Joint Masked CPC and CTC Training for ASR

Chaitanya Talnikar, Tatiana Likhomanenko, Ronan Collobert, Gabriel Synnaeve

IEEE ICASSP 2021

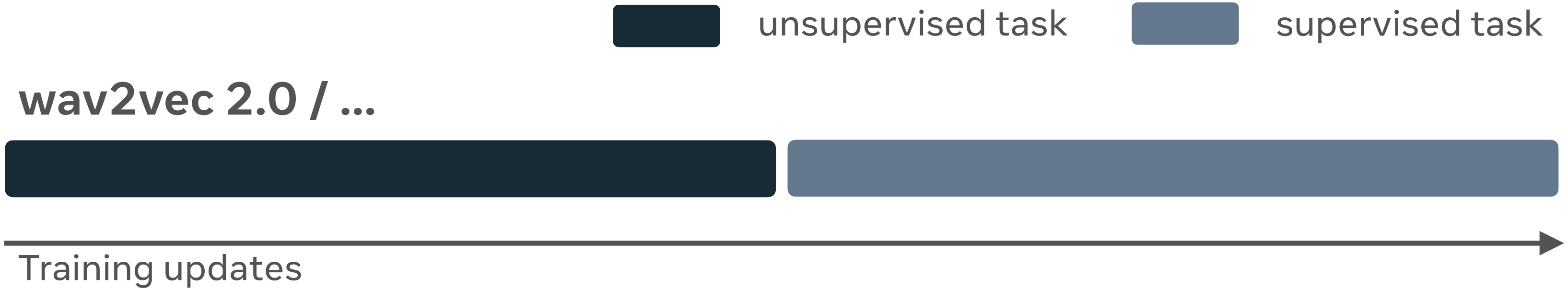
FACEBOOK AI

Agenda

1. Motivation
2. Joint training
3. Experimental setup
4. Results
5. Ablations
   - effect of hyperparameters on downstream task
   - regularization effect on supervised loss

Motivation

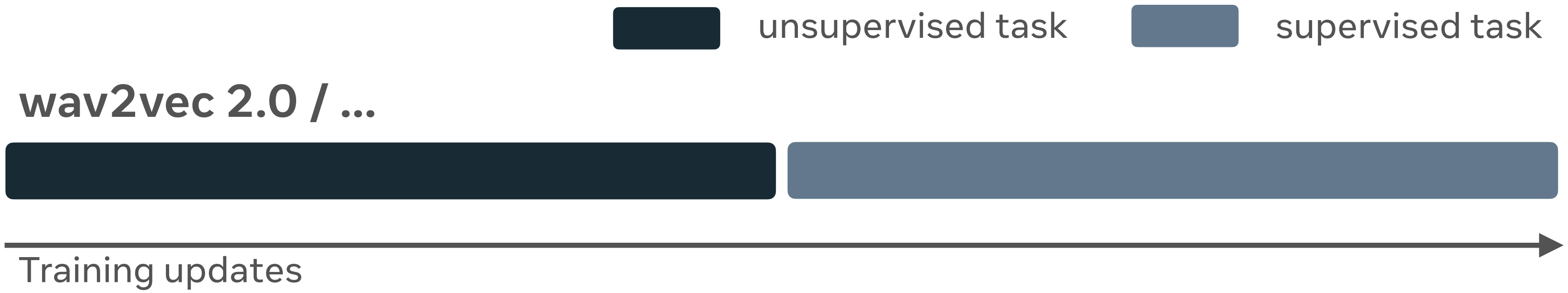
Two-stage Training



Self-supervised training for ASR requires **two stages**

- pre-training on unlabeled data
- fine-tuning on labeled data

Two-stage Training



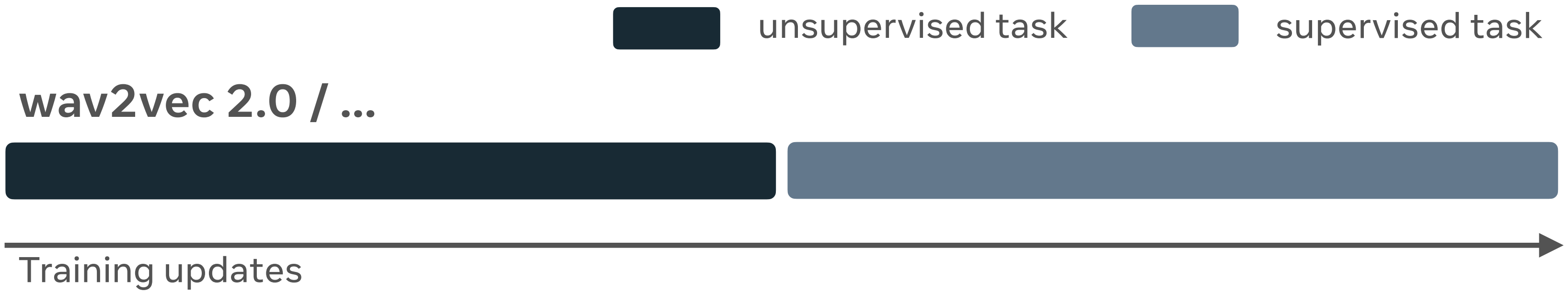
Self-supervised training for ASR requires **two stages**

- pre-training on unlabeled data
- fine-tuning on labeled data

Two-stage training is **hard to optimize** for a downstream task

unsupervised loss is not perfectly correlated with supervised task

Two-stage Training



Self-supervised training for ASR requires **two stages**

- pre-training on unlabeled data
- fine-tuning on labeled data

Two-stage training is **hard to optimize** for a downstream task

unsupervised loss is not perfectly correlated with supervised task

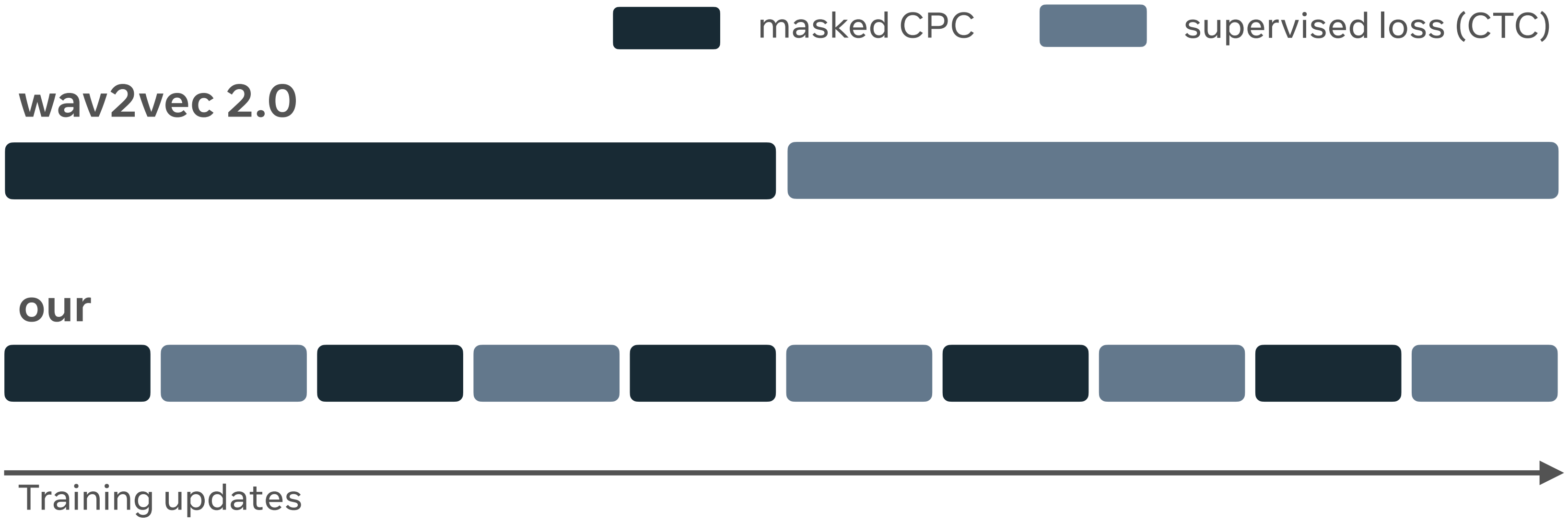
Revisit

propose alternate supervised and unsupervised minimization

Joint Training

Joint Training at Glance

We jointly minimize two losses, supervised L_s and an unsupervised L_u, by alternating between minimizing L_s on labeled data and minimizing L_u on unlabeled data.



Model

Model takes input raw audio $\boldsymbol{x}$ and outputs token $\boldsymbol{y}$ probabilities at time $\boldsymbol{t}$

$$\boldsymbol{z} = f(\boldsymbol{x}) \quad (1)$$

convolutional encoder

$$\tilde{\boldsymbol{z}} = g(\mathrm{mask}(\boldsymbol{z})) \quad (2)$$

transformer context network

$$p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x}) = h(\tilde{\boldsymbol{z}}). \quad (3)$$

Supervised and Unsupervised Losses

Supervised loss: Connectionist Temporal Classification (CTC)

$$\boldsymbol{z} = f(\boldsymbol{x}) \quad (1)$$

$$\tilde{\boldsymbol{z}} = g(\text{mask}(\boldsymbol{z})) \quad (2)$$

$$p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x}) = h(\tilde{\boldsymbol{z}}). \quad (3)$$

Supervised and Unsupervised Losses

Supervised loss: Connectionist Temporal Classification (CTC)

Unsupervised loss: wav2vec 2.0 self-supervision loss

can be viewed as a contrastive predictive coding (CPC) loss where the task is to predict the masked encoder features rather than predicting future encoder features given past encoder features

$$\mathcal{L}_u(\boldsymbol{\theta}, \boldsymbol{x}) = \frac{1}{T} \underbrace{\sum_t}_{\text{masked positions}} -\log \frac{s(\boldsymbol{z}_t, \tilde{\boldsymbol{z}}_t)}{s(\boldsymbol{z}_t, \tilde{\boldsymbol{z}}_t) + \underbrace{\sum_{t'} s(\boldsymbol{z}_{t'}, \tilde{\boldsymbol{z}}_t)}_{\text{non-masked positions}}} \qquad (4)$$

$$s(\boldsymbol{z}_t, \tilde{\boldsymbol{z}}_t) = \frac{1}{\tau} \exp\left(\frac{\boldsymbol{z}_t \cdot \tilde{\boldsymbol{z}}_t}{\|\boldsymbol{z}_t\| \|\tilde{\boldsymbol{z}}_t\|}\right)$$

$$\boldsymbol{z} = f(\boldsymbol{x}) \qquad (1)$$

$$\tilde{\boldsymbol{z}} = g(\text{mask}(\boldsymbol{z})) \qquad (2)$$

$$p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x}) = h(\tilde{\boldsymbol{z}}). \qquad (3)$$

Algorithm Overview

$$z = f(x) \quad (1)$$

$$\tilde{z} = g(\text{mask}(z)) \quad (2)$$

$$p_\theta(y|x) = h(\tilde{z}). \quad (3)$$

Alternate minimization:

separate adaptive momentum optimizers are used for each of the two losses with different learning rates η_s and η_u

optimizers maintain their state independently, while sharing the model parameters

Algorithm 1: Alternating minimization algorithm.

Data: Labeled data $L = \{x, y\}$, Unlabeled data $U = \{x\}$
Result: Acoustic model p_θ
Randomly initialize parameters of the acoustic model p_θ;
repeat
  repeat
    1. Forward the model with Eq. (1) and (2) obtaining z and $\tilde{z}$
    2. Compute $g_u = \nabla_\theta \mathcal{L}_u(\theta, x)$ using z, $\tilde{z}$
    3. Update p_θ with η_u and g_u
  until N *times for* $x \in U$;
  4. Forward the model for $x \in L$ with Eq. (1)-(3) obtaining $p_\theta(y|x)$
  5. Compute $g_s = \nabla_\theta \mathcal{L}_s(\theta, x, y)$ using $p_\theta(y|x)$
  6. Update p_θ with η_s and g_s
until *convergence in word error rate or maximum iterations are reached*;

Experimental Setup

Experiments

Data:

i) 960h of LibriSpeech is used as unlabeled set

ii) 100h of train-clean LibriSpeech is used as labeled

Experiments

Data:

i) 960h of LibriSpeech is used as unlabeled set

ii) 100h of train-clean LibriSpeech is used as labeled

Models have wav2vec 2.0 architectures

- Base 94M
- Large 315M

Tokens: English alphabet

Experiments

Data:

i) 960h of LibriSpeech is used as unlabeled set;

ii) 100h of train-clean LibriSpeech is used as labeled.

Models have wav2vec 2.0 architectures

- Base 94M
- Large 315M

Tokens: English alphabet

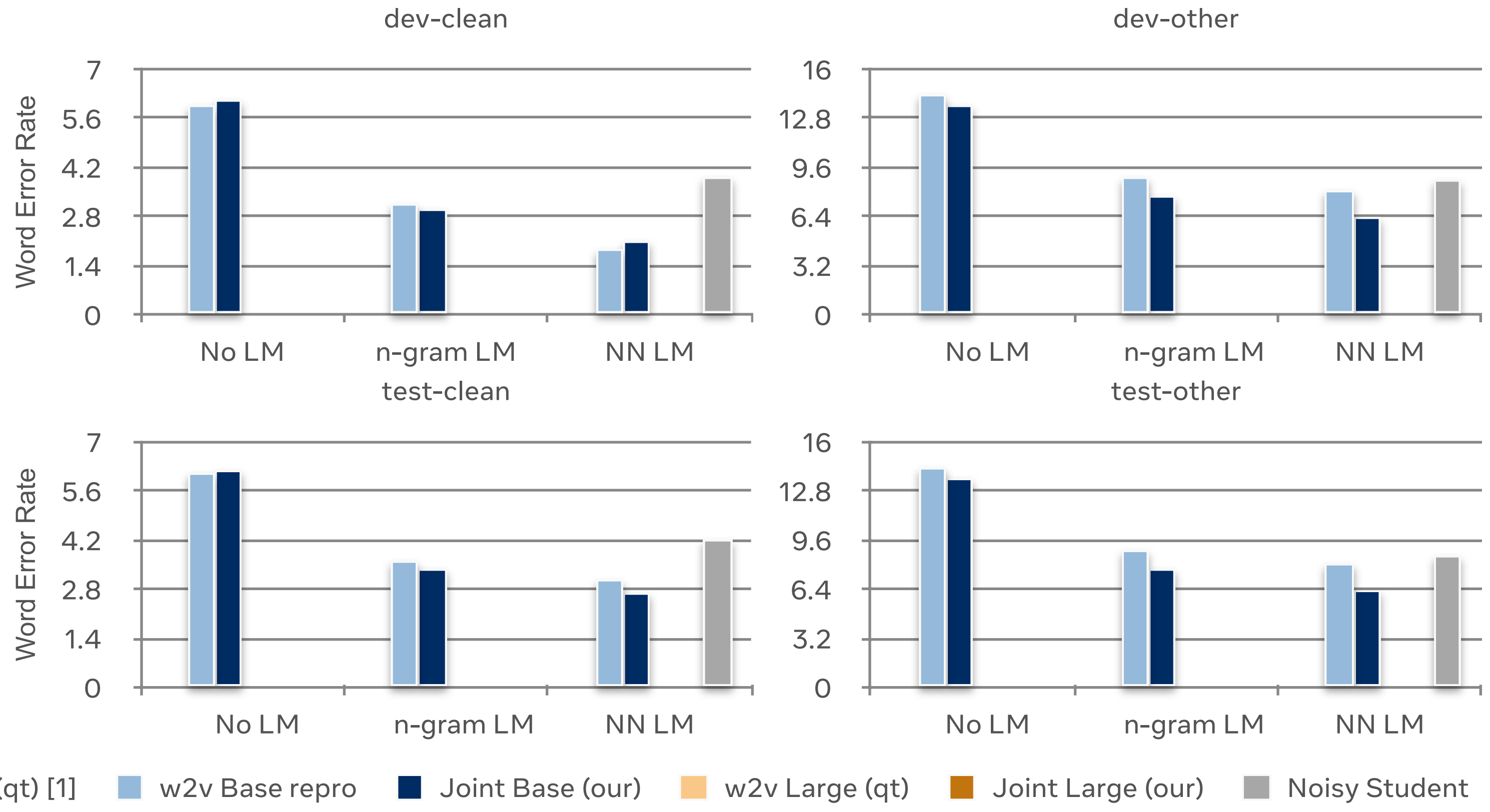
Data augmentation in the ASR task:

a variation of SpecAugment that uses the same masking procedure as the contrastive loss

Training: 500k updates with Adam optimizer

Results

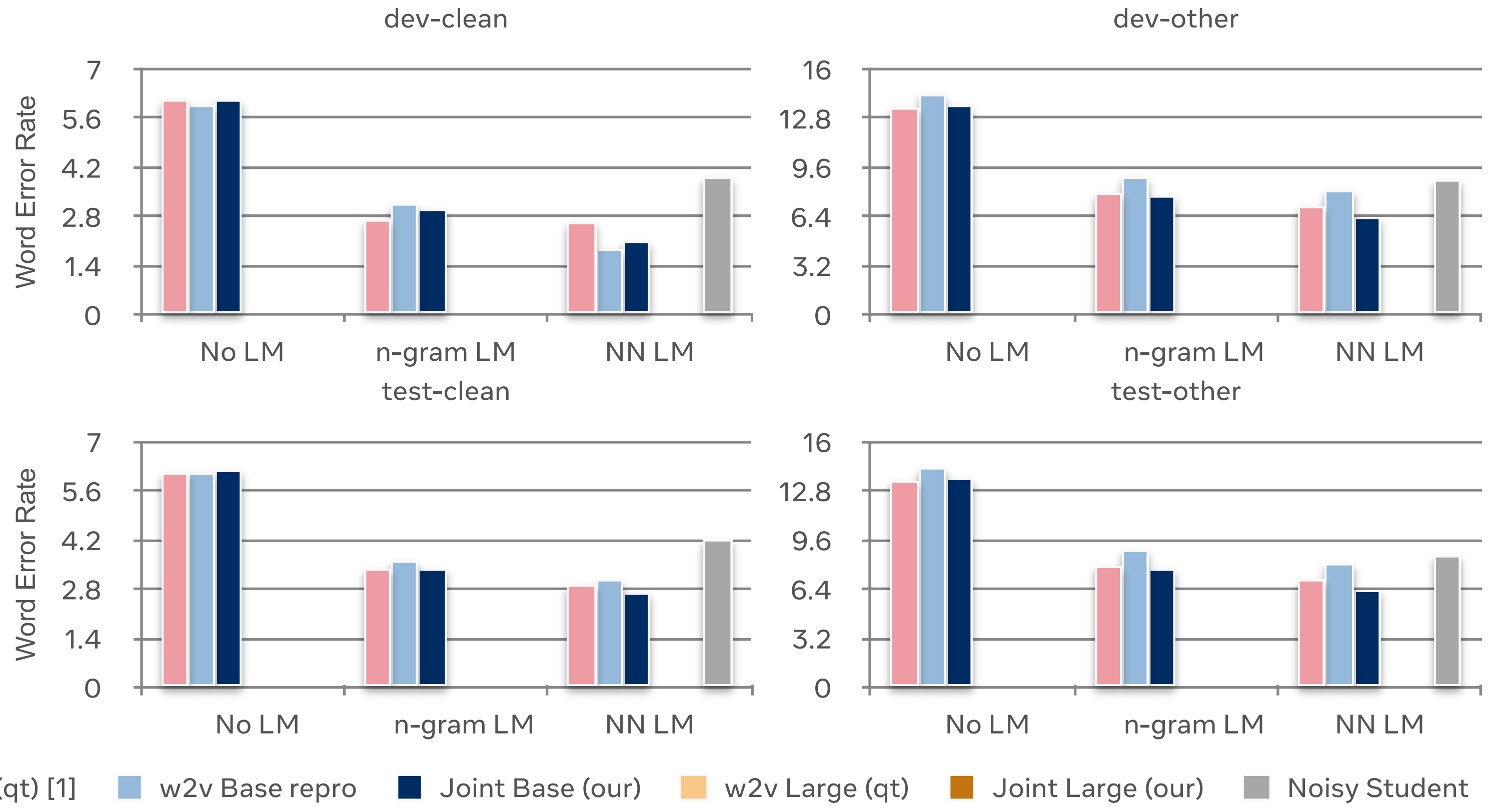
Results: Base Model (Continuous)



joint training is **better** than wav2vec 2.0

Results: Base Model

joint training
≈
wav2vec 2.0 (qt)

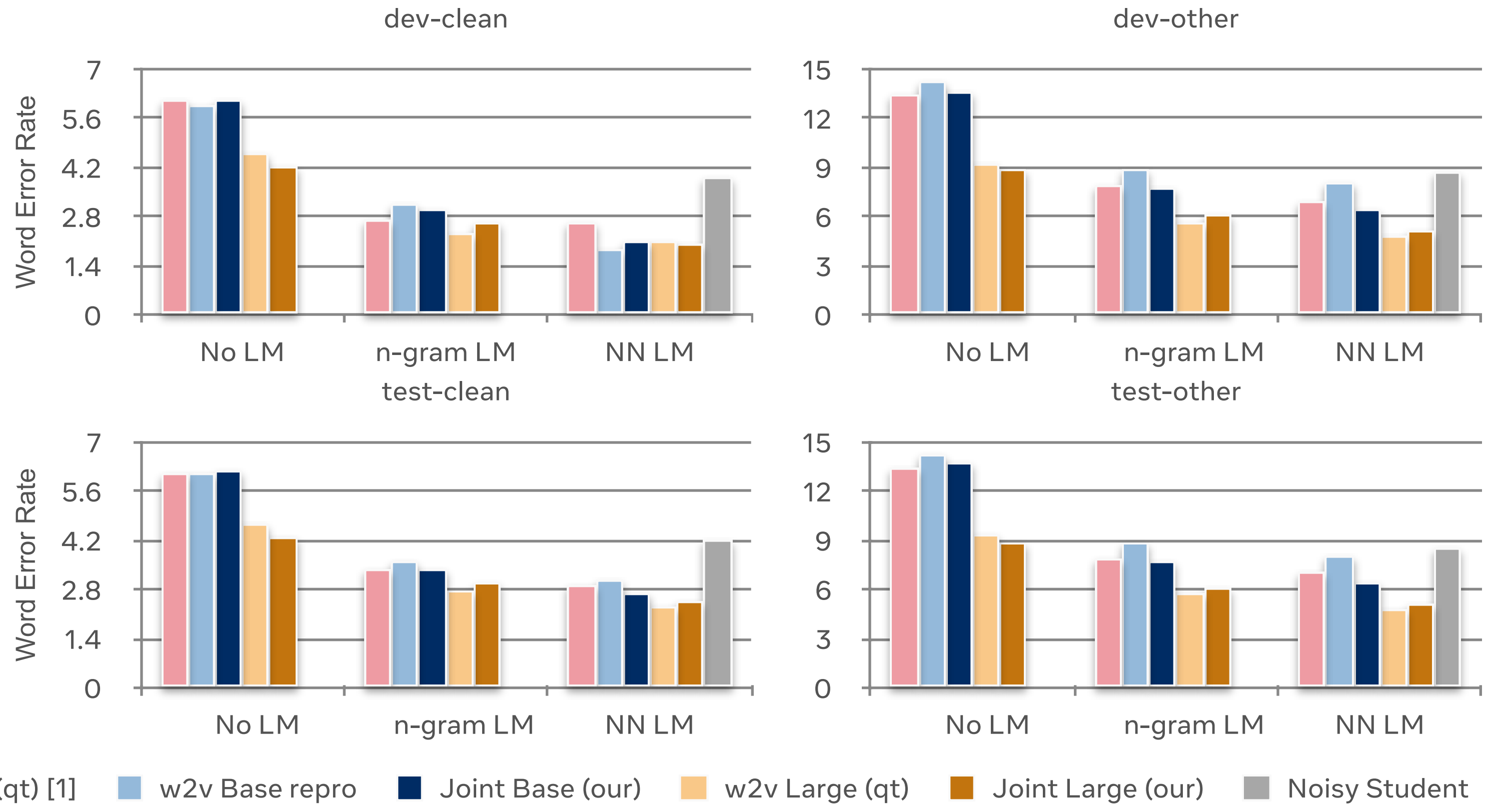


Results: Large Model

joint training

≈

wav2vec 2.0 (qt)



Results: Simpler but with the Same WER

Best wav2vec 2.0 models use

- features quantization

Joint model in contrast

- **quantization-free,** operates in the continuous space

Results: Simpler but with the Same WER

Best wav2vec 2.0 models use

- features quantization
- unsupervised penalty terms during training

Joint model in contrast

- quantization-free, operates in the continuous space
- **does not** use any unsupervised penalty terms

Ablations

Ablation:
Effect of Hyperparameters on Downstream Tasks

- Training is not sensitive to the number of L_u to L_s updates

Hyperparameter	Updates	LR	dev-other WER
Baseline	1:1	20:1	**8.0**
L_u to L_s update ratio	5:1	20:1	**7.9**
L_u to L_s learning rate ratio	1:1	4:1	9.0
Single optimizer	1:1	20:1	11.1

Ablation:
Effect of Hyperparameters on Downstream Tasks

- Training is not sensitive to the number of L_u to L_s updates
- Lower L_u to L_s learning rate ratio or a single optimizer results in a higher WER

Hyperparameter	Updates	LR	dev-other WER
Baseline	1:1	20:1	8.0
L_u to L_s update ratio	5:1	20:1	7.9
L_u to L_s learning rate ratio	1:1	4:1	**9.0**
Single optimizer	1:1	20:1	**11.1**

Ablation: Regularization Effect on Supervised Loss

Baseline model

- a supervised model trained on full labeled LibriSpeech (960h)

Ablation: Regularization Effect on Supervised Loss

Baseline model

- a supervised model trained on full labeled LibriSpeech (960h)

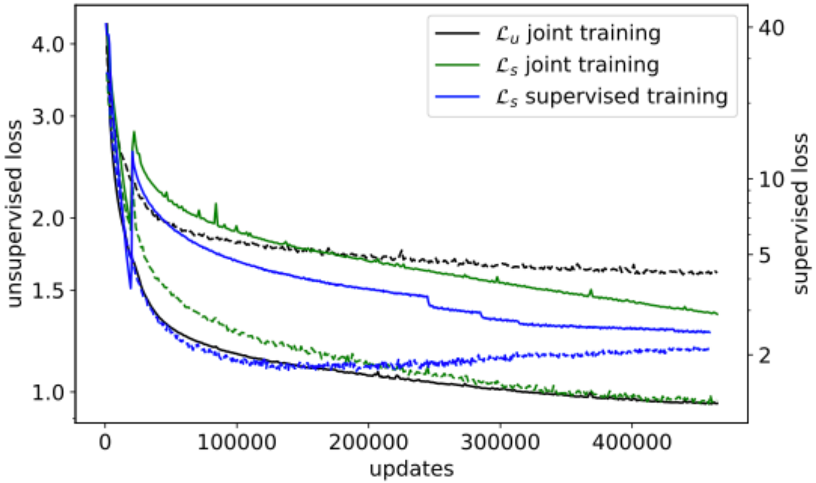
Joint model

- full LibriSpeech without labels is used to compute unsupervised loss
- full LibriSpeech with labels is used to compute supervised loss

Ablation: Regularization Effect on Supervised Loss

Joint training achieves (compared to supervised training):

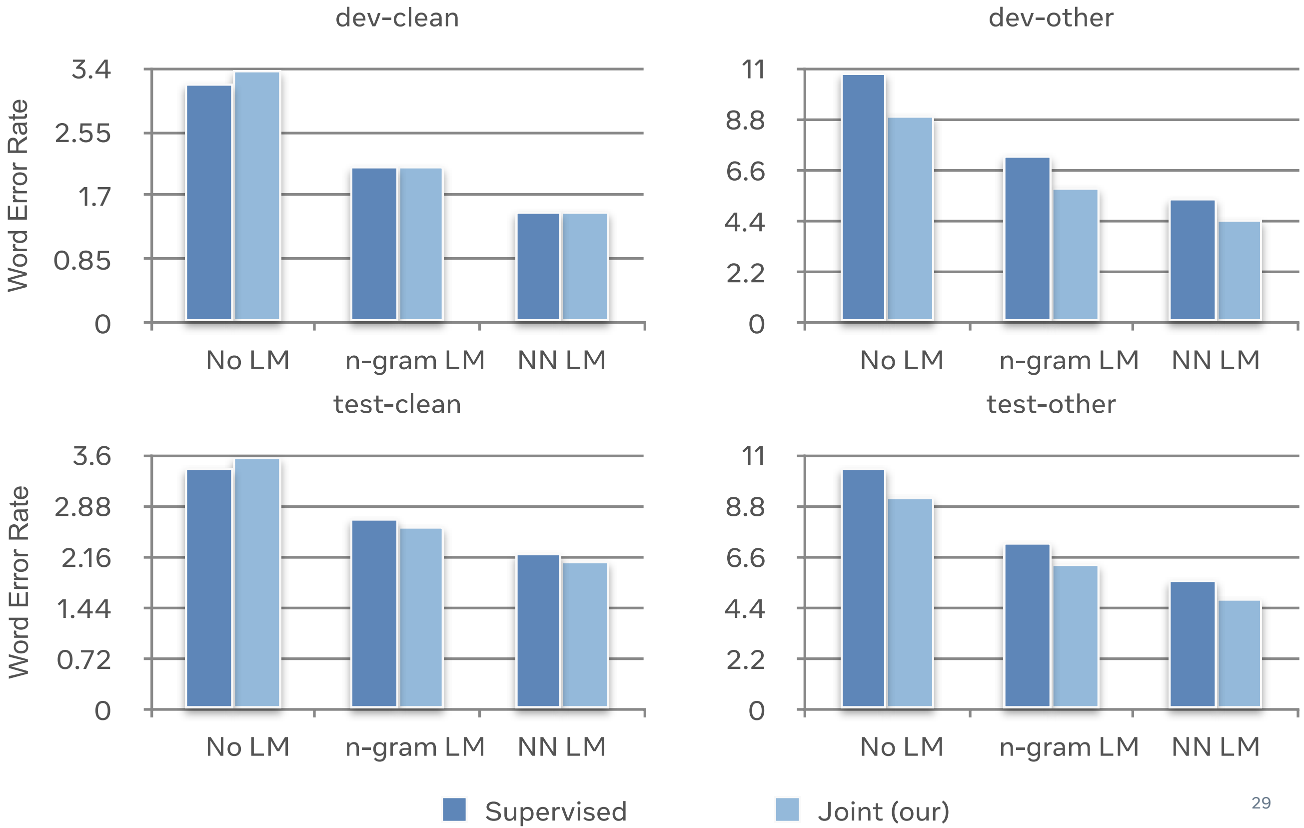
- **lower** supervised loss on the **validation** (dotted)
- **higher** supervised loss on the **train** (solid)



Ablation: Regularization Effect on Supervised Loss

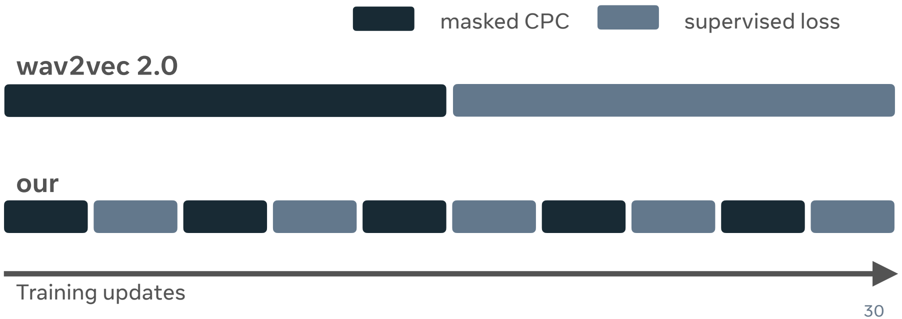
Also joint training achieves lower WER

despite lower number of updates from supervised loss



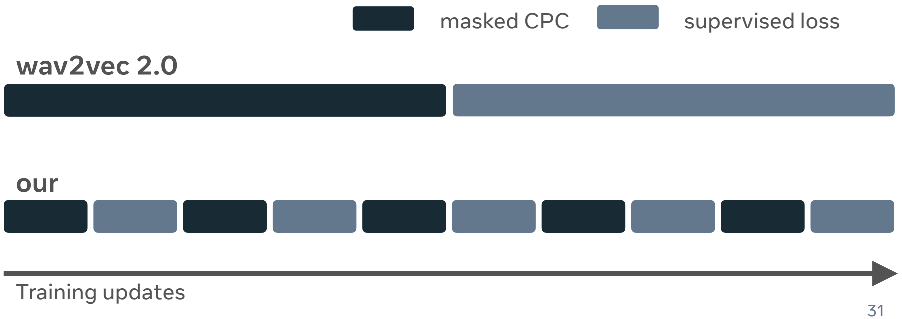
Conclusion

- We proposed joint training: alternate supervised and unsupervised losses minimization



Conclusion

- We proposed joint training: alternate supervised and unsupervised losses minimization
- Joint training
    - simplifies training process
    - directly optimizes for ASR task rather than for unsupervised task
    - matches state-of-the-art two-stages training



Thank You