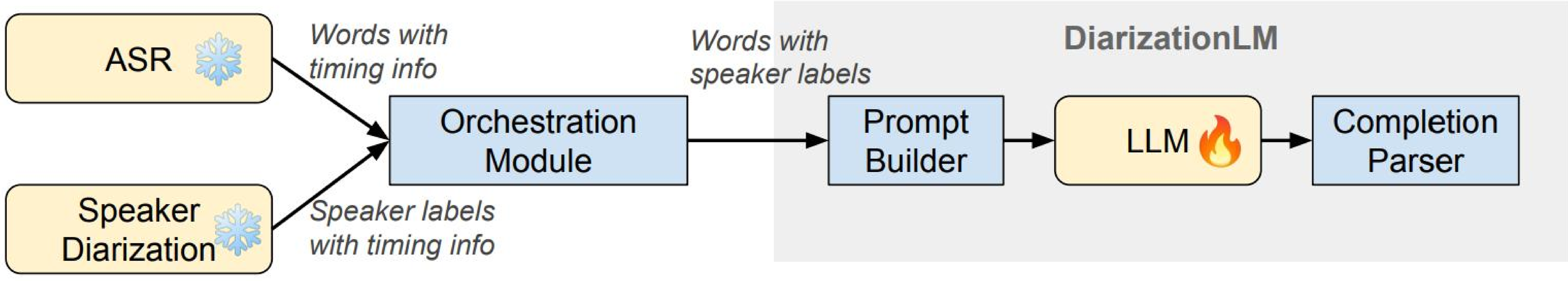
DiarizationLM: Speaker Diarization Post-Processing with Large Language Models

Quan Wang, Yiling Huang*, Guanlong Zhao*, Evan Clark, Wei Xia, Hank Liao*
*Google LLC.   *Equal contribution*
quanw@google.com



Overview

- DiarizationLM is a post-processing framework for ASR + speaker diarization systems.
- Works with any off-the-shelf ASR models and diarization algorithms.
- As a correction model, reduces Word Diarization Error Rates by >40% rel.
- We also discovered other diarization capabilities of LLM: autofilling speaker names, autofilling speaker roles, or replacing the orchestration module.

Prompt:

In the speaker diarization transcript below, some words are potentially misplaced. Please correct those words and move them to the right speaker. Directly show the corrected transcript without explaining what changes were made or why you made those changes.

<spk:1> Good morning Patrick, how
<spk:2> are you? Good, good. How are you Tom? Pretty
<spk:1> good. Going to work?
<spk:2> Yes. Busy day. How are your kids? Do they go
<spk:1> to school? Oh they are too young for that. I sent them to daycare earlier
<spk:2> today. Oh yeah I forgot about that.

Completion:

<spk:1> Good morning Patrick, how are you?
<spk:2> Good, good. How are you Tom?
<spk:1> Pretty good. Going to work?
<spk:2> Yes. Busy day. How are your kids? Do they go to school?
<spk:1> Oh, they are too young for that. I sent them to daycare earlier today.
<spk:2> Oh yeah, I forgot about that.

Fig. A motivating example where LLM can easily correct speaker diarization errors on the word level, even with a zero shot instruction.

Source of word diarization errors

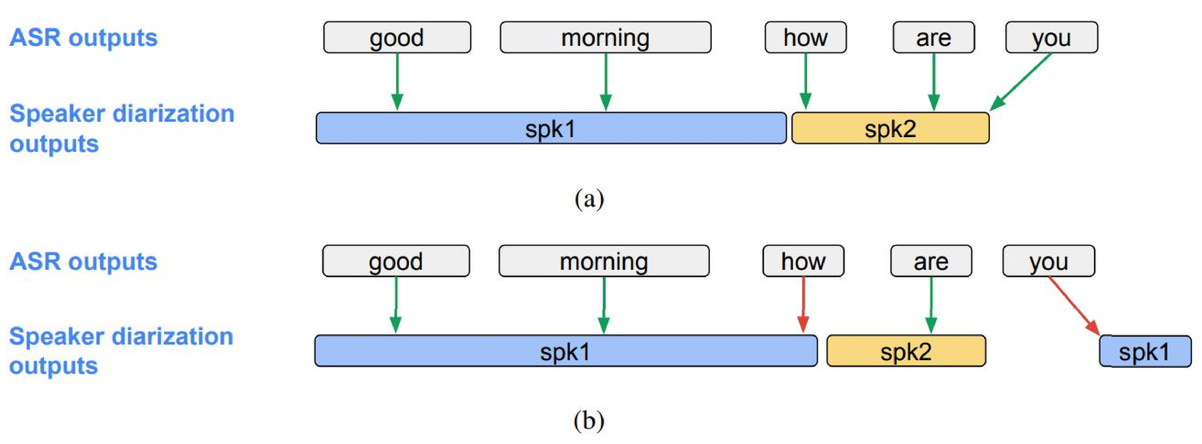


Fig. Orchestration between ASR and diarization results. (a) Correct output; (b) Incorrect output due to bad word-vs-speak timestamps.

- Errors may come from any individual module, e.g. VAD, ASR, segmentation, speaker embedding, clustering, etc.
- Errors may also come from the orchestration between these modules.
- Different components are trained with different datasets, algorithms, models, etc. Thus their timestamps may not match.

System description

- **Prompt builder:**
  - Converts word and speaker sequences to a text representation
  - Segment the sequences recursively to fit the LLM input length limit

```
Word sequence:        ["good", "morning", "how", "are", "you"]
Speaker sequence:     [1, 1, 2, 2, 2]
Text representation:  "<spk:1> good morning <spk:2> how are you"
```

- **Completion parser:**
  - Truncate undesired LLM output
  - Extract word and speaker sequences from text completion
  - Concatenate sequences of all segments from same utterance
  - Transfer speakers to original word sequence

Transcript-preserving spk transfer

- LLM input & output should have the same text, and only different speakers
- We need TPST in both prompt builder and completion parser
- `TPST(src_word, src_spk, tgt_word, tgt_spk) -> trans_spk`

```
Source words:         hello good morning hi how are you pretty good
Source speakers:      1 1 1 2 2 2 2 1 1
Target words:         hello morning hi hey are you be good
Target speakers:      1 2 2 2 1 1 2 1
transferred speakers: 1 1 2 2 2 2 1 1
```

Algorithm 1 The transcript-preserving speaker transfer (TPST) algorithm.

inputs
Source word sequence of length N: $\mathbf{w}^{src} = (w_1^{src}, \cdots, w_N^{src})$
Source speaker sequence of length N: $\mathbf{s}^{src} = (s_1^{src}, \cdots, s_N^{src})$
Target word sequence of length M: $\mathbf{w}^{tgt} = (w_1^{tgt}, \cdots, w_M^{tgt})$
Target speaker sequence of length M: $\mathbf{s}^{tgt} = (s_1^{tgt}, \cdots, s_M^{tgt})$

outputs
transferred speaker sequence of length M: $\mathbf{s}^{tra} = (s_1^{tra}, \cdots, s_M^{tra})$

procedure TPST($\mathbf{w}^{src}, \mathbf{s}^{src}, \mathbf{w}^{tgt}, \mathbf{s}^{tgt}$)
  Align $\mathbf{w}^{src}$ to $\mathbf{w}^{tgt}$ with the Levenshtein algorithm [38], resulting in a transform $f_{align}(\cdot)$
  $\mathbf{s}^{ali} \leftarrow f_{align}(\mathbf{s}^{src})$ ▷ $\mathbf{s}^{ali}$ is a speaker sequence of length M, and may contain blank speakers $\emptyset$ due to insertion errors in the alignment
  $K \leftarrow \max\{\max(\mathbf{s}^{ali}), \max(\mathbf{s}^{tgt})\}$ ▷ the maximal number of speakers in $\mathbf{s}^{ali}$ and $\mathbf{s}^{tgt}$
  Initialize a cost matrix $\mathbf{C} \in \mathbb{R}^{K \times K}$
  for $1 \le i \le K$ and $1 \le j \le K$ **do**
    $\mathbf{C}_{i,j} \leftarrow \sum_{1 \le m \le M} \delta(s_m^{ali} = i \text{ and } s_m^{tgt} = j)$
  end for
  Solve the assignment problem with cost matrix $\mathbf{C}$ using the Hungarian algorithm [39], resulting in a transform $f_{assign}(\cdot)$ ▷ handle speaker permutations
  for $1 \le m \le M$ **do**
    if $s_m^{ali} \neq \emptyset$ **then**
      $s_m^{tra} \leftarrow f_{assign}(s_m^{ali})$ ▷ transfer the speakers from the source
    else
      $s_m^{tra} \leftarrow s_m^{tgt}$ ▷ preserve the target speaker if source speaker is unavailable
    end if
  end for
end procedure

- **3 flavors of data processing:**
  - Flavor 1: hypothesis-to-oracle (hyp2ora)
    - *Prompt*: hypothesis word+ hypothesis speaker
    - *Completion*: hypothesis word + oracle speaker
  - Flavor 2: degraded-to-reference (deg2ref)
    - *Prompt*: reference word + degraded speaker
    - *Completion*: reference word + reference speaker
  - Flavor 3: mixed

Experiment results

- **Datasets:**
  - Training: 11527 utterances from Fisher
  - Testing: 172 utterance from Fisher + 20 utterances from Callhome English
- **Metrics:**
  - Word Error Rate (WER)
  - Word Diarization Error Rate (WDER)
  - Concatenated minimum-permutation Word Error Rate (cpWER)
- **Baseline system:**
  - ASR: Google Universal Speech Model (USM)
  - Speaker diarization: Turn-to-diarize + multi-stage clustering (unknown number of speakers)
- **LLMs experimented:**
  - PaLM 2-S & PaLM 2-L: 0-shot, 1-shot, finetuning with 3 flavors
  - Llama 2 & Llama 3: LoRA fintuning with mixed flavor

System	Fisher testing set			Callhome testing set		
	WER	WDER	cpWER	WER	WDER	cpWER
USM + turn-to-diarize baseline	15.48	5.32	21.19	15.36	7.72	24.39
+ PaLM 2-S zero-shot	-	11.96	30.19	-	12.26	30.60
+ PaLM 2-S one-shot	-	16.58	38.03	-	14.50	34.32
+ PaLM 2-L zero-shot	-	11.36	31.78	-	13.29	34.30
+ PaLM 2-L one-shot	-	5.94	22.21	-	7.95	24.67
+ PaLM 2-S finetuned (hyp2ora)	-	**2.37**	**16.93**	-	**4.25**	**20.22**
+ PaLM 2-S finetuned (deg2ref)	-	3.94	18.55	-	5.33	21.47
+ PaLM 2-S finetuned (mixed)	-	2.41	16.94	-	4.76	20.84
+ Llama 2 13B finetuned (mixed) v1	-	3.65	18.92	-	8.02	25.11
+ Llama 3 8B finetuned (mixed) v1	-	4.40	19.76	-	12.27	30.80
+ Llama 3 8B finetuned (mixed) v2	-	3.28	18.37	-	6.66	23.57

Patterns of successful corrections

- Different parts of sentence incorrectly attributed to the same speaker
- Short speaker turns due to disfluency
- Speaker turns due to interruptions

Utterance	Before DiarizationLM	After DiarizationLM
fe_03_07146 (Δ WDER =8.80%)	... <spk:3> it's it's <spk:1> more of summer always like you know we never experience a bit cold over here <spk:4> usually it'll <spk:1> be warm or like very hot in summer yeah and <spk:3> extremely hot yeah with high humidity my humidity is pretty <spk:1> much high because i stay close to the sea coast over here <spk:3> yeah <spk:1> so <spk:3> that makes you live houston is it like houston where you live yeah i i i live <spk:1> in houston ...	... <spk:1> it's it's more of summer always like you know we never experience a bit cold over here usually it'll be warm or like very hot in summer <spk:2> yeah and extremely hot yeah with high humidity my <spk:1> humidity is pretty much high because i stay close to the sea coast over here <spk:2> yeah so that makes you live houston is it like houston where you live <spk:1> yeah i i i live in houston ...
fe_03_06816 (Δ WDER =6.61%)	... <spk:3> uhuh <spk:2> did you see the the woman golfer that was on this the one <spk:1> monica yeah yeah <spk:2> what's her name monica stone yeah mhm she she <spk:1> blew out she fell out of that tournament but i didn't think she'd do it she she's girls can't compete against guys ...	... <spk:2> uhuh did you see the the woman golfer that was on this the one <spk:1> monica yeah yeah <spk:2> what's her name monica stone <spk:1> yeah <spk:2> mhm <spk:1> she she blew out she fell out of that tournament but i didn't think she'd do it she she's girls can't compete against guys ...

Utterance	Before DiarizationLM	After DiarizationLM
en_6447 (Δ WDER =12.49%)	... <spk:1> i'm <spk:2> going to see if i can talk to the guy that's selling the trailer if i can chew him down a bit uhhuh <spk:1> and <spk:2> you know what you just said benedicta is are you living with benedicta <spk:1> yes yes yes <spk:2> you know what i bet she answered the phone ...	... <spk:2> i'm going to see if i can talk to the guy that's selling the trailer if i can chew him down a bit <spk:1> uhhuh <spk:2> and you know what you just said benedicta is are you living with benedicta <spk:1> yes yes yes <spk:2> you know what i bet she answered the phone ...
en_6408 (Δ WDER =10.87%)	... <spk:1> uhu <spk:2> so <spk:1> he had big surgery again and he's in a wheelchair oh my <spk:2> and <spk:1> he doesn't want to go to school in a wheelchair uhuh but <spk:2> he might he wants to have tutoring at home but they're still where they lived on 45th street <spk:1> yeah they're there ...	... <spk:2> uhu <spk:1> so he had big surgery again and he's in a wheelchair <spk:2> oh my <spk:1> and he doesn't want to go to school in a wheelchair <spk:2> uhuh <spk:1> but he might he wants to have tutoring at home <spk:2> but they're still where they lived on 45th street <spk:1> yeah they're there ...

Python library

`https://github.com/google/speaker-id/tree/master/DiarizationLM`

- **Install the library:**
  - `pip install diarizationlm`
- **Prepare data in json format:**

Field	Description
`"utterance_id"`	This stores the utterance ID.
`"hyp_text"`	This stores the sequence of hypothesis words, but joined by spaces.
`"hyp_spk"`	This stores the sequence of hypothesis speakers, but joined by spaces.
`"hyp_diarized_text"`	This is the text representation of the hypothesis words and speakers. It can be used for debugging and to build the prompts to LLM.
`"ref_*"`	Similar to the `"hyp_*"` fields, but these are ground truth reference, rather than hypothesis.

- **Convert between word/spk sequences and pure text:**
  - `create_diarized_text()` - sequences to text
  - `extract_text_and_spk()` - text to sequences
- **Transcript-preserving speaker transfer (TPST):**
  - `def transcript_preserving_speaker_transfer(`
  - `    src_text: str, src_spk: str,`
  - `    tgt_text: str, tgt_spk: str) -> str`
- **Compute metrics:**
  - `python3 compute_metrics_on_json.py \`
  - `  --input=testdata/example_data.json \`
  - `  --output=/tmp/example_metrics.json`
- **Finetune llama model with unsloth:**
  - `cd unsloth && python3 1_finetune.py`
- **Play with demo:**
  - `https://huggingface.co/spaces/diarizers-community/DiarizationLM-GGUF`

prompt

<speaker:1> Hello, my name is Tom. May I speak to Laura <speaker:2> please? Hello, this is Laura. <speaker:1> Hi Laura, how are you? This is <speaker:2> Tom. Hi Tom, I haven't seen you for a <speaker:1> while.

output

<speaker:1> Hello, my name is Tom. May I speak to Laura please? <speaker:2> Hello, this is Laura. <speaker:1> Hi Laura, how are you? This is Tom. <speaker:2> Hi Tom, I haven't seen you for a while.