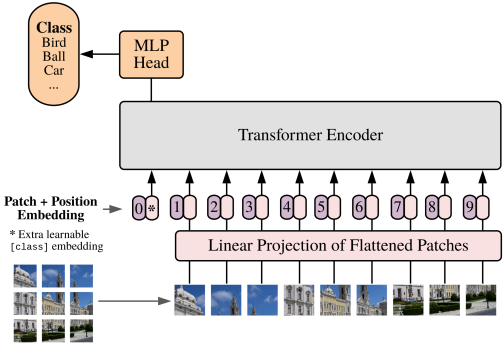
Vision Transformer (ViT)

ViT splits an image into fixed-size patches, embeds them as a sequence of tokens, and feeds them to a convolution-free Transformer encoder to learn scalable visual representations.



∗ ViT-L/16 : $N = 16\times16$ patches, 307M parameters

$L = 24$ layers, $d_m = 1024$, $h = 16$

(a) Split an image $\boldsymbol{x} \in \mathbb{R}^{H\times W\times C}$ into flattened patches $[\boldsymbol{x}_p^1; \cdots; \boldsymbol{x}_p^N] \in \mathbb{R}^{N\times P^2\cdot C}$ with patch size $P\times P$ and $N = HW/P^2$.

(b) Linear projection of each patch (token) into a learnable patch+position embedding $\boldsymbol{z}_0^n = \boldsymbol{x}_0^n W + PE_n$ with $W \in \mathbb{R}^{P^2\cdot C\times d_m}$ with $\boldsymbol{z}_0^0 = \boldsymbol{x}_{\text{class}} + PE_0$ for classification task.

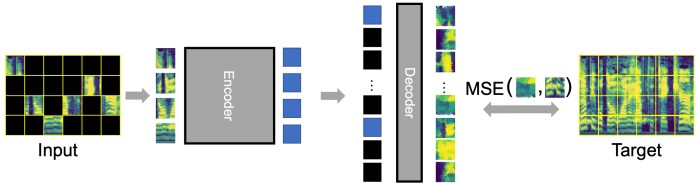
(c) Feed the token sequence $\boldsymbol{z}_0 = [\boldsymbol{z}_0^0; \cdots; \boldsymbol{z}_0^N]$ to a pure Transformer encoder.

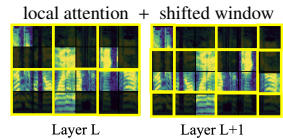
(d) The output of the Transformer encoder $\boldsymbol{z}_L^0$ after processing a classification MLP head is an image representation $\hat{\boldsymbol{y}}$.

An Image is Worth 16×16 Words: Transformers for Image Recognition at Scale, Google, ICLR 2021

Audio-MAE

Audio-MAE (Audio Masked AutoEncoder) is a Transformer encoder-decoder based self-supervised audio representation pre-training framework employing a reconstruction loss.





(a) Transform audio inputs into spectrograms and split them into patches.
(b) Flatten and linearly project each patch, and add a fixed position embedding.
(c) Mask spectrogram patches (80% due to high redundancy).
(d) Encode only unmasked patches (20%) using 12-layer ViT-Base.
(e) Decode the order-restored encoded context padded with mask tokens to reconstruct the input, using local attention (most bottom layers) + global attention (few top layers) for audio spectrograms.
(f) Minimize MSE between the masked portion of the reconstruction and the input spectrogram.

Masked Autoencoders that Listen, Meta AI, NIPS 2022

BEATs

BEATs (Bidirectional Encoder representation from Audio Transformers) is Transformer-based self-supervised general audio representation pre-training framework, which optimizes an acoustic tokenizer and an audio SSL model by iterations.

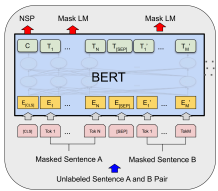
- Acoustic tokenizer quantizes continuous audio/acoustic features (spectrograms) from unlabeled audio into patch-level semantic-rich discrete labels (tokens).
- Audio SSL model (ViT) employs a discrete label prediction loss (labels from the tokenizer), rather than using the audio feature reconstruction loss of previous methods (Audio-MAE).
- This joint iterative pipeline (up to 3 iterations) helps the model capture high-level semantics by having the converged SSL model teach the tokenizer via knowledge distillation.
- While previous speech SSL models (Wav2vec, HuBERT, and WavLM) excel at speech tasks, they struggle with the diverse general audio domain (voices, environmental sounds, and music).

BEATs bridges this gap using less training data and fewer model parameters (90M ~ Wav2vec 2.0 or Audio-MAE, and tokenizer 8 V100 45hrs + SSL model 16 V100 75hrs).

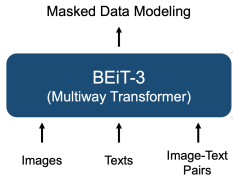
BEATs: Audio Pre-Training with Acoustic Tokenizers, Microsoft, ICML 2023

Background: SSL models with discrete label prediction

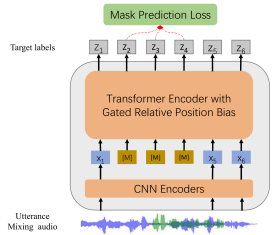
- Transformer-based self-supervised learning (SSL) has achieved great success in language, vision, speech, and audio domains.
- SSL with discrete label prediction loss is adopted for language, vision, speech modalities, and shows better performance than feature reconstruction loss.
- Compared with reconstruction loss, semantic-rich discrete label prediction encourages the SSL model to abstract the high-level semantics and discard the redundant details.



BERT/GPT
for language



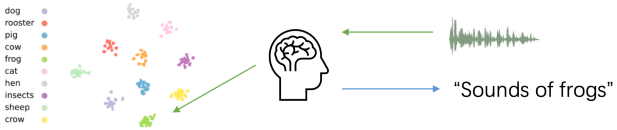
BEiT series
for vision and vision-language



HuBERT/WavLM
for speech

Benefits of discrete label prediction for audio pre-training

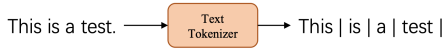
- Previous audio SSL models (Audio-MAE) relying on the reconstruction of audio features (spectrograms), overlook high-level audio semantics (abstract audio meaning).
- Human-inspired semantics: Humans understand audio by extracting and clustering the abstract high-level semantics instead of focusing on the low-level time-frequency details.



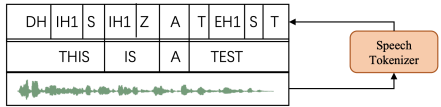
- Efficiency: By targeting semantic-rich tokens and discarding the redundant details, it results in a superior audio understanding with a lower pre-training cost.
- Unification: Advances the unification of language, vision, speech, and audio pre-training, which enables building a foundation model across modalities with a single pre-training task.

How to design the semantic-rich acoustic tokenizer?

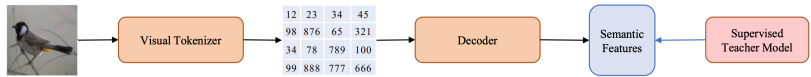
- Text tokenizer



- Speech tokenizer



- Visual tokenizer



- Audio property:
  - Continuous signals and various durations
  - Wide variations of environmental events (human voices, nature sounds, and music)

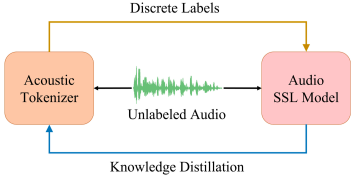
Human speak | Dog bark | Machine operating in a factory | Wind blow while bird singing

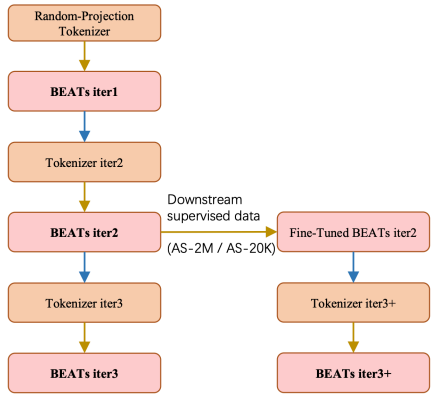
BEATs: Transformer-based iterative audio pre-training framework

- Given unlabeled audio inputs (raw waveforms), extract the corresponding spectrograms, split them into $N = 16 \times 16$ patches, and flatten them to the patch vectors $\boldsymbol{X} = \{\boldsymbol{x}_n\}_{n=1}^{N}$.

Acoustic tokenizer and audio SSL model are optimized by (up to 3) iterations; in each iteration:

- Generate patch-level discrete labels $\hat{Z} = \{\hat{z}_n\}$ for the patches $\{\boldsymbol{x}_n\}$ using the acoustic tokenizer.
- Train the pre-trained/fine-tuned audio SSL model using a discrete label prediction loss between the output $\hat{\boldsymbol{O}} = \{\hat{\boldsymbol{o}}_n\}$ and the vectors related to $\hat{Z}$.
- Update the tokenizer via knowledge distillation, using the converged SSL model as a teacher.



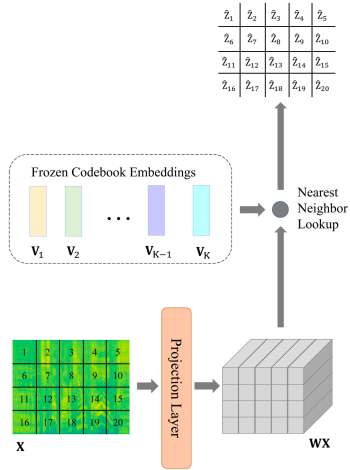


Acoustic Tokenizer

- Acoustic tokenizer converts continuous unlabeled audio features into patch-level discrete labels for each iteration.
- It has two types:

(1) Random-Projection Tokenizer for the first iteration
  * Frozen (non-learnable) due to no teacher model yet

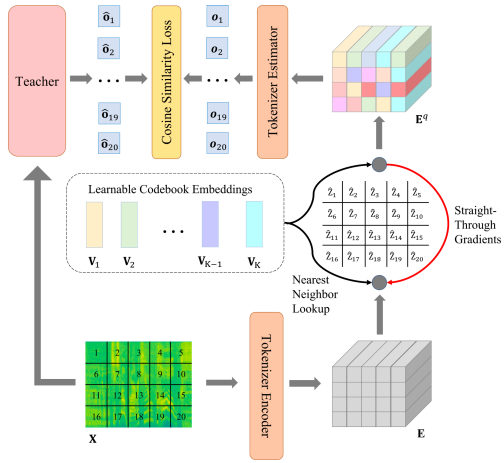
- Linear Projection: Flattened spectrogram patches $\boldsymbol{x}_n$ are projected to $W\boldsymbol{x}_n$ with a randomly initialized weight W.
- Codebook Lookup: Find the nearest neighbor vector in a randomly initialized frozen codebook $\boldsymbol{V} = \{\boldsymbol{v}_k\}_{k=1}^{K}$.
  * codebook $\boldsymbol{V}$: $K = 1024$ embeddings with 256 dimension
- Discrete Label: Assign $\hat{z}_n = \arg\min_k \|\boldsymbol{v}_k - W\boldsymbol{x}_n\|_2^2$.



(2) Self-Distilled Tokenizer for 2nd/3rd iterations

- Teacher: Converged SSL model from previous iteration.
- (1) Tokenizer Encoder: 12-layer Transformer encoder mapping $\boldsymbol{X}=\{\boldsymbol{x}_n\}_{n=1}^N$ to encoded vectors $\boldsymbol{E}=\{\boldsymbol{e}_n\}$.
  (2) Vector Quantization: Nearest-neighbor lookup $\hat{z}_n = \arg\min_k \|\ell_2(\boldsymbol{v}_k)-\ell_2(\boldsymbol{e}_n)\|_2^2$ from $\boldsymbol{V}=\{\boldsymbol{v}_k\}_{k=1}^K$ with a normalization ℓ_2 to unit length, and then get the quantized vectors $\boldsymbol{E}^q=\{\boldsymbol{v}_{\hat{z}_n}\}$.
  (3) Tokenizer Estimator: 3-layer Transformer taking $\boldsymbol{E}^q$ as the input, and trained to predict the SSL teacher model's outputs $\hat{\boldsymbol{O}}=\{\hat{\boldsymbol{o}}_n\}$.
- Straight-Through Gradients: Bypass quantization's non-differentiability by copying gradients from $\boldsymbol{E}^q$ back to $\boldsymbol{E}$ during backpropagation.
- Training objective: For the pre-training dataset $\mathcal{D}$, tokenizer's outputs $\{\boldsymbol{o}_n\}$, stopgradient operator $sg[\cdot]$,

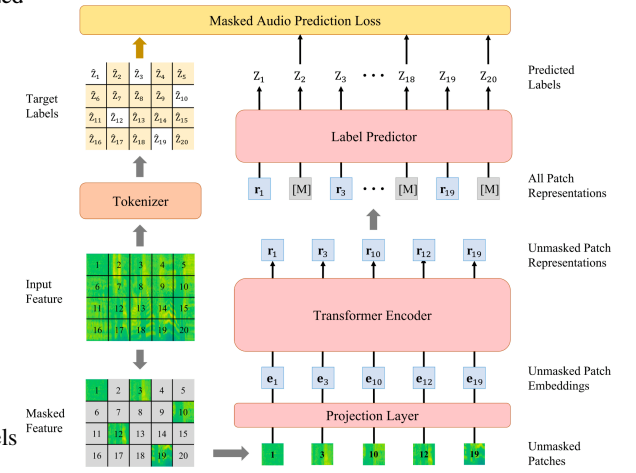
$$\mathcal{L} = \max \sum_{X\in\mathcal{D}} \sum_{n=1}^{N} \cos(\boldsymbol{o}_n, \hat{\boldsymbol{o}}_n) - \|sg[\ell_2(\boldsymbol{e}_n)] - \ell_2(\boldsymbol{v}_{\hat{z}_n})\|_2^2 - \|\ell_2(\boldsymbol{e}_n) - sg[\ell_2(\boldsymbol{v}_{\hat{z}_n})]\|_2^2.$$



Audio SSL Model (pre-training)

- Employ ViT as the backbone network, optimized to predict the discrete labels generated by the tokenizers as a masked pre-training task.
- Masking: Given the input patches $\boldsymbol{X}=\{\boldsymbol{x}_n\}$ and related target discrete labels $\hat{Z}=\{\hat{z}_n\}$, randomly mask 75% of the patches.
- ViT Encoder: Linearly project the unmasked patches $\{\boldsymbol{x}_{n'}\}$ to the patch embeddings $\{\boldsymbol{e}_{n'}\}$, then encode them using 12-layer ViT-Base to get the unmasked patch representations $\{\boldsymbol{r}_{n'}\}$.
- Label Predictor: Feed the order-restored unmasked $\{\boldsymbol{r}_{n'}\}$ and masked features $\{\mathbf{0}\}$ to the label predictor to get labels $Z=\{z_n\}_{n=1}^{N}$.
- Pre-training objective: Cross-entropy loss maximizing the log-likelihood of correct labels in masked positions given unmasked patches,

$$\mathcal{L}=-\sum_{\text{masked } n}\log p(\text{masked targets } \hat{z}_n \,|\, \text{unmasked patches } \{\boldsymbol{x}_{n'}\}).$$



Audio SSL Model (fine-tuning)

- Masking: Randomly mask time–frequency regions of the raw audio feature (spectrogram) as SpecAugment, then split them into flattened patches $\boldsymbol{X} = \{\boldsymbol{x}_n\}$.
- ViT Encoder: Unlike pre-training, linearly project the all patches $\{\boldsymbol{x}_n\}$ and encode them using 12-layer ViT-Base to get the entire patch representations $\boldsymbol{R} = \{\boldsymbol{r}_n\}$.
- Linear Classifier: Remove the discrete-label predictor used in pre-training, then attach a task-specific linear classifier to calculate the category probabilities,
  $p(C) = \text{Softmax}(\text{MeanPool}(\boldsymbol{W}_c \boldsymbol{R}))$
  with a linear projection $\boldsymbol{W}_c$.
- Fine-tuning objective: Employ the cross entropy loss for the single label classification tasks, and the binary cross entropy loss for the multi-label classification tasks.