ICASSP 2023
4 - 10 JUNE, RHODES ISLAND, GREECE

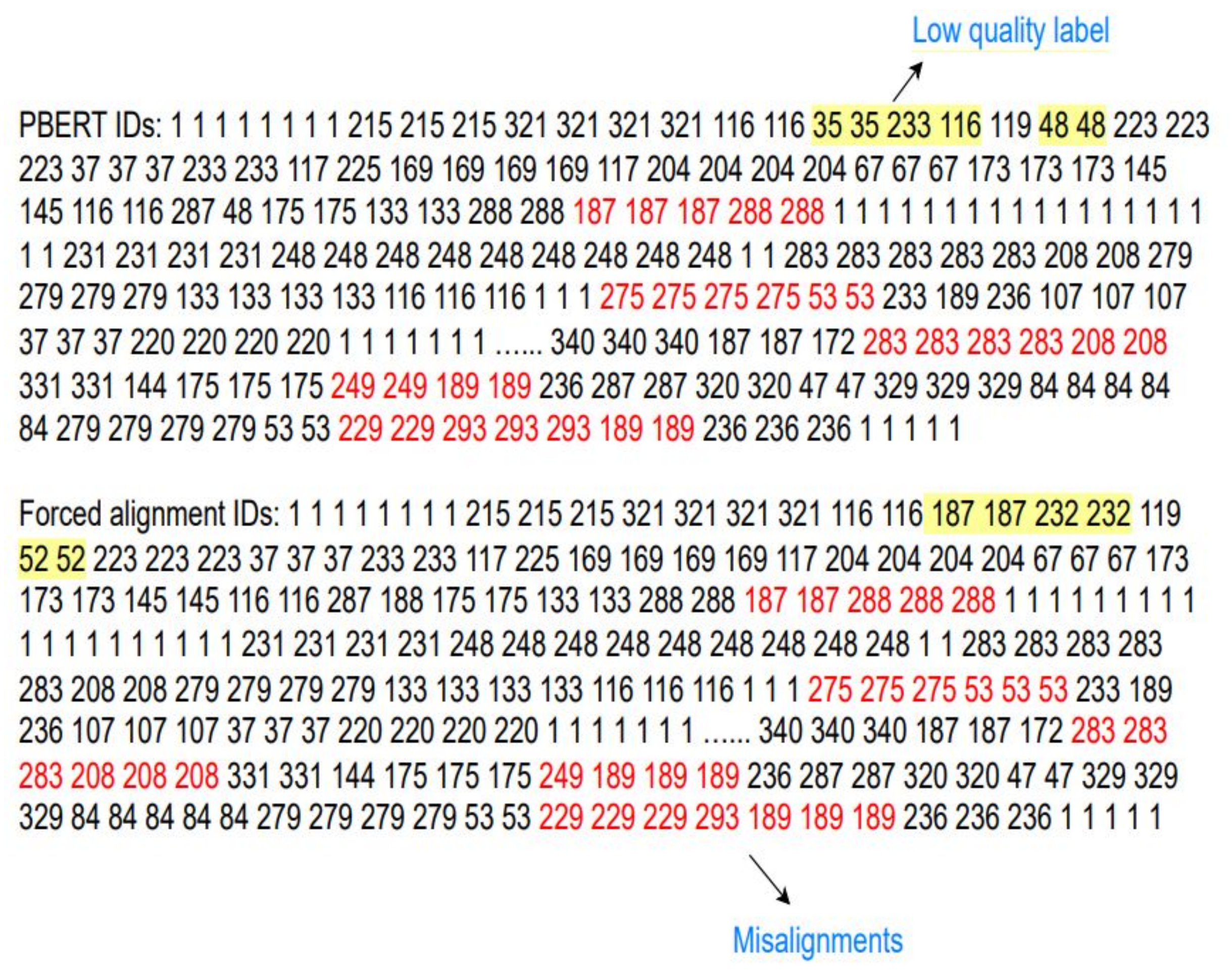
CTCBERT: Advancing Hidden-unit BERT with CTC Objectives

Ruchao Fan[1], Yiming Wang[2], Yashesh Gaur[2], Jinyu Li[2]
fanruchao@g.ucla.edu
[1]University of California, Los Angeles, USA
[2]Microsoft Corporation, USA

Introduction

- Recently, self-supervised learning (SSL) has achieved a great success in ASR, especially for low-resource tasks.
- Typical ID-based SSL learning in speech:
  - HuBERT: clustered IDs from MFCC or hidden features
  - PBERT: aligned IDs from hybrid models trained with a small amount of supervised data
- ID-based SSL methods are similar to acoustic model training with cross-entropy (CE) loss.
- **However, we observe significant errors in these IDs, including low-quality labels and misalignment (Figure 1).**

Figure 1. PBERT and forced alignment IDs for an utterance in Librispeech test other data.. The forced aligned IDs are obtained from a hybrdi model trained with 960h data.


Low quality label
Misalignments

PBERT IDs: 1 1 1 1 1 1 1 1 215 215 215 321 321 321 321 116 116 35 35 233 116 119 48 48 223 223 223 37 37 37 233 233 117 225 169 169 169 169 117 204 204 204 204 67 67 67 173 173 173 145 145 116 116 287 48 175 175 133 133 288 288 187 187 187 288 288 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 231 231 231 231 248 248 248 248 248 248 248 248 248 1 1 283 283 283 283 283 208 208 279 279 279 279 133 133 133 133 116 116 116 1 1 1 275 275 275 275 53 53 233 189 236 107 107 107 37 37 37 220 220 220 220 1 1 1 1 1 1 1 ...... 340 340 340 187 187 172 283 283 283 283 208 208 331 331 144 175 175 175 249 249 189 189 236 287 287 320 320 47 47 329 329 329 84 84 84 84 84 279 279 279 279 53 53 229 229 293 293 293 189 189 236 236 236 1 1 1 1 1

Forced alignment IDs: 1 1 1 1 1 1 1 1 215 215 215 321 321 321 321 116 116 187 187 232 232 119 52 52 223 223 223 37 37 37 233 233 117 225 169 169 169 169 117 204 204 204 204 67 67 67 173 173 173 145 145 116 116 287 188 175 175 133 133 288 288 187 187 288 288 288 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 231 231 231 231 248 248 248 248 248 248 248 248 248 1 1 283 283 283 283 283 208 208 279 279 279 279 133 133 133 133 116 116 116 1 1 1 275 275 275 53 53 53 233 189 236 107 107 107 37 37 37 220 220 220 220 1 1 1 1 1 1 1 ...... 340 340 340 187 187 172 283 283 283 208 208 208 331 331 144 175 175 175 249 189 189 189 236 287 287 320 320 47 47 329 329 329 84 84 84 84 84 279 279 279 279 53 53 229 229 229 293 189 189 189 236 236 236 1 1 1 1 1

- Low quality labels can be improved by iterative pretraining, such as HuBERT 1st and 2nd iterations.
- Inspired by the example in Figure 1, **we propose to use CTC as the objective function to perform ID-based SSL training, and denote it as CTCBERT.**
- We examine CTCBERT on IDs from HuBERT Iter1, HuBERT Iter2, and PBERT. The CTC training **brings consistent improvements compared to the CE training.**
- Slight improvements are observed when loading blank-related parameters during finetuning.

Background

1. ID-based SSL for speech

- **Hidden-unit BERT (HuBERT)**
  - K-means clustering to create a learning ID (pseudo-label) for each frame
  - Iterative pretraining to improve the label quality
  - 1st iter on MFCC features, 2nd iter on HuBERT features
- **Phoneme BERT (PBERT):**
  - training a hybrid model with the paired finetuning data
  - Applying decoding on the training data to get the ID for each frame
  - One iteration is enough because of the high label quality
- Let Z be the hidden feature sequence, C be the learning IDs, then the ID-based SSL models can be trained with:.

$$L_{\text{ID\_SSL}} = -\sum_{t \in Z_{mask}} logP(c_t|Z)$$

2. Connectionist Temporal Classification (CTC)

- CTC introduces an additional “blank” token to solve the length mismatch between the input and output sequences.
- Define a mapping rule β that removes blank and repetitive tokens of a sequence, CTC can be described as:

$$L_{\text{CTC}} = -\sum_{a \in \beta^{-1}(Y)} \prod_{t=1}^{T} P(a_t|X')$$

CTCBERT

1. Motivation

- The example (last row) in Figure 1:
  - Aligned IDs: 229 229 293 293 293 189 189
  - Ground-truth IDs: 229 229 229 293 189 189 189
  - For CE loss, two frames get wrong targets
  - For CTC loss, the targets are {229, 293, 189} for both IDs.
- CTC training is not sensitive to the frame alignment
- CTCBERT: constructing training targets
  - Define a rule A to remove the repetitive IDs in each mask region, the IDs after operation A are the training targets.
  - Let $Z_{mask} = \{Z_1, Z_2, ..., Z_M\}$ be the masked regions, $C = \{C_1, C_2, ..., C_M\}$ be the original IDs for those regions, the CTCBERT loss can be described as:

$$L_{\text{CTCBERT}} = -\sum_{m=1}^{M} \sum_{a \in \beta^{-1}(A(C_m))} \prod_{t \in Z_m} P(a_t|Z)$$

2. Stabilize the CTC training

- CTC training of the ID-based SSL methods causes slow convergence sometimes.
- To stabilize the training, we can use:
  - CE loss warmup
  - CTC and CE joint training

$$L_{\text{joint}} = \alpha L_{\text{CTCBERT}} + (1-\alpha) L_{\text{HuBERT}}$$

3. Manipulate blank-related parameters

- CTCBERT pretraining involves blank-related parameters, which also exist in finetuning with CTC loss.
- The corresponding parameters can be used for initialization for better performance.

Experimental Settings

- Dataset:
  - Librispeech 960 hours as the pretraining data
  - Train-clean-100 subset as the finetuning data
- Model architecture is the same as HuBERT [22]
- Pretraining:
  - HuBERT iter1 IDs: k-means of MFCC features
  - HuBERT iter2 IDs: k-means of 9-th layer HuBERT features from the first iteration
  - PBERT IDs: decode 960-hour data with a hybrid model trained with train-clean-100 subset
  - All models are trained on 32 GPUs with a batch size of at most 87.5 seconds of audios per GPU.
  - Same training strategies as the HuBERT [22]
- Finetuning
  - 8 GPUs with a batch size of 200s audio per GPU
  - Freeze the convolutional feature extractor
  - CTC loss training for 80k steps
  - Beam search decoding: a beam size of 1500 and a 4-gram LM using wav2letter++ decoder [30].

Results

1. Main Results

Table 1. The performance of CTCBERT using IDs from HuBERT Iter1, HuBERT Iter2, and PBERT. “Load blk” indicates loading blank related parameters during finetuning.

Model	LM	dev		test	
		clean	other	clean	other
Supervised baseline	None	-	-	8.8	26.5
	4-gram	-	-	5.0	16.8
HuBERT iter1 [22]	None	7.1	16.7	7.2	16.5
	4-gram	3.2	9.6	3.8	9.5
→ CTCBERT	None	6.9	15.9	7.0	15.8
	4-gram	3.2	9.2	3.8	9.4
+ load blk.	None	6.8	15.8	6.9	15.7
	4-gram	3.2	9.2	3.7	9.3
HuBERT iter2 [22]	None	5.3	13.7	5.5	13.6
	4-gram	2.8	8.7	3.4	8.7
→ CTCBERT	None	5.2	12.3	5.4	12.1
	4-gram	2.7	7.8	3.3	7.9
+ load blk.	None	5.2	12.4	5.4	12.1
	4-gram	2.7	7.9	3.3	7.9
PBERT [23]	None	4.6	11.7	4.8	11.8
	4-gram	2.6	7.3	3.2	7.7
→ CTCBERT	None	4.7	11.5	4.8	11.4
	4-gram	2.5	7.1	3.2	7.5
+ load blk.	None	4.6	11.3	4.8	11.3
	4-gram	2.5	7.1	3.1	7.4

- CTCBERT consistently improves the WER performance on the three examined training IDs over the CE training.
- **The improvements are larger on noisy data.** It may be because noisy speech are more likely to generate IDs with misalignments.
- Using blank-related parameters during finetuning further improves the performance slightly.
- Note that we replicate the results of HuBERT and PBERT, and may cause WER differences (different IDs from different k-means cluster). However, the comparisons are reasonable because of the same learning IDs for CE and CTC training.

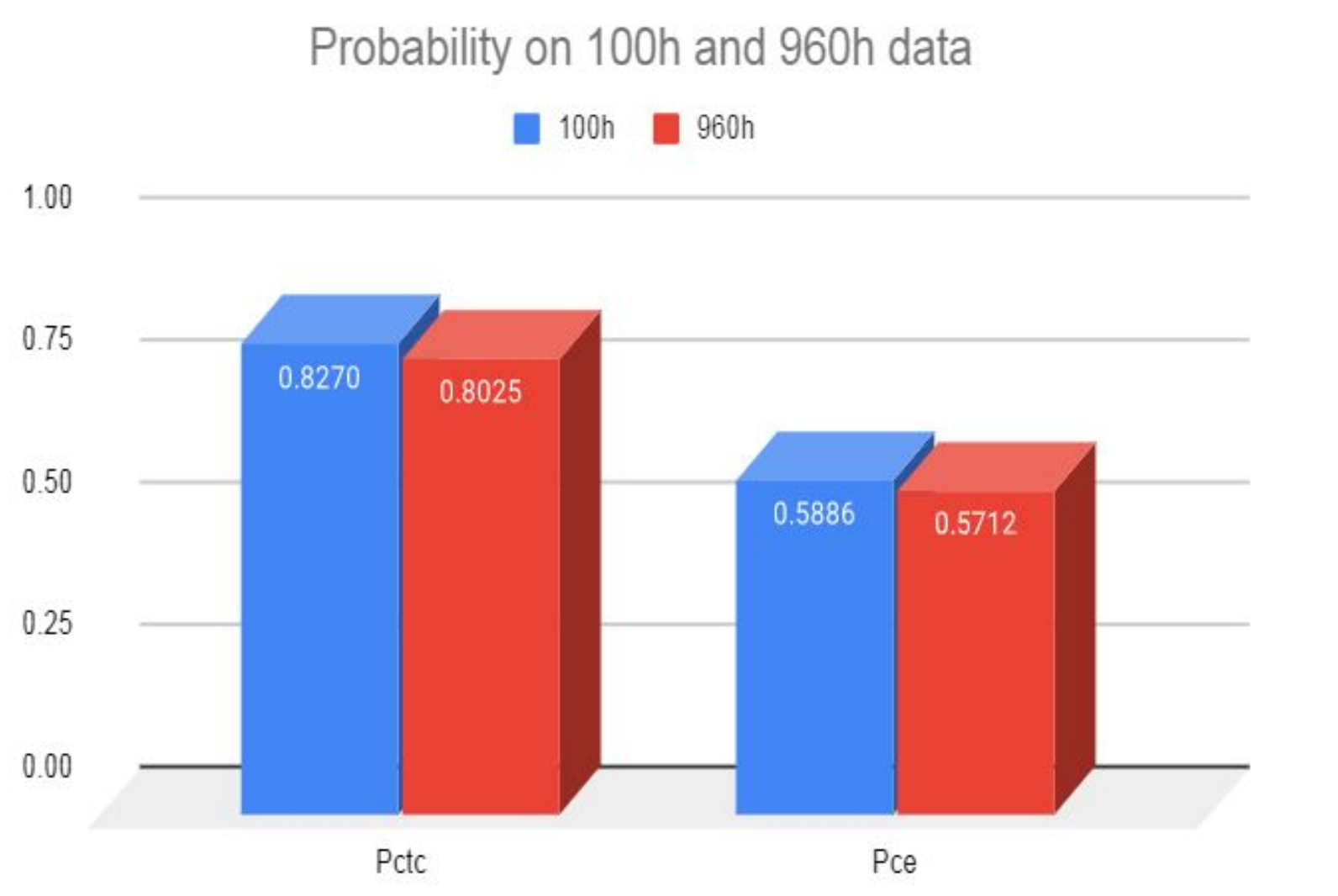
2. Stabilize the CTC training

Table 2. The effect of the CE warmup and CTC, CE joint training.

Model	ce		dev		test	
	ratio	warmup	clean	other	clean	other
CTCBERT with iter1 IDs	0.0	0	7.2	16.8	7.5	16.7
	0.0	32k	7.2	15.8	7.3	15.8
	0.2	0	7.2	16.6	7.3	16.7
	0.5	0	6.9	15.9	7.0	15.8
	0.8	0	6.8	16.0	7.0	16.1
	1.0	0	7.1	16.7	7.2	16.5
CTCBERT with iter2 IDs	0.0	0	5.2	12.3	5.4	12.1
	0.0	32k	5.2	12.6	5.4	12.4
	0.5	0	5.5	13.3	5.6	13.1
	1.0	0	5.3	13.7	5.5	13.6
CTCBERT with PBERT IDs	0.0	0	5.5	13.8	5.7	13.9
	0.0	150k	4.7	11.6	4.9	11.7
	0.5	0	4.7	11.5	4.8	11.4
	1.0	0	4.6	11.7	4.8	11.8

- CTC training sometimes shows slow convergence.
- ratio = 0, pure CTC training; ratio=1, pure CE training
- CE warmup and CTC, CE joint training can both help the convergence in the pretraining.
- CE and CTC joint training with a half-half ratio achieves the best performance with HuBERT Iter1 and PBERT IDs.
- Pure CTC training for HuBERT iter2 IDs is better.
- Overall, 0**.5CTC + 0.5CE would the best choice for CTCBERT.**

3. Quantitative Analysis



- Forced aligned IDs from a hybrid model trained with 100 hours data. The quality of IDs on 100h is better than 960h.
- Using the forced aligned IDs to compute posterior probability on 100h and 960h data.
- The relative probability decrease:
  - CTC: (0.8270-0.8025)/0.8270 = 2.9%
  - CE: (0.5886 - 0.5712)/0.5886 = 3.1%
- CTC causes less degradation on posterior probability, showing that it is more tolerate to misalignment.

Conclusion

- Training ID-based SSL system with CTC objectives offers more robust model to misalignment in the learned IDs.
- CTCBERT achieves consistent improvements on the HuBERT and PBERT IDs compared to the CE training.
- Empirical setting of a half-half ratio for CTC CE joint training can best stabilize the CTCBERT training.

References

The number is appeared as the same in the paper.