speech translation. We perform extensive preprocessing of data performed to ensure quality and suitability for training. The proposed solution establishes a robust pipeline including code development and workflow setup. The training and experimentation is focused on one language for an in-depth analysis. We perform close monitoring of performance metrics and numerical evaluations for model assessment.

This paper is committed to advancing ST technology for low resource languages. Through the creation of dedicated datasets and the development of robust models, our aim is to facilitate seamless communication and accessibility across diverse linguistic communities, ultimately promoting inclusivity and empowerment.

References

[1] Milind Agarwal, Sweta Agrawal, Antonios Anastasopoulos, Luisa Bentivogli, Ondřej Bojar, Claudia Borg, Marine Carpuat, Roldano Cattoni, Mauro Cettolo, Mingda Chen, William Chen, Khalid Choukri, Alexandra Chronopoulou, Anna Currey, Thierry Declerck, Qianqian Dong, Kevin Duh, Yannick Estève, Marcello Federico, et al. 2023. Findings of the iwslt 2023 evaluation campaign. In *Proceedings of the 20th International Conference on Spoken Language Translation (IWSLT 2023)*, pages 1–61, Toronto, Canada. Association for Computational Linguistics.

[2] Raymond Li, Wen Xiao, Lanjun Wang, Hyeju Jang, and Giuseppe Carenini. 2021. T3-Vis: visual analytic for training and fine-tuning transformers in nlp. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 220–230, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

[3] Alec Radford, Jong Wook Kim, Teng Xu, Greg Brockman, Conor McLeavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. Technical Report arXiv:whisper, OpenAI.

[4] Elizabeth Salesky, Kareem Darwish, Mohamed Al-Badrashiny, Mona Diab, and Jan Niehues. 2023. Evaluating multilingual speech translation under realistic conditions with resegmentation and terminology. In *Proceedings of the 20th International Conference on Spoken Language Translation (IWSLT 2023)*, pages 62–78, Toronto, Canada. Association for Computational Linguistics.

[5] Elizabeth Salesky, Marcello Federico, and Marine Carpuat. 2023. Proceedings of the 20th international conference on spoken language translation (iwslt 2023). In *Proceedings of the 20th International Conference on Spoken Language Translation (IWSLT 2023)*, Toronto, Canada. Association for Computational Linguistics.