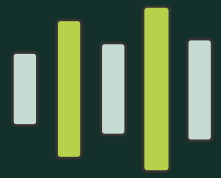


wav2vec 2.0

A Framework for Self-Supervised Learning of Speech Representations

Alexei Baevski, Henry Zhou, Abdelrahman Mohamed & Michael Auli

Advances in Neural Information Processing Systems 33 (NeurIPS 2020), pp. 12449–12460

Facebook AI · arXiv:2006.11477 · code and models released in fairseq

Why this paper matters

The result that changed what a speech dataset needs to look like

10 min

of labelled speech — 48 recordings — fine-tuned to 4.8/8.2 WER

53.2k h

of unlabelled audio used for pre-training

100×

less labelled data than the previous state of the art it beat

1.8/3.3

WER on Librispeech test-clean/other with the full 960 labelled hours

The claim

Learn representations from raw audio alone, then fine-tune on transcribed speech, and you can outperform the best semi-supervised methods while being conceptually simpler. The paper states this as a first: not a marginal gain, but a demonstration that the labelled-data requirement for usable speech recognition can drop by two orders of magnitude.

Why it matters to us specifically

Nothing in the pre-training objective mentions words, phonemes or transcripts. The representations turn out to be useful far beyond recognition — for emotion, speaker, intent, health screening. In the last section we will see that they are useful for emotion in a way the authors neither designed nor predicted.

The problem the paper starts from

Labelled speech is scarce; audio is not

The supervised bottleneck

Speech recognition systems of the period needed thousands of hours of transcribed speech to reach acceptable accuracy. Transcription is slow, expensive and requires literate annotators fluent in the target language.

The coverage problem

There are around seven thousand languages in the world, and many more dialects. For the vast majority, no such corpus exists and none is likely to be built. The paper makes this its explicit broader-impact argument.

The learning-theoretic complaint

Learning purely from labelled examples does not resemble human language acquisition. Infants learn by listening to the adults around them — a process that requires building good representations of speech before any labels arrive.

The proposal: use the unlabelled audio, which is abundant, to learn the structure of speech; then use the scarce labelled data only to attach that structure to a symbol set.

Note the intellectual debt: this is the recipe that had already reshaped natural language processing through ELMo, GPT and BERT. Much of the paper's contribution is working out what has to change when the input is a continuous waveform rather than a sequence of discrete tokens.

1 Background and lineage

What came before, and what this paper changes

Why BERT does not simply transfer to speech

Three properties of audio that break the masked-language-modelling recipe

No vocabulary	Text arrives pre-segmented into a finite set of tokens. A waveform is a continuous, real-valued signal with no inventory to predict over and no natural unit boundaries.
No segmentation	Words and phonemes are not delimited in the signal. There is no equivalent of whitespace, and segment durations vary enormously with speaker and rate.
Enormous redundancy	Sixteen thousand samples per second, most of which are predictable from their neighbours. Reconstructing the input is close to trivial and teaches the model almost nothing.
Nuisance variation	The same words carry speaker identity, room acoustics, microphone response, background noise. A model that reconstructs faithfully has learned the nuisance along with the content.

So the design problem is: what do you predict, and over what inventory?

The lineage

Four immediate predecessors, all cited in the paper

CPC
van den Oord et al., 2018

Contrastive predictive coding: predict future latents rather than reconstruct. Establishes the contrastive objective for sequential data.

wav2vec
Schneider et al., 2019

CPC applied to ASR with a convolutional encoder and context network. Continuous representations, no discretisation, no Transformer.

vq-wav2vec
Baevski et al., 2020

Adds vector quantisation, producing a discrete token sequence — which a BERT model can then be trained on. But in two separate stages.

Discrete BERT
Baevski, Auli & Mohamed, 2019

The two-stage pipeline realised: quantise first, then run masked language modelling over the frozen units. The direct baseline in every results table.

The problem with the two-stage pipeline

In vq-wav2vec and Discrete BERT the codebook is frozen before the context model is ever trained. Whatever the quantiser throws away in stage one is gone for good, and the quantiser has no signal telling it what the context model will need.

wav2vec 2.0's central move is to learn both at once, end to end — and, as we will see, to feed the Transformer the continuous latents rather than the discrete ones.

The paper in one slide

Four decisions, each of which the rest of the paper defends

Work from the raw waveform

A convolutional encoder learns its own front end instead of consuming mel filter banks. Nothing hand-engineered enters the model.

Mask in latent space, not input space

Spans of encoder output are replaced by a learned vector before the Transformer sees them — the BERT recipe, moved one stage later.

Predict a learned discrete inventory

The target is a quantised version of the model's own latent, chosen from codebooks learned during the same training run.

Train the inventory and the context model jointly

One objective, one training run, gradients flowing to the quantiser through a Gumbel softmax.

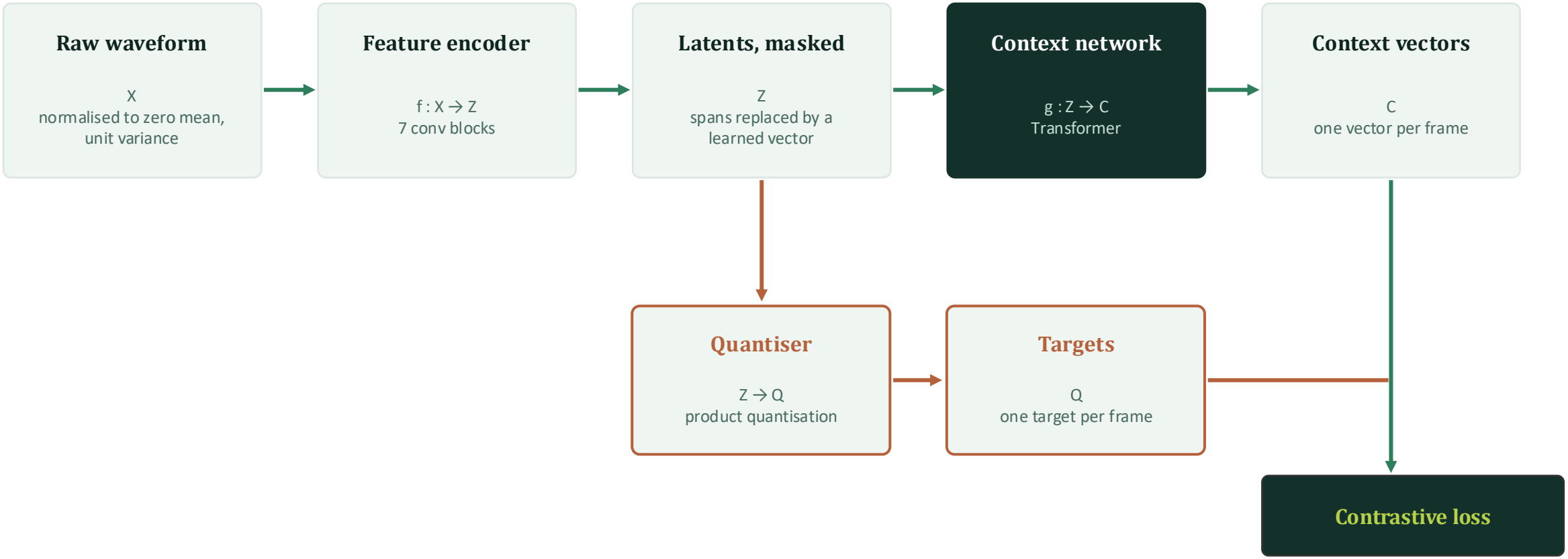
The rest of Part I is these four decisions in detail. Keep them in view — every hyperparameter we are about to meet exists to make one of them work.

2 The model

Feature encoder, context network, quantisation module

The architecture end to end

Three components, two paths, one loss



Read it as two paths from the same latents: the upper path builds context, the lower path builds targets. The loss asks the c ontext vector at a masked position to identify its own quantised latent.

The feature encoder

A learned front end replacing mel filter banks

Structure

Seven blocks, each a temporal convolution followed by layer normalisation and a GELU activation. All blocks have 512 channels.

Strides: (5, 2, 2, 2, 2, 2, 2)
Kernel widths: (10, 3, 3, 3, 3, 2, 2)

The raw waveform is normalised to zero mean and unit variance before it enters.

Why convolutions here

Two jobs at once. First, downsampling: sixteen thousand samples per second is far too many for self-attention, whose cost is quadratic in sequence length.

Second, local feature extraction: the receptive field of the stack is short, so each output summarises a small window — the same job a filter bank does, but learned from data.

- The encoder is trained during pre-training and then frozen. It is not updated at all during fine-tuning.
- For the smaller Librispeech pre-training runs the authors add an L2 penalty on the final encoder layer's activations and scale encoder gradients down by a factor of ten — a stability measure, and a hint that this component is delicate.
- Those runs also use a variant without layer normalisation, normalising the output of the first encoder layer instead of the raw waveform.
- Later analysis found the final convolutional layers correlate strongly with mel spectrogram features: the model rediscovers something close to the hand-engineered front end it replaced.

The arithmetic that sets the model's time resolution

Every downstream design choice inherits these three numbers

320

samples of total stride
(5×2^6) between outputs

20 ms

of audio per output frame
at 16 kHz

49 Hz

output frame rate of the
encoder

Receptive field

400 input samples, or 25 milliseconds of audio, feeds each encoder output. That is close to the standard analysis window used by conventional speech front ends — a 25 ms window with a 10 ms hop.

Why it matters downstream

The 20 ms frame is the atom of everything that follows: the masking spans, the contrastive targets, the CTC output rate, and the temporal granularity of any representation you later extract for a downstream task.

A practical note for anyone extracting features from this model

A three-second emotional utterance from CREMA-D yields roughly 150 frames of 768 or 1024 dimensions. How you pool those frames into an utterance-level vector — mean, attention, or a learned weighted sum across layers — turns out to matter as much as which model you started from. We return to this in the final section.

The context network

A Transformer with one non-standard choice

Standard parts

A Transformer encoder in the usual configuration — self-attention over the whole sequence of latent frames, feed-forward blocks, residual connections, layer normalisation.

Its job is to turn each local 20 ms latent into a contextualised vector that has seen the entire utterance.

The non-standard part

No fixed positional embeddings. Instead a convolutional layer with kernel size 128 and 16 groups is applied to the input; its output, passed through a GELU, is added to the input, and the sum is layer-normalised.

The convolution acts as a relative positional embedding.

- Why relative rather than absolute: speech has no canonical origin. A vowel forty frames into an utterance is not a different kind of object from the same vowel at frame four hundred — only its neighbourhood matters.
- It also removes any maximum sequence length baked into the model, which matters when utterance durations vary from one second to twenty.
- The kernel of 128 frames spans about 2.5 seconds of audio, so the positional signal itself carries a fairly wide window of context.
- Layer dropout is used during pre-training — 0.05 for BASE and 0.2 for LARGE, and none at all for the largest LibriVox run.

The quantisation module

Product quantisation over jointly learned codebooks

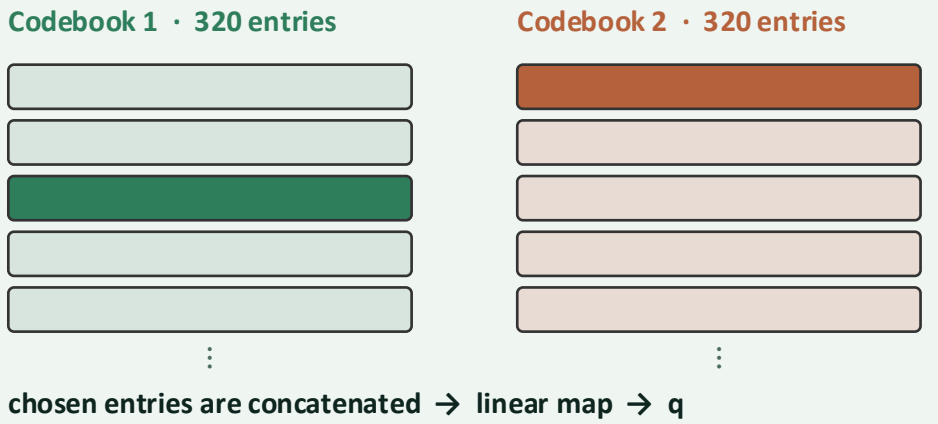
How a target is built

Given G codebooks (or groups), each with V entries, the module picks one entry from each codebook, concatenates the chosen vectors, and applies a linear transformation to produce the target q.

In both BASE and LARGE: G = 2 and V = 320.

Entry size is d/G — 128 for BASE, 384 for LARGE.

Two choices from 320 options each gives a theoretical maximum of 102,400 distinct codewords.



Why product quantisation rather than one big codebook

A single codebook covering a hundred thousand entries would be unwieldy to learn and most entries would go unused. Factoring the choice into two independent selections of 320 gives the same nominal capacity with 640 vectors to train, and lets each codebook specialise.

In practice the number of codewords actually used is far below the theoretical maximum — which is precisely why the objective needs the diversity term we meet in the next section.

Making a discrete choice differentiable

The Gumbel softmax and the straight-through estimator

The problem

Choosing a codebook entry is an argmax — a hard, discrete decision with zero gradient almost everywhere. If the choice is not differentiable, no gradient reaches the codebooks and they never learn.

The solution

The encoder output z is mapped to logits over the G × V entries. Gumbel noise is added, the result is divided by a temperature τ and passed through a softmax.

Forward pass: take the hard argmax. Backward pass: use the true gradient of the softmax. This is the straight-through estimator.

- The temperature τ controls how close the softmax is to a hard one-hot. High temperature means a smooth, high-entropy distribution and informative gradients; low temperature means a near-discrete choice.
- The schedule matters: τ is annealed from 2 down to a minimum of 0.5 for BASE and 0.1 for LARGE, multiplied by 0.999995 at every update.
- So training starts soft — the model explores the codebook — and gradually hardens into genuine discrete selection. It is a curriculum, not a fixed setting.
- This is the machinery that makes 'jointly learned' in the abstract true. Without it the codebook would have to be fitted in a separate stage, which is exactly what vq-wav2vec did.

Two model configurations

Same encoder, different Transformers

	BASE	LARGE
Transformer blocks	12	24
Model dimension	768	1,024
Feed-forward dimension	3,072	4,096
Attention heads	8	16
Parameters	95 million	317 million
Codebook entry size	128	384
Gumbel temperature floor	0.5	0.1
Layer drop	0.05	0.2 (none for LibriVox)
Audio crop per example	250k samples ≈ 15.6 s	320k samples ≈ 20 s
Effective batch	1.6 hours of audio	2.7 hours of audio
Peak learning rate	5×10^{-4}	3×10^{-4}
Pre-training updates	400k	250k on LS-960 · 600k on LV-60k

Both use the same seven-block convolutional encoder, the same masking, and the same quantiser configuration of two codebooks with 320 entries each.

3 Training

Masking, the objective, and fine-tuning

Masking

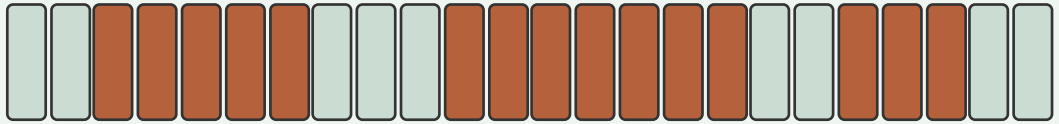
Spans, not individual frames

The procedure

Sample a proportion p of all time steps, without replacement, as starting indices. From each sampled index, mask the following M consecutive steps. Spans may overlap.

Masked frames are replaced by a single trained vector shared across all masked positions.

A stretch of latent frames



masked span visible

Each block is one 20 ms frame

p = 0.065

of steps sampled as starting indices

M = 10

consecutive steps masked from each index

≈ 49%

of all time steps end up masked

299 ms

mean span length — 14.7 frames

A 299 ms span is roughly a syllable. Masking individual 20 ms frames would be trivial — neighbouring frames are nearly identical — so the span length is what makes the task require phonetic structure rather than interpolation.

The contrastive loss

Identify your own quantised latent among distractors

The task

For a masked time step t, take the context vector c_t. The model must identify the true quantised latent q_t among a candidate set containing q_t and K distractors.

Distractors are sampled uniformly from other masked time steps of the same utterance.

Similarity is cosine similarity between c_t and each candidate, divided by a temperature κ, and the loss is the negative log of the softmax over candidates.

The settings

K = 100 distractors.

Temperature κ = 0.1.

Cosine similarity rather than a dot product, so the loss is insensitive to vector magnitude.

The checkpoint used is the one with the lowest contrastive loss on the validation set — the objective itself is the model-selection criterion.

- Why distractors from the same utterance: it makes the task hard in exactly the right way. Distractors from a different speaker or recording could be rejected on timbre alone, and the model would learn to be a speaker classifier.
- Because negatives share the speaker, the channel and the recording session, the only thing that distinguishes the true target is its phonetic content. The sampling scheme is doing quiet but essential work.
- This is noise-contrastive estimation in the InfoNCE form used by contrastive predictive coding, applied to masked positions rather than future ones.
- The prediction is not of a raw latent but of its quantised form, which is what makes the candidate set a set of discrete units rather than a set of arbitrary vectors.

The diversity loss

A regulariser that stops the codebook from collapsing

The failure it prevents

Nothing in the contrastive objective requires the model to use many codebook entries. A quantiser that maps everything to a handful of codewords would still let the contrastive task be solved, and it is the path of least resistance.

The result would be an inventory with almost no capacity to distinguish speech sounds.

The mechanism

Maximise the entropy of the averaged softmax distribution over codebook entries, computed across a batch of utterances, for each codebook separately.

This distribution excludes the Gumbel noise and the temperature. The implementation maximises perplexity, which is equivalent.

Crucially it is an average over the batch, not per frame.

Why averaging over the batch is the right choice

Per-frame entropy would push every individual frame toward an ambiguous, spread-out code — the opposite of what a discrete inventory should do. Batch-averaged entropy asks only that all entries get used *somewhere*, leaving each individual frame free to commit sharply to one codeword.

Total objective: $L = L_m + \alpha L_d$, with $\alpha = 0.1$. One tuned scalar balances a task term against a regulariser — a reminder that the contrastive loss alone is not enough.

The design decision that matters most

Continuous representations in, discrete representations out

INPUT to the Transformer

Continuous

Unquantised latents retain more information, which lets the context network build richer representations of the surrounding speech.

TARGET in the loss

Quantised

Discretising the target removes the fine detail that would otherwise let the model succeed by matching speaker and channel characteristics.

This is the whole argument against the two-stage predecessors

vq-wav2vec and Discrete BERT quantise on the input side as well, so the context model only ever sees a lossy, discrete view of the audio. wav2vec 2.0 keeps the input rich and the target coarse.

The paper's justification, in its own terms: continuous latents retain more information and enable better context representations, while quantised targets lead to more robust training. Continuous targets let the model capture detailed artefacts of the current sequence — speaker, background — which make the task easier and prevent the model from learning general representations.

Section 6 puts numbers on this. It is worth roughly a third of the word error rate.

Fine-tuning for speech recognition

Where labels finally enter

Output layer	A randomly initialised linear projection is added on top of the context network. For Librispeech the target set is 29 tokens for character targets plus a word boundary token.
Loss	Connectionist temporal classification. No alignment between audio frames and characters is needed — CTC marginalises over all valid alignments.
Frozen encoder	The convolutional feature encoder is not trained at all during fine-tuning. Only the Transformer and the new output layer are updated.
Warm start	For the first 10,000 updates only the output classifier is trained; the Transformer is held fixed and only afterwards begins to update.
Schedule	Adam with a tri-state rate: warm up over the first 10% of updates, hold constant for the next 40%, then decay linearly for the remainder.
Augmentation	A modified SpecAugment masking both time steps and channels. The paper reports this delays overfitting and significantly improves error rates, especially on the tiny labelled subsets.

What it cost to train

A note on who can actually do this

64

V100 GPUs for BASE,
for 1.6 days

128

V100 GPUs for LARGE,
2.3 days on Librispeech

5.2 d

on 128 GPUs for LARGE
on the LibriVox audio

600k

updates for the largest
pre-training run

The asymmetry this creates

Pre-training is out of reach for most academic groups. Fine-tuning is not — a released checkpoint plus a single GPU and ten minutes of labelled audio reproduces the headline result.

Which is why the release mattered

The authors released code and checkpoints in fairseq. Almost all subsequent use of wav2vec 2.0 — including every speech-emotion paper we will see later — starts from those weights rather than repeating the pre-training.

Worth sitting with: a paper whose stated motivation is making speech technology available for under-resourced languages requires a hundred and twenty-eight datacentre GPUs to produce. The democratising effect is real, but it runs through model release, not through method accessibility.

Part I in five sentences

Before the break

1. A convolutional encoder turns the raw waveform into 20 ms latent frames at 49 Hz, learning its own front end.
2. Roughly half of those frames are masked in contiguous spans averaging 299 ms — about a syllable — and replaced by one learned vector.
3. A Transformer with convolutional relative positional embeddings turns the partially masked sequence into contextualised vectors.
4. In parallel, the unmasked latents are quantised into a jointly learned inventory of two codebooks with 320 entries each.
5. The model is trained to pick out the correct quantised latent for each masked position from 100 same-utterance distractors, with a diversity term keeping the codebook in use.

4 Experimental setup

Data, splits, and decoding

Data

Unlabelled audio for pre-training, labelled subsets for fine-tuning

Unlabelled

LS-960 — the 960 hours of Librispeech audio, with transcriptions discarded.

LV-60k — audio from LibriVox, which after the Libri-light preprocessing yields 53.2 thousand hours.

Both are read audiobook speech. That homogeneity matters and we return to it under limitations.

Labelled fine-tuning settings

Five settings, spanning three orders of magnitude:

960 hours (all of Librispeech) · 100 hours (train-clean-100) · 10 hours · 1 hour · 10 minutes.

The last three are the Libri-light limited-resource subsets, evaluated under the Libri-light protocol.

Evaluation

The standard Librispeech dev-clean, dev-other, test-clean and test-other sets throughout, so numbers are comparable with the wider ASR literature.

A second task

TIMIT phoneme recognition — five hours of audio with detailed phoneme labels, standard splits, phones collapsed to the usual 39 classes.

Note what a well-designed benchmark looks like: the same pre-trained model is evaluated across five labelled-data regimes with published protocols. Compare this with the split-invention problem we found in both IEMOCAP and CREMA-D.

Language models and decoding

The part of the pipeline that quietly moves the numbers

Two language models

A 4-gram model, and a Transformer language model with 20 blocks, model dimension 1,280, inner dimension 6,144 and 16 attention heads. Both are trained on the Librispeech language-model corpus.

Tuned, not fixed

The language-model weight and the word insertion penalty are tuned by Bayesian optimisation over 128 trials, selected on dev-other. Test decoding uses beam 1,500 for the n-gram model and beam 500 for the Transformer.

A vocabulary mismatch

The acoustic model outputs characters; the language model operates over words. The authors flag this themselves as delaying feedback from the language model and likely hurting results — most contemporaneous work used word pieces for both.

Why to keep this in view while reading the results tables

Most headline numbers use the Transformer language model. A large external language model can recover a great deal of accuracy from a weak acoustic model, so the reported word error rates are a property of the whole pipeline, not of the learned representation alone.

The paper is upfront about this and reports results with no language model and with the n-gram model in an appendix. When you compare across papers, check which decoding configuration produced each number — the same model can differ by several points.

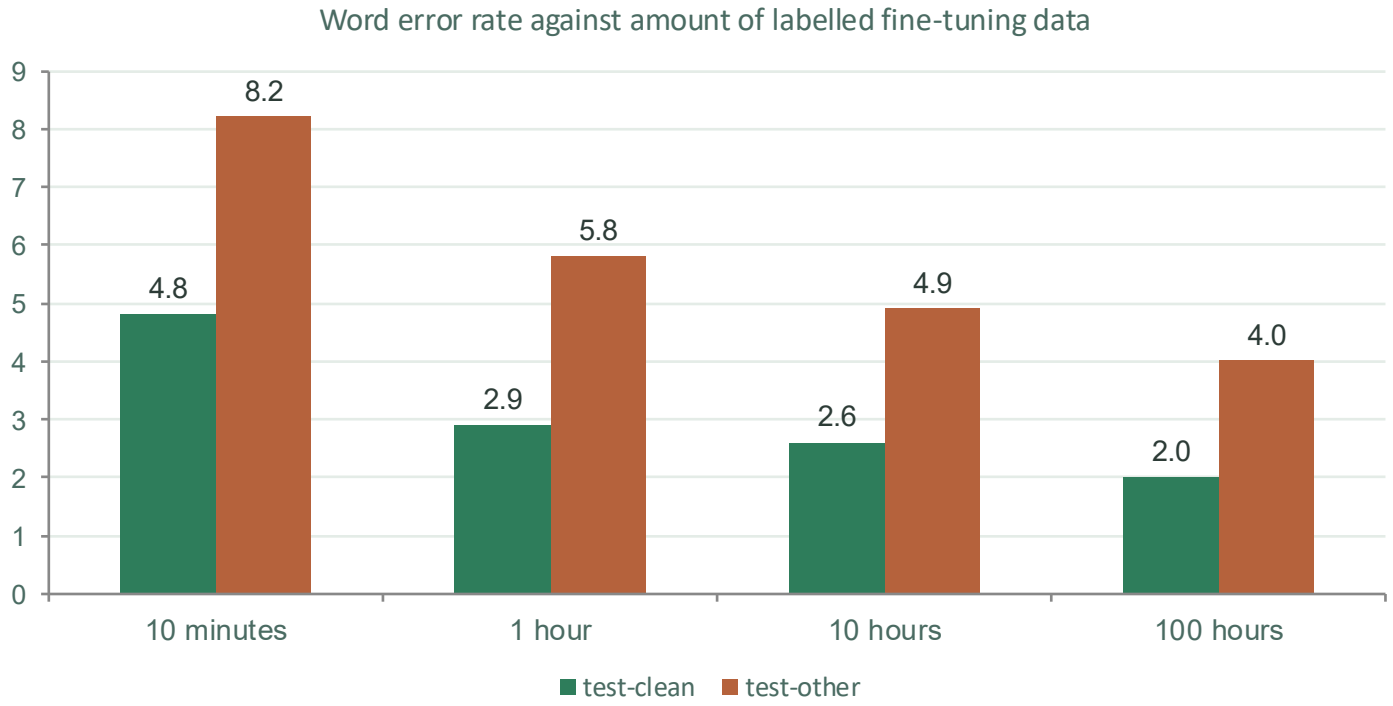
This is the same lesson as the split problem in the corpus seminars, in a different costume: the reported number depends on choices that are not the contribution being claimed.

5 Results

Low-resource, high-resource, and phoneme recognition

Low-resource results

LARGE pre-trained on LV-60k, decoded with the Transformer language model



- Ten minutes to one hour is the steepest jump — a 40% relative reduction on test-clean for six times the data.
- From one hour to a hundred, a hundredfold increase in labels, buys less than three points on test-other.
- The curve is flattening long before the labelled data runs out. That shape is the paper's real argument.
- test-other is consistently roughly double test-clean. The noisy condition remains much harder at every data scale.

Read the shape, not the individual bars: most of what supervised ASR was buying with thousands of transcribed hours can be obtained from unlabelled audio instead.

The ten-minute result, unpacked

What forty-eight recordings actually means

48

recordings in the entire labelled training set

12.5 s

average length of each recording

4.8 / 8.2

WER on test-clean and test-other

For comparison

Discrete BERT, the two-stage predecessor, reaches 16.3/25.2 in the same ten-minute setting — though with a 4-gram rather than a Transformer language model, and pre-trained on LS-960 rather than the larger LibriVox set.

wav2vec 2.0 BASE on the same LS-960 audio with the same 4-gram model

What made the difference

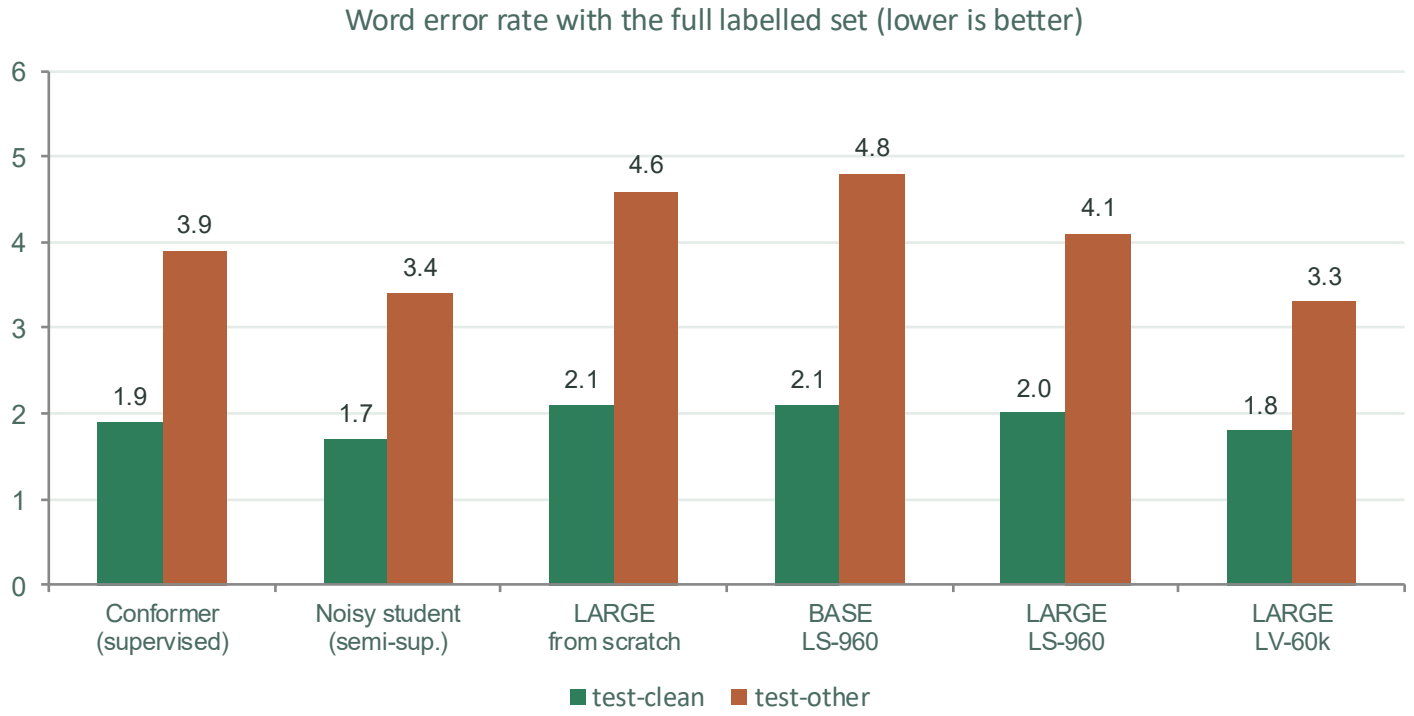
That BASE-versus-Discrete-BERT comparison is the cleanest one available: same unlabelled data, same language model, same labelled subset. The word error rate falls by roughly a third, and the only substantive change is joint end-to-end learning with continuous inputs.

Everything beyond that comes from scale — a bigger model, more

Be precise when quoting this result: the headline 4.8/8.2 combines the method, the LARGE model, 53.2 thousand hours of unlabelled audio and a Transformer language model. Attributing all of it to the objective would be an overstatement — the paper does not make that mistake, and neither should we.

High-resource results

Fine-tuning on all 960 labelled hours of Librispeech



- 1.8/3.3 sets a new state of the art on test-other, the noisy condition.
- The same architecture trained from scratch on the same labels reaches only 2.1/4.6 — so pre-training helps even when no labels are missing.
- Note the from-scratch baseline is weaker than Conformer at 1.9/4.1. The gain comes from pre-training, on top of a worse starting architecture.
- BASE with pre-training roughly matches LARGE without it — a useful way to think about what unlabelled data is worth in parameters.

Self-supervision is not only a low-resource trick. It improves the fully supervised setting too.

The authors' own caveats

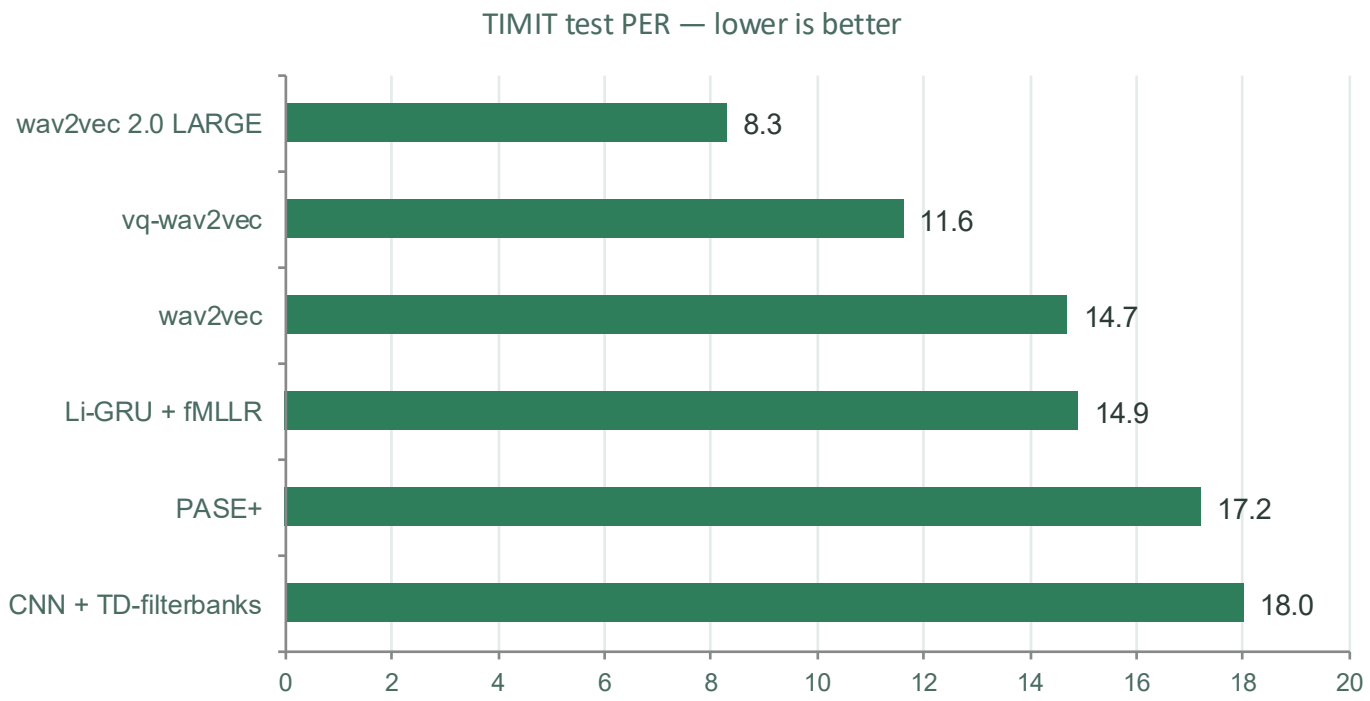
A short list of things they say could be better

- **Weaker architecture** — A plain Transformer with a CTC loss, which does not perform as well as sequence-to-sequence models. They expect gains from switching.
- **Character vocabulary** — The acoustic model predicts characters while the language model works over words, delaying language-model feedback. Word pieces would likely help.
- **No data balancing** — The comparison method applies data balancing during training; this work does not, and the authors note the omission.
- **Self-training left on the table** — They observe that self-training is probably complementary to pre-training and that combining the two may do better still.

All four predictions turned out to be right, and all four were taken up within about a year.

Phoneme recognition on TIMIT

A second task, no language model



A new state of the art

8.3% test PER, against 7.4% on dev — a relative reduction of 23% on dev and 29% on test over the next best result, which was vq-wav2vec.

Why this result matters

TIMIT has five hours of data and phoneme-level labels. Doing well here says the representation encodes phonetic structure directly, not just enough word-level regularity to satisfy a language model.

The progression along this chart is the progression of the wav2vec programme itself: 14.7 for wav2vec, 11.6 for vq-wav2vec, 8.3 here. Three papers, roughly halving the error.

A small inconsistency worth noticing

An exercise in reading papers carefully

The abstract says

Using just ten minutes of labelled data and pre-training on 53k hours of unlabelled data still achieves 4.8/8.2 WER.

Table 1 says

LARGE, LV-60k, Transformer language model, 10 minutes labelled: dev 4.6/7.9, test 4.8/8.2.

Consistent with the abstract.

But the running text says

In section 5.1, the same configuration is described as achieving 5.2/8.6 on the clean and other test sets.

That figure appears nowhere in the table.

What to make of it

Almost certainly a leftover from an earlier version — the paper went through three arXiv revisions between June and October 2020, and the first version's abstract quotes different numbers again. The prose was not fully updated when the tables were.

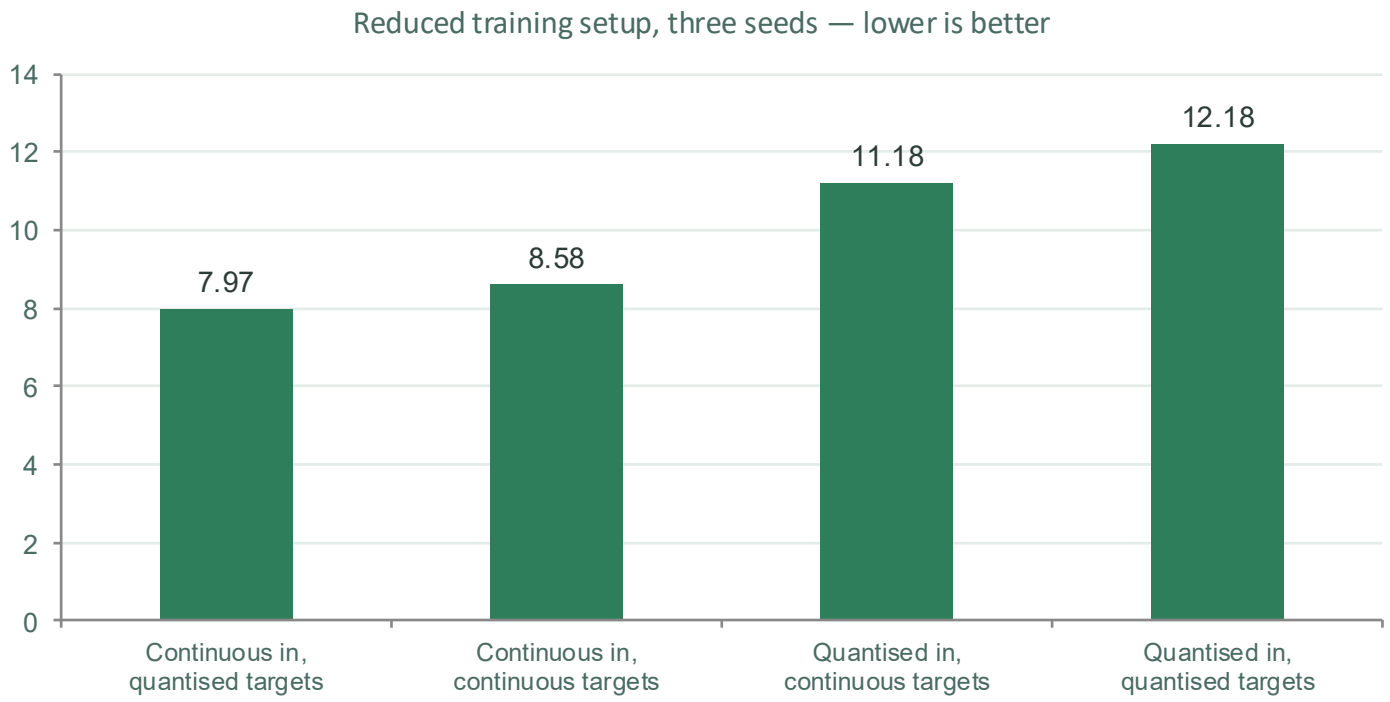
Nothing substantive turns on it. But it is a useful reminder for anyone citing this work: quote Table 1, not the sentence in section 5.1, and say which you used. Small discrepancies like this propagate through secondary literature for years, and a seminar is exactly the place to catch them.

6 Ablations

Which design decision is actually doing the work

The quantisation ablation

Four combinations of continuous and discrete, on inputs and targets



- The configuration wav2vec 2.0 actually uses is the best of the four.
- Quantising the input costs about three points of WER — this is the two-stage penalty that vq-wav2vec and Discrete BERT pay.
- Standard deviations across seeds: 0.02, 0.08, 0.16 and 0.41 respectively. The worst configuration is also the least stable.
- Note this is a reduced setup — BASE, 250k updates, fine-tuned on 10 hours — so the absolute numbers are not comparable with the main tables. The ordering is the result.

This single table is the empirical justification for the architectural change that separates this paper from its predecessors — and it is the only ablation in the main body.

Why an easier pretext task gives worse representations

The most instructive number in the paper

62%

training accuracy at identifying the correct latent — with quantised targets

78%

training accuracy with continuous targets — and worse downstream WER

The pretext task got easier and the representation got worse

With continuous targets the model solves its own training task substantially better — and transcribes speech substantially worse. The authors' explanation: continuous targets capture detailed artefacts of the current sequence, such as speaker identity and background, which make the task easier and prevent general representations.

The model finds a shortcut. Matching a continuous target can be done on timbre alone; matching a discrete one cannot, because speakers saying the same sound share a codeword.

The lesson generalises: pretext-task accuracy is not a measure of representation quality, and can be inversely related to it.

7

What do the representations encode?

Evidence from the paper, and from the interpretability work that followed

Do the learned discrete units correspond to phonemes?

The question the quantiser makes it possible to ask

What the paper offers

An appendix analysis relating the learned discrete latent speech representations to phonemes, alongside the TIMIT result that shows the representation supports phoneme recognition after very little fine-tuning.

The inventory is not designed to be phonemic — nothing in the objective mentions phonemes — so any alignment is emergent.

Why this is interesting beyond ASR

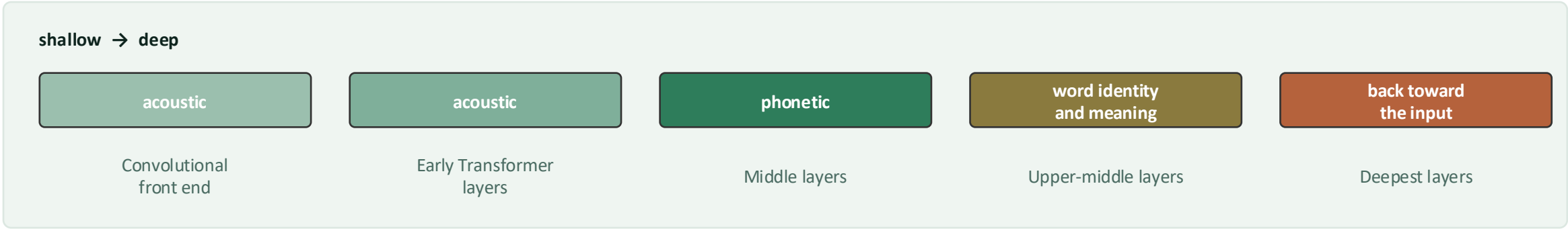
A discrete inventory learned from raw audio is a candidate model of how a listener might acquire sound categories without supervision.

It is also the direct ancestor of the discrete speech units used in textless language modelling and in the token vocabularies of modern speech language models.

- The alignment is partial, not clean. There are far more codewords in use than there are phonemes, and the correspondence is many-to-one rather than one-to-one.
- Codewords also carry information the phoneme label does not — context, position within a segment, and some speaker and channel characteristics.
- For our purposes, that residue is the interesting part: information the ASR objective treats as nuisance is exactly the information an affective-computing task wants.
- Be careful with the strong claim. 'The model discovers phonemes' is an overstatement of what this analysis supports; 'the units are phonetically structured' is defensible.

What each layer encodes

From the interpretability work that followed — Pasad, Chou & Livescu (2021)



- The Transformer layers behave like an autoencoder: representations move away from the input as depth increases, then move back toward it in the deepest layers, as if reconstructing what they started from.
- That reversal is a direct consequence of the objective — the model is trained to predict a quantised version of its own encoder output, so the last layers are pulled back toward that output.
- In between, the layer-wise evolution tracks the linguistic hierarchy: acoustic, then phonetic, then word identity and word meaning, in that order.
- The final convolutional layers and earliest Transformer layers correlate strongly with mel spectrogram features — the model relearns something like a conventional front end.
- Fine-tuning for speech recognition breaks the autoencoder behaviour, especially in the last few layers, which then become much better at encoding word identity.

The practical consequence

The last layer is usually the wrong layer

The habit

Take the final hidden state, pool it, put a classifier on top. Inherited from image models, where the deepest layer is the most abstract.

Why it fails here

In a pre-trained wav2vec 2.0 the deepest layers are drifting back toward the encoder output. The most linguistically abstract information sits in the middle of the stack, not the top.

What to do instead

Learn a weighted sum over all layers, with the weights trained jointly with the downstream head. This is now standard practice in speech benchmark suites.

Three things this implies for anyone using these models

First, always try layer aggregation before concluding a pre-trained model is unsuited to your task — the difference is often larger than the difference between models.

Second, the learned layer weights are themselves an interpretability result: they tell you where in the linguistic hierarchy your task lives.

Third, an ASR-fine-tuned checkpoint is not strictly better than a purely pre-trained one. Fine-tuning specialises the upper layers toward word identity; if your task is not transcription, that may be actively harmful.

8 Limits, legacy, and affective computing

Where the paper falls short, what it started, and what it means for our work

Limitations

Some acknowledged, some structural, some visible only with hindsight

One domain of audio

Both unlabelled sets are read audiobook speech: clean, fluent, mostly North American, largely a narrow demographic of volunteer readers. Nothing here is conversational, noisy or emotional.

One language

Everything is English. The multilingual claim in the broader-impact section is a hope, tested only by later work.

Evaluated almost entirely on ASR

Two tasks, both transcription-shaped. The general-purpose claim implied by 'speech representations' is not tested in the paper.

A single main ablation

One table isolating quantisation. Masking parameters, distractor count, codebook size and the diversity weight are explored only in the appendix.

Results are pipeline results

Headline word error rates involve a large Transformer language model. The contribution of the representation alone is harder to read off.

Compute inaccessibility

The method cannot be reproduced without a hundred-plus GPUs for days. Verification depends on trusting released checkpoints.

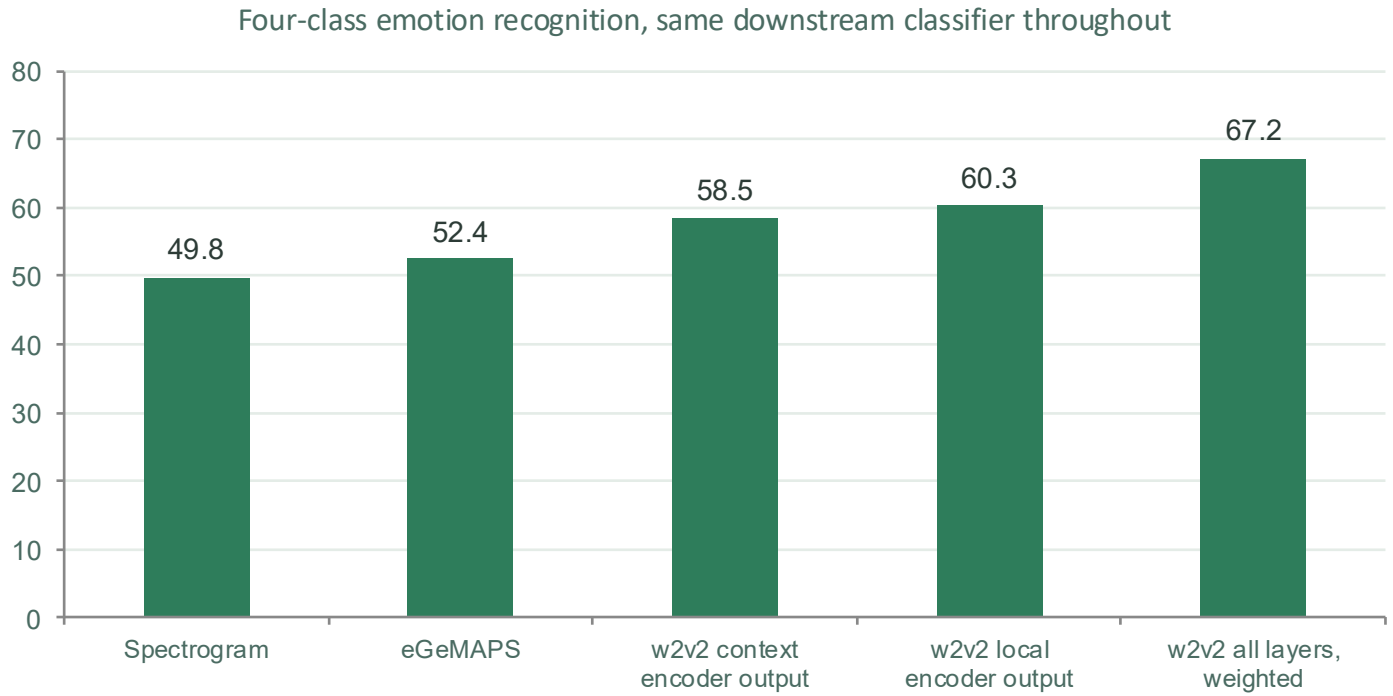
What came next

Five years of direct descendants

HuBERT (2021)	Replaces the contrastive loss with masked prediction of offline k-means cluster targets, refreshed over iterations. Same architecture, simpler and more stable objective; now often preferred.
WavLM (2022)	Adds simulated overlapped and noisy speech to pre-training, targeting the full stack of speech tasks rather than recognition alone — including speaker and paralinguistic tasks.
XLSR-53 and XLS-R	The multilingual continuation, pre-training on dozens and then over a hundred languages, and delivering on the broader-impact promise made here.
data2vec (2022)	From the same group: predict contextualised latent targets from a teacher network, with one recipe shared across speech, vision and text.
Whisper (2022)	The counter-argument — large-scale weakly supervised training on 680k hours of paired audio and text. A reminder that self-supervision was not the only route to robustness.

Back to emotion

wav2vec 2.0 features on IEMOCAP — Pepino, Riera & Ferrer (2021)

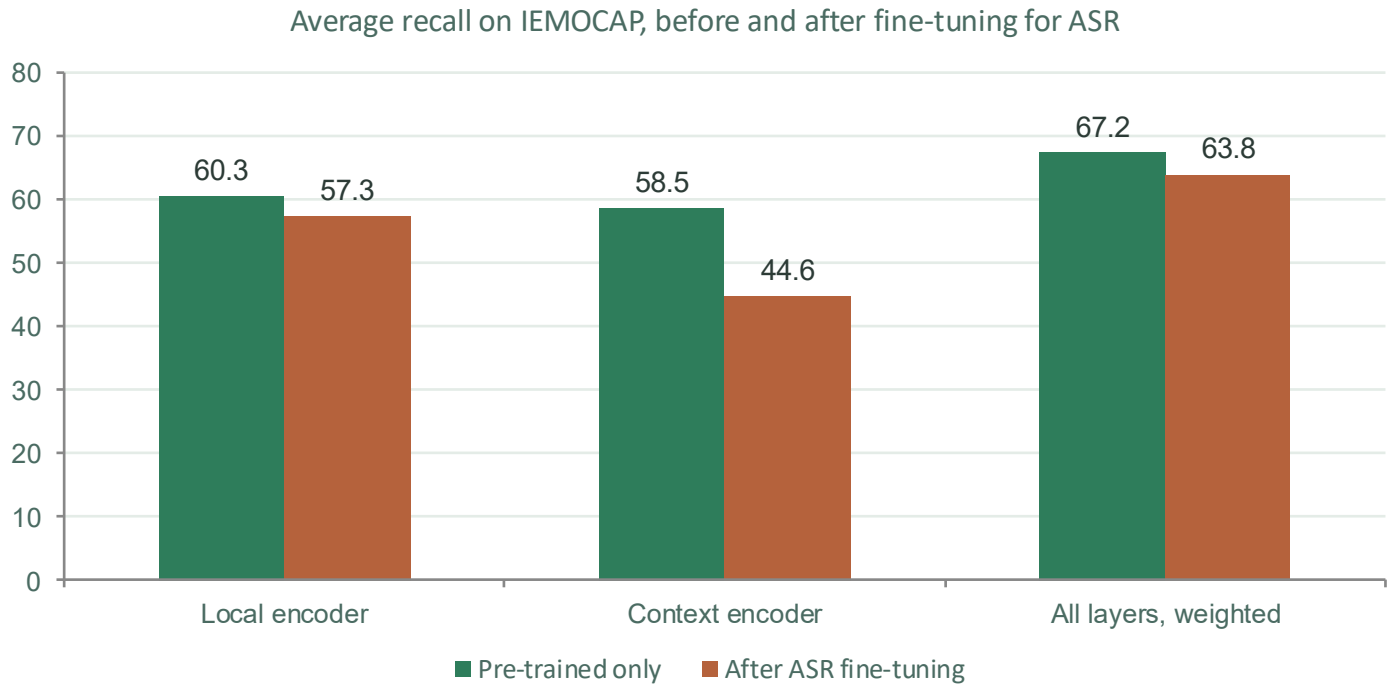


- The classifier is held constant; only the input features change. This is a representation comparison, not a modelling contest.
- Even the local encoder output — a 25 ms window, no context at all — beats hand-crafted eGeMAPS features designed specifically for paralinguistics.
- Aggregating across layers with learned weights beats any single layer by a wide margin, exactly as the layer-wise analysis predicts.
- On RAVDESS the same weighted-layer approach reaches 84.3% average recall, against 57.0% for eGeMAPS.

A model trained on audiobooks, with an objective that never mentions emotion, produces better emotional representations than features designed by phoneticians for exactly this task.

The uncomfortable finding

Fine-tuning for speech recognition damages emotional information



- Every configuration is worse after ASR fine-tuning. The context encoder output collapses by nearly fourteen points.
- This is the design working exactly as intended. Transcription treats speaker state as nuisance, so fine-tuning removes it.
- Which means an ASR-fine-tuned checkpoint is the wrong default for paralinguistic work — a mistake still made routinely, because those checkpoints are the most downloadable.
- It also reframes what the objective is doing: the discretisation bottleneck we admired in section 6 is discarding precisely what our field wants to keep.

The property that makes wav2vec 2.0 an excellent speech recogniser is in tension with the property that would make it an excellent emotion recogniser. That tension is a research programme, not a footnote.

Three papers, one argument

Pulling the seminar series together

IEMOCAP asked

How do we elicit genuine emotion, and how do we know what a recording contains?

Answer: dyadic interaction, and labels from human perception rather than the actor's intent.

CREMA-D asked

How do we get enough speakers, and how do we know which channel carries the signal?

Answer: ninety-one actors, fixed sentences, and every clip rated three separate ways.

wav2vec 2.0 asked

What if the bottleneck was never the labels at all?

Answer: learn the structure of speech from unlabelled audio, and use the scarce labels only to attach a symbol set.

What the three together imply

The corpus papers show that emotional labels are noisy, expensive and shaped by design choices that most downstream users never examine. This paper shows that the acoustic modelling problem can be largely solved without labels.

Put together, they say something specific about where the remaining difficulty in our field lives: not in learning representations of speech, which is now close to a solved and downloadable problem, but in defining what we are trying to predict, and in evaluating it honestly on data whose construction we understand.

Take-aways

Five things to carry out of this seminar

1	**The bottleneck matters more than the objective**	Continuous inputs with quantised targets is the whole architectural argument, and the ablation shows it is worth roughly a third of the error. The contrastive loss itself was later replaced.
2	**Pretext accuracy is not representation quality**	Sixteen points better at its own training task, several points worse at transcription. In self-supervised learning these can move in opposite directions.
3	**Design your negatives**	Sampling distractors from within the same utterance is what forces the model to use phonetic rather than speaker information. An implementation detail doing conceptual work.
4	**The last layer is usually the wrong layer**	Pre-trained wav2vec 2.0 layers behave like an autoencoder. Aggregate across layers before concluding a model does not suit your task.
5	**Match the checkpoint to the task**	ASR fine-tuning measurably damages paralinguistic information. For emotion, speaker or health tasks, start from the pre-trained model, not the fine-tuned one.

Questions for discussion

1. The quantisation bottleneck works by throwing away speaker and channel information. If you were designing a self-supervised objective for emotion rather than transcription, what would you throw away instead — and what would the target inventory be over?

2. Pre-training used fifty-three thousand hours of read audiobook speech. How much of what these representations know is a property of speech, and how much is a property of people reading books calmly into a microphone?

3. Making the pretext task easier made the representations worse. Is there a principled way to know in advance which shortcuts to block, or is this necessarily empirical?

4. The broader-impact section argues this work makes speech technology available for under-resourced languages — but reproducing it needs a hundred and twenty-eight datacentre GPUs. Is releasing checkpoints an adequate answer to that tension?

5. Given that ASR fine-tuning damages emotional information, and that IEMOCAP and CREMA-D are small, noisy and without official splits: what would a genuinely convincing speech-emotion result look like in 2026?

Baevski, Zhou, Mohamed & Auli (2020), NeurIPS 33, 12449–12460 · arXiv:2006.11477 · layer analysis: Pasad, Chou & Livescu (ASRU 2021) · emotion results: Pepino, Riera & Ferrer (Interspeech 2021)