

# Audio-3DVG: Unified Audio - Point Cloud Fusion for 3D Visual Grounding

Duc Cao-Dinh<sup>\*1</sup> Khai Le-Duc<sup>\*1,2,3</sup>

Anh Dao<sup>4</sup> Bach Phan Tat<sup>5</sup> Chris Ngo<sup>1</sup>

Duy M. H. Nguyen<sup>6,7,8</sup> Nguyen X. Khanh<sup>9</sup> Thanh Nguyen-Tang<sup>10</sup>

<sup>1</sup>Knovel Engineering Lab, Singapore <sup>2</sup>University of Toronto, Canada

<sup>3</sup>University Health Network, Canada <sup>4</sup>Michigan State University, USA <sup>5</sup>KU Leuven, Belgium

<sup>6</sup>German Research Center for Artificial Intelligence (DFKI), Germany

<sup>7</sup>Max Planck Research School for Intelligent Systems (IMPRS-IS), Germany

<sup>8</sup>University of Stuttgart, Germany <sup>9</sup>UC Berkeley, USA <sup>10</sup>Johns Hopkins University, USA

✉ ducdd@hanyang.ac.kr

✉ duckhai.le@mail.utoronto.ca



<https://github.com/leduckhai/Audio-3DVG>

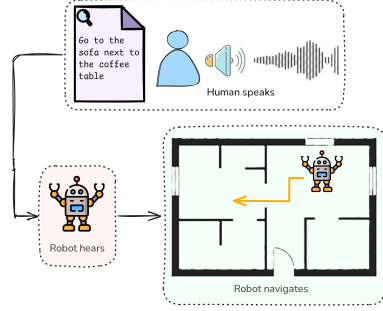
## Abstract

3D Visual Grounding (3DVG) involves localizing target objects in 3D point clouds based on natural language. While prior work has made strides using textual descriptions, leveraging spoken language—known as Audio-based 3D Visual Grounding—remains underexplored and challenging. Motivated by advances in automatic speech recognition (ASR) and speech representation learning, we propose Audio-3DVG, a simple yet effective framework that integrates audio and spatial information for enhanced grounding. Rather than treating speech as a monolithic input, we decompose the task into two complementary components. First, we introduce Object Mention Detection, a multi-label classification task that explicitly identifies which objects are referred to in the audio, enabling more structured audio-scene reasoning. Second, we propose an Audio-Guided Attention module that captures interactions between candidate objects and relational speech cues, improving target discrimination in cluttered scenes. To support benchmarking, we synthesize audio descriptions for standard 3DVG datasets, including ScanRefer, Sr3D, and Nr3D. Experimental results demonstrate that Audio-3DVG not only achieves new state-of-the-art performance in audio-based grounding, but also competes with text-based methods—highlighting the promise of integrating spoken language into 3D vision tasks.

## 1 Introduction

Visual grounding (VG) of referring expressions—the task of identifying visual entities described in natural language—has made significant progress in the 2D computer vision domain [54, 59, 46, 84]. With the rapid advancement of 3D sensing technologies and spatial data representations, this task has naturally extended into the 3D domain, where spatial reasoning becomes increasingly crucial. Unlike 2D images composed of grid-aligned pixels, 3D data—typically represented as point clouds—encodes richer geometric and spatial structures. This shift introduces both novel opportunities and unique challenges for accurately grounding language in three-dimensional space.

<sup>(\*)</sup>Equal contribution



(a) Audio-based 3D Visual Grounding enables object localization to support robot navigation.



(b) The **chair** has its back to the **windows**. the **chair** is next to the **desk** in the corner of the room facing the brown **armchair**.

Figure 1: a) Application of Audio based 3DVG to robot navigation task. b) Example sentences referring to objects in 3D scenes. Green boxes indicate objects of the same category as the target, while blue boxes highlight relational objects mentioned in the speech.

In line with this evolution, recent studies have transitioned from grounding objects in 2D images [53, 11, 15, 70] to grounding in 3D scenes [10, 1], where the goal is to localize objects referenced by natural language within a point cloud. While these advancements have yielded strong results, most approaches remain reliant on textual input. This dependence poses a barrier to practical deployment, as it requires users to manually input referring expressions using keyboards or touchscreens—a process that is inefficient in hands-busy or eyes-busy situations and inaccessible for users with motor impairments. To overcome the limitations of text-only inputs, recent research has begun exploring audio-based 3D visual grounding. A pioneering example is AP-Refer [96], a multimodal framework that replaces text with spoken language as the input modality. This framework aligns raw point clouds with corresponding audio signals to localize objects mentioned in natural speech, enabling audio-driven robot navigation, as illustrated in Figure 1a.

Despite its innovation, AP-Refer exhibits two critical limitations. First, it lacks an effective attention mechanism for fusing audio and spatial features, resulting in limited cross-modal understanding. Second, it ignores relational objects mentioned in speech, relying solely on audio and individual object features for grounding. This approach is especially inadequate in densely populated scenes where spatial context is vital.

In this paper, we present a novel framework that addresses both issues and significantly narrows the performance gap between audio-based and text-based methods. Our contributions are threefold: To address the first limitation, we build on the observation that target objects are often spatially related to other instances of the same category and to additional objects explicitly mentioned in the spoken input (as shown in Figure 1b). We introduce an auxiliary task called *Object Mention Detection*, which aims to identify the presence of relational objects referenced in the utterance. These relational objects serve as spatial anchors that guide the model in identifying the correct target among candidates. To resolve the second limitation, we propose an *Audio-Guided Attention Module*, which learns spatial and semantic relationships between candidate objects and relational entities, all conditioned on the audio signal. This attention mechanism improves the model’s ability to focus on relevant spatial dependencies, enhancing localization performance. In addition, we contribute new benchmark datasets for the 3DVG-Audio task. These include high-quality synthetic speech datasets based on existing 3DVG benchmarks, as well as a real-world spoken dataset to evaluate generalization.

As shown in our experiments, our model achieves substantial improvements over previous audio-based methods, reaching 17.03% and 17.15% accuracy on Acc@25 and Acc@50, respectively, while maintaining competitive performance with text-based systems.

In summary, the key contributions of this paper are as follows:

- We introduce a new framework for Audio 3D Visual Grounding that incorporates both target proposals and relational objects, effectively reducing noise in cluttered point clouds.
- We design a novel Audio-Guided Attention module that captures semantic and spatial relationships conditioned on spoken input.

- We create standardized benchmark datasets for the 3DVG-Audio task, including both synthetic and real-world audio, to facilitate robust evaluation and comparison.
- Our model establishes new state-of-the-art results among audio-based methods and achieves performance comparable to leading text-based approaches, highlighting its strong generalization ability and computational efficiency.

## 2 Related Work

### 2.1 Multi-modal research based on audio

Audio, a ubiquitous and easily accessible modality, has been extensively studied since the early days of Artificial Intelligence [78], particularly in tasks such as audio classification [23, 24, 51]. With the rapid advancement of deep learning in recent years, there has been growing interest in integrating audio with other modalities to address real-world challenges. Notable areas of multi-modal research involving audio include audio-text fusion [55, 89, 87]. These audio-based multi-modal approaches commonly rely on pre-trained audio feature extraction modules to effectively capture meaningful audio representations.

### 2.2 3D Visual Grounding

3D grounding aims to identify the object in a 3D scene that is referred to by a natural language expression. Datasets [1, 10] and methods [10, 12, 101, 86, 99] have been proposed to address this challenging task. Existing approaches can generally be categorized into two groups: one-stage and two-stage frameworks. One-stage methods directly fuse text features with visual representations at the patch or point level to regress the target bounding box [56, 44], enabling flexible detection of various objects from the input sentence. In contrast, two-stage methods follow a detect-then-match paradigm [92, 12, 99, 86], where the first stage generates object proposals and the second stage selects the best match based on the input description. This decoupling of object perception and cross-modal matching makes two-stage methods more interpretable and easier to analyze.

Following the pioneering works of ScanRefer [10] and ReferIt3D [1], research in 3D visual grounding (3DVG) has gained significant momentum, with numerous subsequent contributions substantially advancing the field and expanding its potential across a wide range of applications. Zhu et al. [101] introduced 3D-VisTA, a pre-trained Transformer optimized for aligning 3D visual and textual information, which can be effectively adapted to various downstream tasks. Guo et al. [29] proposed ViewRefer, a 3DVG framework that explores the integration of perspective knowledge from both language and 3D modalities, and further introduced a learnable multi-view model. Wang et al. [86] presented  $G^3 - LQ$ , a method specifically designed for 3D visual grounding, incorporating two specialized modules to explicitly model geometrically aware visual representations and generate fine-grained, language-guided object queries. Shi et al. [77] investigated the role of viewpoint information in 3DVG and proposed VPP-Net, a model that explicitly predicts the speaker’s viewpoint based on referring expressions and scene context. Additionally, several other influential works, including CORE-3DVG [90], Multi3DRefer [99], and D-LISA [97], have further contributed to the progress and richness of the 3DVG landscape.

### 2.3 Audio 3D Visual Grounding

While text-based 3D visual grounding has been extensively studied, audio-based multimodal approaches grounded in point clouds remain relatively underexplored and face notable limitations. Zhang et al. [96] introduced a novel multimodal task, termed AP-Refer, which integrates audio signals with 3D point cloud data. This work represents the first attempt to explore audio–point cloud fusion for multimodal understanding. By leveraging spatial cues from point clouds and semantic information from audio input, AP-Refer facilitates accurate localization of audio-referred objects within a 3D scene. Despite its promising potential, the performance of AP-Refer still lags behind that of text-based methods, underscoring the need for further research in this emerging area.

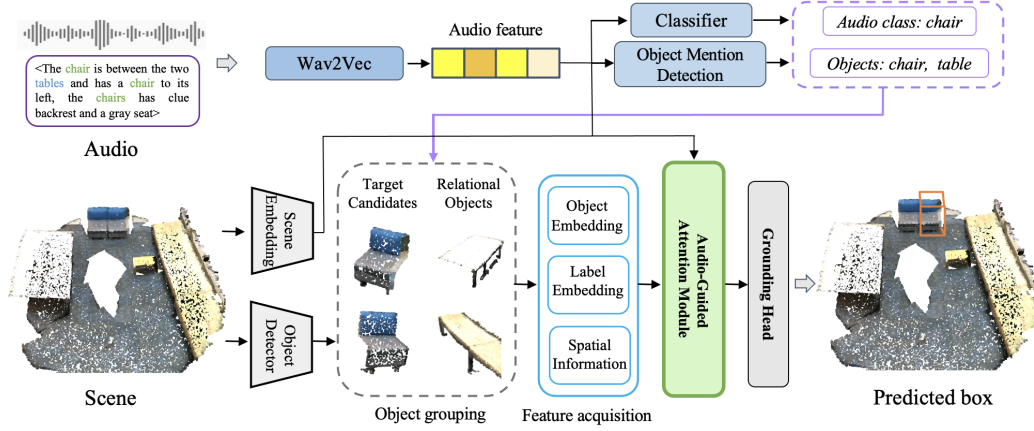


Figure 2: Illustration of the overall pipeline. Our network processes the 3D point cloud using an object detector and encodes the audio input with a Wav2Vec model. The extracted audio and different group object features are then fused via Audio-Guided Attention, which consists of Audio-Guided Self Attention and Audio-Guided Cross Attention modules, followed by a grounding head responsible for target object identification.

### 3 Method

Audio-3DVG is a novel framework for audio-based 3D visual grounding that performs target-relation referring to identify the most relevant instance-level object. As illustrated in Figure 2, the framework leverages point cloud instance segmentation to first extract individual object instances and then construct rich representations for each object within the entire scene. In the upper branch, we utilize the Wav2Vec model [6] to extract contextual audio representations. These features are then processed by audio classification and Object Mention Detection heads to identify the audio class—used to filter target proposals—and to detect the presence of relational entities within the context. Finally, an Audio-Guided Attention Module is introduced to fuse the multi-modal input representations and guide the selection of the optimal candidate.

#### 3.1 Instances Generation

Unlike ScanRefer [10], which treats all object proposals as potential candidates, our approach follows a recent *detection-then-matching* framework. We first extract all foreground instances from the input point cloud and leverage audio classification to identify a set of likely object candidates. The 3D visual grounding task is then reformulated as an instance-level matching problem. Specifically, given a scene  $S$  with point cloud data  $P_{scene} \in \mathbb{R}^{K \times 6}$ , we use PointGroup [43] to detect the object instances present in the scene, producing a set of objects  $(o_1, \dots, o_M)$ , each object  $o_i$  is represented by a subset of points  $P_i \in \mathbb{R}^{K \times 6}$ , where each point contains  $xyz$  coordinates and  $rgb$  color values. In our experiments, we sample  $K = 1024$  points per object. Each proposal is also associated with a 3D bounding box  $B_T \in \mathbb{R}^6$ , encoding the center coordinates and the dimensions of the box.

#### 3.2 Audio Encoding with Scene Embedding

Following Zhang et al. [96], Audio-3DVG employs an ASR pre-trained Wav2Vec model [6] for audio feature extraction (see Appendix Section D). Wav2Vec is an unsupervised speech representation learning framework that has shown strong performance across a wide range of speech-related downstream tasks (see Appendix Section A for a detailed discussion of our rationale). Given an input audio signal  $\mathcal{A}_T$ , Wav2Vec produces a feature representation  $F_{Audio} \in \mathbb{R}^{C \times L_a}$ , where  $L_a$  is the sequence length and  $C$  is the feature dimensionality of the high-dimensional latent space. To further encode temporal dependencies and contextual information,  $F_{Audio}$  is passed through bidirectional GRU layers, resulting in a fixed-length 768-dimensional vector ( $a$ ) used for downstream optimization.

To incorporate scene-level geometric context, we also embed the raw scene point cloud using a sparse convolutional neural network. Specifically, we employ the Minkowski Engine[17], a highly

efficient library for processing sparse tensor data, to extract spatial features from the 3D scene. The point cloud is voxelized and passed through a series of sparse convolutional layers to produce a global scene representation. This results in a compact 512-dimensional feature vector, which is concatenated with the audio features to capture the overall structural layout of the environment.

### 3.3 Audio Classification

Given the contextualized audio representation, we design a classifier to identify the target object referred to in the spoken utterance. The classifier is implemented as a simple multilayer perceptron (MLP), followed by a softmax layer to produce class probabilities. Let  $N_{cls}$  denote the total number of unique object classes defined in the dataset, the probability of the target class is computed as:

$$C_i^{N_{cls}} = Softmax(MLP(feature_{i=0}^{N_{cls}})) \quad (1)$$

### 3.4 Object Mention Detection

Most prior works [10, 1, 92] focus solely on analyzing candidate object proposals, often neglecting the presence of relational objects referenced in the input. This limitation can lead to ambiguity when identifying the correct target in scenes with dense object instances. To address this issue, we propose a novel auxiliary task called Object Mention Detection, which aims to identify relational objects mentioned in the audio. Concretely, we employ a lightweight multilayer perceptron (MLP) with  $N_{cls}$  binary classification heads, corresponding to maximum  $N_{cls}$  object classes may appear in the scene. Each head predicts the probability that its corresponding object class is mentioned in the spoken utterance. During inference, objects with predicted probabilities exceeding a predefined threshold are classified as relational objects.

### 3.5 Object Grouping

Before passing instances to subsequent modules, Audio-3DVG leverages the predicted instance set from earlier stages to filter candidate objects and identify relevant relational references. For example, as illustrated in Figure 2, given an audio description such as ‘*The chair is between two tables and has a chair to its left, the chairs has clue backrest and a gray seat*’, we begin with all object instances extracted from the original point cloud ( $o_1, \dots, o_M$ ). From this set, we retain only those instances classified as the target category, ‘*chair*’, along with the related category, ‘*table*’. These filtered sets correspond to the *target candidates* point set ( $P_{i=0}^N \subset P_{Target}$ , where each  $P_i \in \mathbb{R}^{1024 \times 6}$ ), and the *relational objects* point set ( $P_{j=0}^N \subset P_{Relational}$ , where each  $P_j \in \mathbb{R}^{1024 \times 6}$ ) in our pipeline.

### 3.6 Object Feature Acquisition

In contrast to Chen et al. [12], who separate semantic and spatial information in object instance representations, we argue that these features are inherently correlated. Neural networks can effectively learn to disentangle them using the positional encoding associated with each feature type. Motivated by this, we represent each object instance using a unified embedding formed by concatenating multiple feature modalities, including:

**Object Embedding:** For each object  $o_i$  in the set of target candidates and relational objects is represented as  $P_i \in \mathbb{R}^{K \times (3+F)}$ , where  $K$  denotes the number of points, 3 corresponds to the spatial coordinates ( $x, y, z$ ) associated with each point, and  $F$  includes additional point-wise attributes such as RGB color values in our case. we first normalize the coordinates of its point cloud into a unit ball. We then use PointNet++ [69]-a widely adopted framework for 3D semantic segmentation and object detection to extract object-level features, resulting in  $o_i^{obj} \in \mathbb{R}^{1 \times 1024}$

**Label Embedding:** To enhance the target classifier’s awareness of candidate categories, ReferIt3D [1] incorporates an auxiliary classification task within a joint optimization framework. Although this approach improves category understanding, it also adds to the overall learning complexity. In our network, we incorporate object labels as part of the object representation by embedding them using word embedding model. Specifically, for each instance in the set of target candidates and relational objects, we encode its class label using a pre-trained GloVe [67], resulting  $o_i^{label} \in \mathbb{R}^{1 \times 300}$ .

**Spatial Information:** To represent the absolute position of each instance  $O_i$  with corresponding representation  $P_i$ , we compute the object center  $o_i^{center} = [c_x, c_y, c_z] \in \mathbb{R}^3$  and the object size



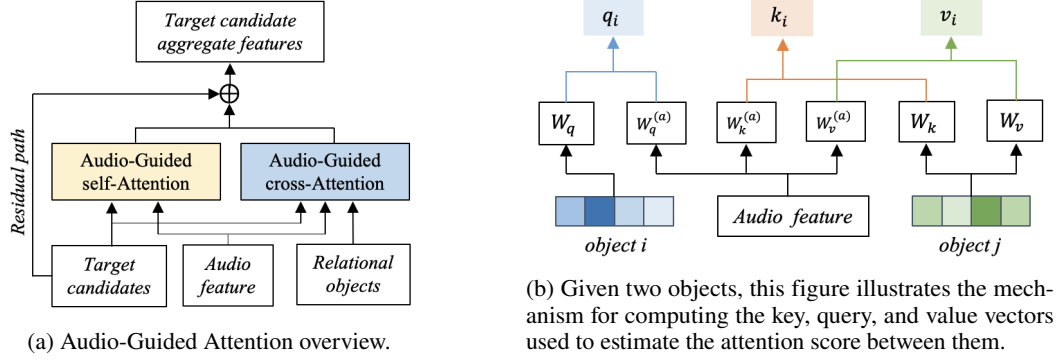


Figure 3: An overview of the proposed Audio-Guided Attention Module, comprising the Audio-Guided self-Attention and Audio-Guided cross-Attention submodules designed to model contextual and relational cues from speech.

$o_i^{size} = [z_x, z_y, z_z] \in \mathbb{R}^3$  These are derived from the object points  $P_i$  where the center is calculated as the mean of  $P_i$ , and the size corresponds to the spatial extent of  $P_i$ .

All of these features are concatenated into a single representation:

$$o_i^{rep} = [o_i^{obj}, o_i^{label}, o_i^{center}, o_i^{size}] \quad (2)$$

### 3.7 Audio-Guided Attention Module

In both text and audio descriptions, the target object is often identified through references to relational objects (e.g., “the chair is in front of the door, opposite to the coffee table”) or through spatial comparisons with objects of the same category (e.g., “of the two brown wooden doors, choose the door on the left when facing them”). Building on this observation, after obtaining feature representations for each target candidates and relational objects, We design an attention module comprising two components: **Audio-Guided Self-Attention**, which helps distinguish the target object from other instances within the same category, and **Audio-Guided Cross-Attention**, which captures spatial relationships between target candidates and relational objects referenced in the audio, as illustrated in Figure 3a

Specifically, as shown the the Figure 3b, given audio feature  $a \in \mathbb{R}^{d_a}$ , and a pair object  $o_i \in \mathbb{R}^{1 \times d}$  and  $o_j \in \mathbb{R}^{1 \times d}$ , each attention module first compute the embedding by projecting object, each modulated by audio feature in to query, key, and value spaces:

$$\mathbf{q}_i = W_q o_i + W_q^{(a)} a, \quad \mathbf{k}_j = W_k o_j + W_k^{(a)} a, \quad \mathbf{v}_i = W_v o_i + W_v^{(a)} a \quad (3)$$

Then the standard scaled dot-product attention value is calculated as:

$$\text{attention}_{ij} = \frac{\mathbf{q}_i^\top \mathbf{k}_j}{\sqrt{d}} \Rightarrow \alpha_{ij} = \text{softmax}_j(\text{attention}_{ij}) \quad (4)$$

Then

$$\mathbf{o}'_i = \sum_{j=1}^N \alpha_{ij} \mathbf{v}_j \quad (5)$$

Each output  $\mathbf{o}'_i$  is audio-modulated feature representing the object with the context from other object. In the implementation, we use stack **multi-head attention** and concatenate the outputs:

$$\text{MultiHead}(O, a) = \text{Concat}(\text{head}_1, \dots, \text{head}_h) W_o \quad (6)$$

The audio-guided attention scores are computed between object pairs in the Audio-Guided Self-Attention module, resulting in  $\mathbf{O}' = \mathbf{o}'_1, \dots, \mathbf{o}'_N$ . In contrast, the Audio-Guided Cross-Attention module computes attention scores between each target candidate and all relational objects mentioned in the audio, producing  $\mathbf{O}'' = \mathbf{o}''_1, \dots, \mathbf{o}''_N$ . Finally, the aggregated feature representation for each target candidate is obtained by summarizing the features from  $\mathbf{O}$ ,  $\mathbf{O}'$ , and  $\mathbf{O}''$ .

### 3.8 Grounding Head

Finally, we employ a classifier to identify the target object referenced in the speech. This classifier consists of a multilayer perceptron (MLP) followed by a softmax layer, which predicts the most likely target among the  $N$  candidate objects.

### 3.9 Loss functions

We employ multiple loss functions to train Audio-3DVG across its various tasks, including an audio classification loss  $\mathcal{L}_{audio}^{cls}$ , a multi-label classification loss for the Object Mention Detection task  $\mathcal{L}_{audio}^{OMD}$ , and an object classification loss for the grounding task  $\mathcal{L}_{object}^{cls}$ . Therefore, the overall training objective is as follows:

$$\mathcal{L} = \lambda_a \mathcal{L}_{audio}^{cls} + \lambda_b \mathcal{L}_{audio}^{OMD} + \lambda_c \mathcal{L}_{object}^{cls} \quad (7)$$

Where  $\lambda_a$ ,  $\lambda_b$ , and  $\lambda_c$  are three hyper-parameters to balance the losses.

## 4 Datasets

**ScanRefer [10]:** The dataset contains 51,583 human-written sentences annotated for 800 scenes in ScanNet dataset [18]. Following the official split, we use 36,665 samples for training and 9,508 for validation. Based on whether the target object belongs to a unique category within the scene, the dataset is further divided into two subsets: "unique", where the target class appears only once, and "multiple", where it appears more than once.

**Nr3D [1]:** The dataset comprises 37,842 human-written sentences that refer to annotated objects in 3D indoor scenes from the ScanNet dataset [18]. It includes 641 scenes, with 511 used for training and 130 for validation, covering a total of 76 target object classes. Each sentence is crafted to refer to an object surrounded by multiple same-class distractors. For evaluation, the sentences are divided into "easy" and "hard" subsets: in the easy subset, the target object has only one same-class distractor, whereas in the hard subset, multiple distractors are present. Additionally, the dataset is categorized into "view-dependent" and "view-independent" subsets, based on whether grounding the referred object requires a specific viewpoint.

**Sr3D [1]:** This dataset is constructed using sentence templates to automatically generate referring expressions. These sentences rely solely on spatial relationships to distinguish between objects of the same class. It contains 1,018 training scenes and 255 validation scenes from ScanNet dataset [18], with a total of 83,570 sentences. For evaluation, it can be partitioned in the same manner as the Nr3D dataset.

To address the data scarcity issue in the Audio-3D visual grounding task, we efficiently convert ScanRefer’s natural language descriptions into audio using Spark-TTS [85]—an advanced and flexible text-to-speech system that leverages large language models (LLMs) to generate highly accurate and natural-sounding speech. The detailed analysis and configuration of the generated data are presented in the appendix.

## 5 Experiments

### 5.1 Experimental Setup

**Evaluation Metrics:** We evaluate models under two evaluation settings. One uses ground-truth object proposals, which is the default setting in the Nr3D and Sr3D datasets. The metric is the accuracy of selecting the target bounding box among the proposals. The other setting does not provide ground-truth object proposals and requires the model to regress a 3D bounding box, which is the default setting for the ScanRefer dataset. The evaluation metrics are  $\text{acc}@0.25$  and  $\text{acc}@0.5$ , which is the percentage of correctly predicted bounding boxes whose IoU is larger than 0.25 or 0.5 with the ground-truth.

**Implementation details.** We adopt the official pre-trained PointGroup [43] as the backbone for instance segmentation. For audio encoding, we utilize a BiGRU to extract word-level features with a channel dimension of 768. All employed MLPs use hidden layers configured as [521,64], followed

by Batch Normalization and ReLU activation. We use 8 attention heads, each producing features with a dimensionality of 128. The network is trained for 30 epochs using the Adam optimizer with a batch size of 32. The learning rate is initialized at 0.0005 and decayed by a factor of 0.9 every 5 epochs. All experiments are implemented in PyTorch and run on a single NVIDIA RTX 3090 GPU.

## 5.2 Experimental Results

We first present the performance results for the Audio Classification and Object Mention Detection tasks, averaged across all three datasets. The Audio Classification task achieves a high accuracy of 96%. For Object Mention Detection, we report the average precision, recall, and F1-score for each object class, as shown in Figure 1.

Metric	cabinet	bed	chair	sofa	table	door	window	bookshelf	picture	counter	desk	curtain	refrigerator	shower curtain	toilet	sink	bathub	others	average
precision	0.97	0.89	1.00	0.85	1.00	0.98	0.95	0.38	0.53	0.71	0.97	0.61	0.00	0.24	0.74	0.85	0.27	0.98	0.72
recall	0.96	0.87	1.00	0.84	1.00	0.97	0.95	0.38	0.51	0.70	0.97	0.61	0.00	0.24	0.73	0.85	0.27	0.97	0.71
F1	0.96	0.88	1.00	0.84	1.00	0.98	0.95	0.37	0.52	0.70	0.97	0.61	0.00	0.24	0.74	0.85	0.27	0.98	0.71

Table 1: Performance of Object Mention Detection task by each class and average.

Method	Venue	Input	Unique		Multiple		Overall	
			acc@0.25	acc@0.5	acc@0.25	acc@0.5	acc@0.25	acc@0.5
ScanRefer [10]	ECCV 20	Text	65.00	43.31	30.63	19.75	37.30	24.32
TGNN [36]	AAAI 21	Text	68.61	56.80	29.84	23.18	37.37	29.70
Non-SAT [91]	ICCV 21	Text	68.48	47.38	31.81	21.34	38.92	26.40
SAT [91]	ICCV 21	Text	73.21	50.83	37.64	25.16	44.54	30.14
3DVG-Trans [100]	ICCV 21	Text	77.16	58.47	38.38	28.70	45.90	34.47
InstanceRefer [92]	ICCV 21	Text	78.37	66.88	27.90	21.83	37.69	30.57
3D-SPS [56]	CVPR 22	Text	81.63	64.77	39.48	29.61	47.65	36.42
Multi-view [37]	CVPR 22	Text	77.67	66.45	31.92	25.26	40.80	33.26
ViL3DRel [12]	NeurIPS 22	Text	81.58	68.62	40.30	30.71	47.94	37.73
3D-VLP [12]	CVPR 23	Text	84.23	64.61	43.51	33.41	51.41	39.46
3D-VisTA [101]	ICCV 23	Text	77.40	70.90	38.70	34.80	45.90	41.50
$G^3 - LQ$ [86]	CVPR 24	Text	88.09	72.73	51.48	40.80	56.90	45.58
3DVG-Trans [100]	ICCV 21	Audio2Text	74.92	56.67	35.43	26.92	43.23	33.87
InstanceRefer [92]	ICCV 21	Audio2Text	73.28	64.20	29.12	22.98	38.46	30.90
AP-Refer [96]	Neurocomputing 24	Audio	48.62	29.59	16.94	9.96	23.09	13.77
Ours		Audio	<b>75.06</b>	<b>64.08</b>	<b>30.54</b>	<b>24.06</b>	<b>40.12</b>	<b>30.92</b>

Table 2: Comparison of grounding accuracy across different methods under different inputs.

For the 3D visual grounding performance, we present the comparative results on the ScanRefer dataset using detected objects from PointGroup [43]. Given the same audio input, our model demonstrates a substantial performance improvement over AP-Refer, highlighting the effectiveness of our approach. Furthermore, our method achieves competitive results compared to text-based methods. However, it is important to note that this comparison is not entirely fair, as text inputs provide richer and error-free linguistic information, whereas our audio-based approach is subject to potential inaccuracies introduced during text-to-speech conversion. To enable a fairer comparison, we convert the audio inputs back to text by Whisper [73] and evaluate the performance of recent state-of-the-art methods, as reported in Table 2.

Table 3 presents a comparison between our Audio-3DVG model and state-of-the-art methods on the Nr3D and Sr3D datasets, where all baseline methods and our approach utilize ground-truth object proposals. It is important to note that this comparison is not entirely fair, as all prior works rely on text-based input, while our method is the first to leverage audio input in these datasets.



Method	Input	Nr3D					Sr3D				
		Overall	Easy	Hard	View dep	View Indep	Overall	Easy	Hard	View Indep	View Indep
ReferIt3D [1]	Text	35.6	43.6	27.9	32.5	37.1	40.8	44.7	31.5	39.2	40.8
ScanRefer [10]	Text	34.2	41.0	23.5	29.9	35.4	-	-	-	-	-
InstanceRefer [92]	Text	38.8	46.0	31.8	34.5	41.9	48.0	51.1	40.5	45.4	48.1
3DVG-Trans [100]	Text	40.8	48.5	34.8	34.8	43.7	51.4	54.2	44.9	44.6	51.7
SAT [91]	Text	49.2	56.3	42.4	46.9	50.4	57.9	61.2	50.0	49.2	58.3
3D-SPS [56]	Text	51.5	58.1	45.1	48.0	53.2	62.6	56.2	65.4	49.2	63.2
Multi-view [37]	Text	55.1	61.3	49.1	54.3	55.4	64.5	66.9	58.8	58.4	64.7
ViL3DRel [12]	Text	64.4	70.2	57.4	62.0	64.5	72.8	74.9	67.9	63.8	73.2
<b>Ours</b>	Audio	37.4	45.2	30.9	34.1	40.7	48.3	51.3	40.9	45.1	48.6

Table 3: Comparison of grounding accuracy (%) on the Nr3D and Sr3D benchmarks. To the best of our knowledge, this is the first study to utilize audio input on these datasets.

## 6 Ablation Study

### 6.1 Result with different audio generation methods

We further evaluate the performance of Audio-3DVG on the ScanRefer dataset using different text-to-speech (TTS) methods. Following the AP-Refer setup, we replace Spark-TTS with Matcha-TTS [60] to generate audio inputs for this experiment. The results, shown in Table 4, indicate that although training the model with Matcha-TTS leads to slightly lower performance compared to Spark-TTS, it still surpasses the performance of AP-Refer [96] as reported in Table 2.

TTS method	Unique		Multiple		Overall	
	acc@0.25	acc@0.5	acc@0.25	acc@0.5	acc@0.25	acc@0.5
Matcha-TTS [60]	72.86	60.05	30.80	22.98	38.72	28.98
Spark-TTS [85]	<b>75.06</b>	<b>64.08</b>	<b>30.54</b>	<b>24.06</b>	<b>40.12</b>	<b>30.92</b>

Table 4: Performance of Audio-3DVG on ScanRefer dataset with different TTS methods

### 6.2 Impact of Audio-Guided Attention

We conduct an experiment on the ScanRefer dataset to evaluate the effectiveness of the proposed Audio-Guided Attention module, which is not used in AP-Refer. Specifically, we replace the Audio-Guided Attention module with a standard MLP-based classifier that treats all detected objects as potential candidates, where the audio feature is simply concatenated with each object’s feature. The comparison results in Table 5 demonstrate the superiority of our Audio-Guided Attention design.

Method	Unique		Multiple		Overall	
	acc@0.25	acc@0.5	acc@0.25	acc@0.5	acc@0.25	acc@0.5
MLP	70.12	58.41	27.08	21.20	36.54	26.14
Audio-Guided Attention	<b>75.06</b>	<b>64.08</b>	<b>30.54</b>	<b>24.06</b>	<b>40.12</b>	<b>30.92</b>

Table 5: Ablation study on the ScanRefer dataset demonstrating the effectiveness of the proposed Audio-Guided Attention module.

## 7 Conclusion

In summary, this work introduces a novel approach that leverages audio for the 3D visual grounding task. Our contributions include a method for detecting target candidates and relational objects, an effective feature formulation strategy, and a robust attention module for identifying targets within dense object scenes. Additionally, we provide a synthetic audio dataset to support future research in this area. Our results demonstrate the effectiveness of using audio for 3D vision tasks and highlight its potential as a promising direction for future exploration.

## **8 Limitations**

Despite its demonstrated effectiveness, leveraging audio for the 3D visual grounding task still faces several limitations that future research should address. First, due to class imbalance in the dataset, the Object Mention Detection task struggles to accurately detect the presence of rare object classes, as shown in Table 1. To mitigate this issue, more balanced and diverse datasets are needed to improve the model’s ability to generalize across all categories. Next, similar to previous work, our approach relies heavily on the performance of the 3D object segmentation method. Therefore, integrating more robust and accurate 3D segmentation solutions could significantly enhance the overall effectiveness and reliability of the model.

## **9 Acknowledgement**

Most of the ASR theory in this work were borrowed from lectures by Prof. Hermann Ney, Ralf Schluter, and PhD Albert Zeyer, as well as from PhD dissertations and especially master thesis by Minh Nghia Phan at RWTH Aachen University.

## References

- [1] Panos Achlioptas, Ahmed Abdelreheem, Fei Xia, Mohamed Elhoseiny, and Leonidas Guibas. Referit3d: Neural listeners for fine-grained 3d object identification in real-world scenes. *16th European Conference on Computer Vision (ECCV)*, 2020.
- [2] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. In *International Conference on Learning Representations (ICLR) Workshops*, 2016.
- [3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.
- [4] Alexei Baevski and Abdelrahman Mohamed. Effectiveness of self-supervised pre-training for asr. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7694–7698. IEEE, 2020.
- [5] Alexei Baevski, Steffen Schneider, and Michael Auli. vq-wav2vec: Self-supervised learning of discrete speech representations. In *International Conference on Learning Representations*.
- [6] Alexei Baevski, Henry Zhou, Abdel rahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *ArXiv*, 2020.
- [7] Junwen Bai, Bo Li, Yu Zhang, Ankur Bapna, Nikhil Siddhartha, Khe Chai Sim, and Tara N Sainath. Joint unsupervised and supervised training for multilingual asr. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6402–6406. IEEE, 2022.
- [8] T. Bayes. An Essay Towards Solving a Problem in the Doctrine of Chances. *Philosophical Transactions of the Royal Society of London*, 53:370–418, 1763.
- [9] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.
- [10] Dave Zhenyu Chen, Angel X Chang, and Matthias Nießner. Scanrefer: 3d object localization in rgb-d scans using natural language. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 202–221. Springer, 2020.
- [11] Howard Chen, Alane Suhr, Dipendra Kumar Misra, Noah Snavely, and Yoav Artzi. Touch-down: Natural language navigation and spatial reasoning in visual street environments. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [12] Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Language conditioned spatial relation reasoning for 3d object grounding. In *NeurIPS*, 2022.
- [13] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, 2022.
- [14] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PmLR, 2020.
- [15] Zhenfang Chen, Peng Wang, Lin Ma, Kwan-Yee Kenneth Wong, and Qi Wu. Cops-ref: A new dataset and task on compositional referring expression comprehension. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

- [16] Chung-Cheng Chiu, James Qin, Yu Zhang, Jiahui Yu, and Yonghui Wu. Self-supervised learning with random-projection quantizer for speech recognition. In *International Conference on Machine Learning*, pages 3915–3924. PMLR, 2022.
- [17] Christopher Choy, Joon Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [18] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas A. Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.
- [20] Ruchao Fan, Yunzheng Zhu, Jinhan Wang, and Abeer Alwan. Towards better domain adaptation for self-supervised models: A case study of child asr. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1242–1252, 2022.
- [21] Ruchao Fan, Natarajan Balaji Shankar, and Abeer Alwan. Benchmarking children’s asr with supervised and self-supervised speech foundation models. In *Proc. Interspeech 2024*, pages 5173–5177, 2024.
- [22] Yonggan Fu, Yang Zhang, Kaizhi Qian, Zhifan Ye, Zhongzhi Yu, Cheng-I Jeff Lai, and Celine Lin. Losses can be blessings: Routing self-supervised speech representations towards efficient multilingual and multitask speech processing. *Advances in Neural Information Processing Systems*, 35:20902–20920, 2022.
- [23] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.
- [24] Mariana-Iuliana Georgescu, Eduardo Fonseca, Radu Tudor Ionescu, Mario Lucic, Cordelia Schmid, and Anurag Arnab. Audiovisual masked autoencoders. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2022.
- [25] Alex Graves. Connectionist temporal classification. In *Supervised sequence labelling with recurrent neural networks*, pages 61–93. Springer, 2012.
- [26] Alex Graves and Navdeep Jaitly. Towards End-to-End Speech Recognition with Recurrent Neural Networks. pages 1764–1772, Beijing, China, 2014.
- [27] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.
- [28] Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. On Using Monolingual Corpora in Neural Machine Translation, 2015. [arXiv:1503.03535](https://arxiv.org/abs/1503.03535).
- [29] Ziyu Guo, Yiwen Tang, Renrui Zhang, Dong Wang, Zhigang Wang, Bin Zhao, and Xuelong Li. Viewrefer: Grasp the multi-view knowledge for 3d visual grounding. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.
- [30] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.
- [31] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

- [32] Takaaki Hori, Shinji Watanabe, Yu Zhang, and William Chan. Advances in Joint CTC-Attention Based End-to-End Speech Recognition with a Deep CNN Encoder and RNN-LM. pages 949–953, Stockholm, Sweden, 2017.
- [33] Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28(3/4):321–377, 1936.
- [34] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM transactions on audio, speech, and language processing*, 29:3451–3460, 2021.
- [35] Wei-Ning Hsu, Yao-Hung Hubert Tsai, Benjamin Bolte, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: How much can a bad teacher benefit asr pre-training? In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6533–6537. IEEE, 2021.
- [36] Pin-Hao Huang, Han-Hung Lee, Hwann-Tzong Chen, and Tyng-Luh Liu. Text-guided graph neural networks for referring 3d instance segmentation. In *AAAI Conference on Artificial Intelligence*, 2021.
- [37] Shijia Huang, Yilun Chen, Jiaya Jia, and Liwei Wang. Multi-view transformer for 3d visual grounding. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [38] Kyuyeon Hwang and Wonyong Sung. Character-level language modeling with hierarchical recurrent neural networks. pages 5720–5724, New Orleans, LA, 2017. IEEE.
- [39] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.
- [40] Kazuki Irie, Zhihong Lei, Liuhui Deng, Ralf Schlüter, and Hermann Ney. Investigation on estimation of sentence probability by combining forward, backward and bi-directional lstm-rnns. In *Interspeech 2018*, pages 392–395, 2018.
- [41] Kazuki Irie, Albert Zeyer, Ralf Schlüter, and Hermann Ney. Language Modeling with Deep Transformers. pages 3905–3909, Graz, Austria, 2019.
- [42] Frederick Jelinek, Robert L. Mercer, Lalit R. Bahl, and Janet M. Baker. Perplexity—a measure of the difficulty of speech recognition tasks. *Journal of the Acoustical Society of America*, 62, 1977.
- [43] Li Jiang, Hengshuang Zhao, Shaoshuai Shi, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Point-group: Dual-set point grouping for 3d instance segmentation. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [44] Aishwarya Kamath, Mannat Singh, Yann LeCun, Ishan Misra, Gabriel Synnaeve, and Nicolas Carion. Mdetr - modulated detection for end-to-end multi-modal understanding. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1760–1770, 2021.
- [45] Anjuli Kannan, Yonghui Wu, Patrick Nguyen, Tara N. Sainath, Zhifeng Chen, and Rohit Prabhavalkar. An Analysis of Incorporating an External Language Model into a Sequence-to-Sequence Model. pages 5824–5828, Calgary, Alberta, Canada, 2018. DOI: [10.1109/ICASSP.2018.8462682](https://doi.org/10.1109/ICASSP.2018.8462682).
- [46] Sahar Kazemzadeh, Vicente Ordonez, Marc andre Matten, and Tamara L. Berg. Referitgame: Referring to objects in photographs of natural scenes. In *Conference on Empirical Methods in Natural Language Processing*, 2014.
- [47] Reinhard Kneser and Hermann Ney. Improved backing-off for m-gram language modeling. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 181–184, Detroit, Michigan, USA, 1995.

- [48] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning (ICML)*, pages 3519–3529. PMLR, 2019.
- [49] Chengxi Lei, Satwinder Dr Singh, Feng Hou, and Ruili Wang. Mix-fine-tune: An alternate fine-tuning strategy for domain adaptation and generalization of low-resource asr. In *Proceedings of the 6th ACM International Conference on Multimedia in Asia*, pages 1–7, 2024.
- [50] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics. Doklady*, 10:707–710, 1965.
- [51] Xian Li, Nian Shao, and Xiaofei Li. Self-supervised audio teacher-student transformer for both clip-level and frame-level tasks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023.
- [52] Andy T Liu, Shang-Wen Li, and Hung-yi Lee. Tera: Self-supervised learning of transformer encoder representation for speech. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2351–2366, 2021.
- [53] Runtao Liu, Chenxi Liu, Yutong Bai, and Alan Loddon Yuille. Clevr-ref+: Diagnosing visual reasoning with referring expressions. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [54] Xihui Liu, Zihao Wang, Jing Shao, Xiaogang Wang, and Hongsheng Li. Improving referring expression grounding with cross-modal attention-guided erasing. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [55] Siyu Lou, Xuenan Xu, Mengyue Wu, and K. Yu. Audio-text retrieval in context. *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022.
- [56] Junyu Luo, Jiahui Fu, Xianghao Kong, Chen Gao, Haibing Ren, Hao Shen, Huaxia Xia, and Si Liu. 3d-sps: Single-stage 3d visual grounding via referred point progressive selection. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [57] Christoph Lüscher, Eugen Beck, Kazuki Irie, Markus Kitza, Wilfried Michel, Albert Zeyer, Ralf Schlüter, and Hermann Ney. Rwth asr systems for librispeech: Hybrid vs attention. In *Proc. Interspeech 2019*, pages 231–235, 2019.
- [58] Vimal Manohar, Daniel Povey, and Sanjeev Khudanpur. Semi-supervised maximum mutual information training of deep neural network acoustic models. In *Interspeech 2015*, pages 2630–2634, 2015.
- [59] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana-Maria Camburu, Alan Loddon Yuille, and Kevin P. Murphy. Generation and comprehension of unambiguous object descriptions. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [60] Shivam Mehta, Ruibo Tu, Jonas Beskow, Éva Székely, and Gustav Eje Henter. Matcha-tts: A fast tts architecture with conditional flow matching. *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.
- [61] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.
- [62] Abdelrahman Mohamed, Hung-yi Lee, Lasse Borgholt, Jakob D Havtorn, Joakim Edin, Christian Igel, Katrin Kirchhoff, Shang-Wen Li, Karen Livescu, Lars Maaløe, et al. Self-supervised speech representation learning: A review. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1179–1210, 2022.
- [63] Ari S Morcos, Maithra Raghu, and Samy Bengio. Insights on representational similarity in neural networks with canonical correlation. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5727–5736, 2018.



- [64] Hermann Ney. Acoustic modeling of phoneme units for continuous speech recognition. In *Proc. Fifth Europ. Signal Processing Conf.*, pages 65–72, 1990.
- [65] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O’Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024.
- [66] Ankita Pasad, Ju-Chieh Chou, and Karen Livescu. Layer-wise analysis of a self-supervised speech representation model. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 914–921. IEEE, 2021.
- [67] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing*, 2014.

- [68] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.
- [69] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.
- [70] Yuankai Qi, Qi Wu, Peter Anderson, Xin Eric Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. Reverie: Remote embodied visual referring expression in real indoor environments. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [71] Lawrence R Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. 77(2):257–286, 1989.
- [72] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [73] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, 2022.
- [74] Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 6078–6087, 2017.
- [75] Sakriani Sakti and Benita Angela Titalim. Leveraging the multilingual indonesian ethnic languages dataset in self-supervised models for low-resource asr task. In *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1–8. IEEE, 2023.
- [76] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised pre-training for speech recognition. In *Proc. Interspeech 2019*, pages 3465–3469, 2019.
- [77] Xiangxi Shi, Zhonghua Wu, and Stefan Lee. Viewpoint-aware visual grounding in 3d scenes. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.
- [78] Eli Shlizerman, Lucio M. Dery, Hayden Schoen, and Ira Kemelmacher-Shlizerman. Audio to body dynamics. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017.
- [79] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. LSTM Neural Networks for Language Modeling. pages 194–197, Portland, OR, 2012.
- [80] Gabriel Synnaeve, Qiantong Xu, Jacob Kahn, Tatiana Likhomanenko, Edouard Grave, Vineel Pratap, Anuroop Sriram, Vitaliy Liptchinsky, and Ronan Collobert. End-to-end asr: from supervised to semi-supervised learning with modern architectures. In *ICML 2020 Workshop on Self-supervision in Audio and Speech*.
- [81] Valentin Vielzeuf. Investigating the ‘autoencoder behavior’ in speech self-supervised models: a focus on hubert’s pretraining. *arXiv preprint arXiv:2405.08402*, 2024.
- [82] Peter Vieting, Christoph Lüscher, Julian Dierkes, Ralf Schlüter, and Hermann Ney. Efficient utilization of large pre-trained models for low resource asr. In *2023 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)*, pages 1–5. IEEE, 2023.
- [83] Elena Voita, Rico Sennrich, and Ivan Titov. The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4396–4406, 2019.

- [84] Liwei Wang, Yin Li, Jing Huang, and Svetlana Lazebnik. Learning two-branch neural networks for image-text matching tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41:394–407, 2017.
- [85] Xinsheng Wang, Mingqi Jiang, Ziyang Ma, Ziyu Zhang, Songxiang Liu, Linqin Li, Zheng Liang, Qixi Zheng, Rui Wang, Xiaoqin Feng, Weizhen Bian, Zhen Ye, Sitong Cheng, Ruibin Yuan, Zhixian Zhao, Xinfu Zhu, Jiahao Pan, Liumeng Xue, Pengcheng Zhu, Yunlin Chen, Zhifei Li, Xie Chen, Lei Xie, Yi-Min Guo, and Weifeng Xue. Spark-tts: An efficient llm-based text-to-speech model with single-stream decoupled speech tokens. *ArXiv*, 2025.
- [86] Yuan Wang, Yali Li, and Shen Wang. G3-lq: Marrying hyperbolic alignment with explicit semantic-geometric modeling for 3d visual grounding. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.
- [87] Yifei Xin, Baojun Wang, and Lifeng Shang. Cooperative game modeling with weighted token-level alignment for audio-text retrieval. *IEEE Signal Processing Letters*, 2023.
- [88] Xiaoshuo Xu, Yueteng Kang, Songjun Cao, Binghuai Lin, and Long Ma. Explore wav2vec 2.0 for mispronunciation detection. In *Interspeech*, pages 4428–4432, 2021.
- [89] Xuenan Xu, Xiaohang Xu, Zeyu Xie, Pingyue Zhang, Mengyue Wu, and Kai Yu. A detailed audio-text data simulation pipeline using single-event sounds. *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024.
- [90] Li Yang, Chunfen Yuan, Ziqi Zhang, Zhongang Qi, Yan Xu, Wei Liu, Ying Shan, Bing Li, Weiping Yang, Peng Li, Yan Wang, and Weiming Hu. Exploiting contextual objects and relations for 3d visual grounding. In *Neural Information Processing Systems*, 2023.
- [91] Zhengyuan Yang, Songyang Zhang, Liwei Wang, and Jiebo Luo. Sat: 2d semantics assisted training for 3d visual grounding. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [92] Zhihao Yuan, Xu Yan, Yinghong Liao, Ruimao Zhang, Zhen Li, and Shuguang Cui. Instancerefer: Cooperative holistic understanding for visual grounding on point clouds through instance multi-level contextual referring. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [93] Albert Zeyer, Parnia Bahar, Kazuki Irie, Ralf Schlüter, and Hermann Ney. A comparison of transformer and lstm encoder decoder models for asr. In *IEEE Automatic Speech Recognition and Understanding Workshop*, pages 8–15, Sentosa, Singapore, 2019.
- [94] Albert Zeyer, André Merboldt, Ralf Schlüter, and Hermann Ney. A new training pipeline for an improved neural transducer. In *Interspeech*, Shanghai, China, 2020. [slides].
- [95] Albert Zeyer, Ralf Schlüter, and Hermann Ney. Why does CTC Result in Peaky Behavior?, 2021. [arXiv:2105.14849](https://arxiv.org/abs/2105.14849).
- [96] Can Zhang, Zeyu Cai, Xunhao Chen, Feipeng Da, and Shaoyan Gai. 3d visual grounding-audio: 3d scene object detection based on audio. *Neurocomputing*, 611:128637, 2024.
- [97] Haomeng Zhang, Chiao-An Yang, and Raymond A. Yeh. Multi-object 3d grounding with dynamic modules and language-informed spatial attention, 2024.
- [98] Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. Revisiting few-sample bert fine-tuning. In *International Conference on Learning Representations*.
- [99] Yiming Zhang, ZeMing Gong, and Angel X. Chang. Multi3drefer: Grounding text description to multiple 3d objects. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.
- [100] Lichen Zhao, Daigang Cai, Lu Sheng, and Dong Xu. 3dvg-transformer: Relation modeling for visual grounding on point clouds. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [101] Ziyu Zhu, Xiaojian Ma, Yixin Chen, Zhidong Deng, Siyuan Huang, and Qing Li. 3d-vista: Pre-trained transformer for 3d vision and text alignment. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Related Work</b>	<b>3</b>
2.1	Multi-modal research based on audio . . . . .	3
2.2	3D Visual Grounding . . . . .	3
2.3	Audio 3D Visual Grounding . . . . .	3
<b>3</b>	<b>Method</b>	<b>4</b>
3.1	Instances Generation . . . . .	4
3.2	Audio Encoding with Scene Embedding . . . . .	4
3.3	Audio Classification . . . . .	5
3.4	Object Mention Detection . . . . .	5
3.5	Object Grouping . . . . .	5
3.6	Object Feature Acquisition . . . . .	5
3.7	Audio-Guided Attention Module . . . . .	6
3.8	Grounding Head . . . . .	7
3.9	Loss functions . . . . .	7
<b>4</b>	<b>Datasets</b>	<b>7</b>
<b>5</b>	<b>Experiments</b>	<b>7</b>
5.1	Experimental Setup . . . . .	7
5.2	Experimental Results . . . . .	8
<b>6</b>	<b>Ablation Study</b>	<b>9</b>
6.1	Result with different audio generation methods . . . . .	9
6.2	Impact of Audio-Guided Attention . . . . .	9
<b>7</b>	<b>Conclusion</b>	<b>9</b>
<b>8</b>	<b>Limitations</b>	<b>10</b>
<b>9</b>	<b>Acknowledgement</b>	<b>10</b>
<b>A</b>	<b>Self-Supervised Speech Representation Learning</b>	<b>20</b>
A.1	wav2vec 2.0 Architecture . . . . .	20
A.1.1	Self-Supervised Pre-training . . . . .	20
A.1.2	Supervised Fine-tuning . . . . .	22
A.2	Rationale . . . . .	22
A.3	Analysis Methods . . . . .	23
A.3.1	Canonical Correlation Analysis . . . . .	23

A.3.2	Mutual Information Estimation . . . . .	23
A.4	Findings of Self-Supervised Representation Learning . . . . .	24
A.4.1	Reconstruction Behavior . . . . .	24
A.4.2	Encoded Acoustic-Linguistic Information . . . . .	24
A.4.3	Word Meaning Representation . . . . .	25
A.4.4	Fine-tuning Effect . . . . .	26
<b>B</b>	<b>Weakly Supervised Speech Representation Learning</b>	<b>28</b>
B.1	Attention Encoder Decoder (AED) . . . . .	28
B.1.1	Whisper Architecture . . . . .	28
B.1.2	Deepgram Nova-2 Architecture . . . . .	28
B.2	Rationale . . . . .	30
<b>C</b>	<b>Raw waveform vs MFCC</b>	<b>31</b>
C.1	Mel-Frequency Cepstral Coefficients (MFCCs) . . . . .	31
C.2	SpecAugment . . . . .	33
<b>D</b>	<b>Automatic Speech Recognition</b>	<b>34</b>
D.1	Overview . . . . .	34
D.2	Language Modeling . . . . .	34
<b>E</b>	<b>Connectionist Temporal Classification (CTC)</b>	<b>36</b>
E.1	Topology . . . . .	36
E.2	CTC Forward-Backward Algorithm . . . . .	38
E.3	CTC Decoding . . . . .	39
E.3.1	Decision Rules . . . . .	39
E.3.2	Greedy Search . . . . .	40
E.3.3	Time-synchronous Search . . . . .	40
E.3.4	Label-synchronous Search . . . . .	41

## A Self-Supervised Speech Representation Learning

This section aims to provide a comprehensive overview of self-supervised learning (SSL) behaviors for speech representation, which is the rationale to motivate our usage of wav2vec 2.0 in our multimodal fusion architecture.

### A.1 wav2vec 2.0 Architecture

wav2vec 2.0, introduced by Baevski et al. [6] (see Figure 4), is a SSL framework designed to learn speech representations directly from raw waveforms. The model enables data-efficient training of ASR systems by separating the pre-training and fine-tuning phases. The architecture consists of three key components: a feature encoder, a context network, and a quantization module.

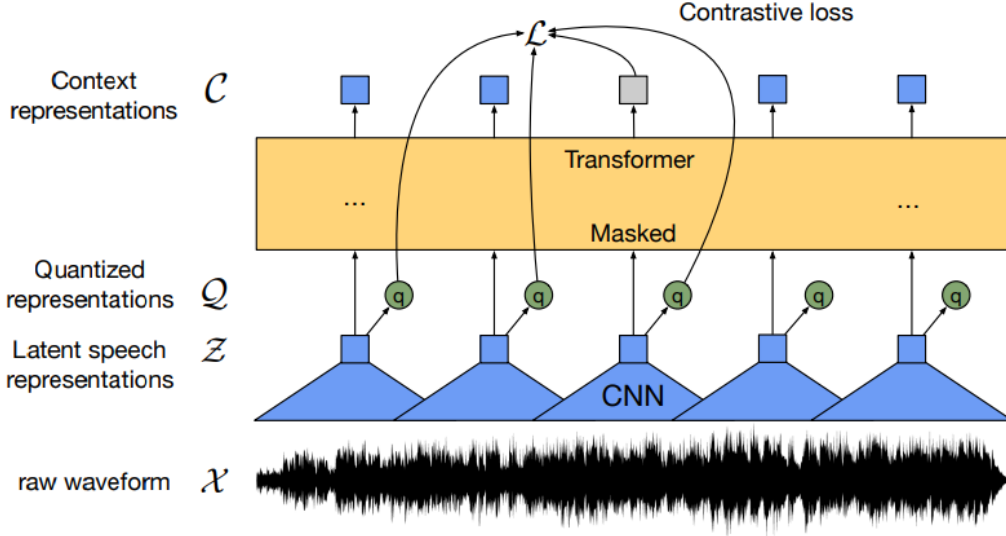


Figure 4: wav2vec 2.0 architecture

#### A.1.1 Self-Supervised Pre-training

**Wave normalization:** The raw audio waveform  $\mathcal{A}_T \in \mathbb{R}^T$  is first normalized to the range between 0 and 1 by the wave normalization function WaveNorm before being pushed into the feature extractor, as shown in Equation 8.

$$\begin{aligned} \mathcal{A}_T &:= \mathcal{A}_T^{\text{WaveNorm}} = \text{WaveNorm}(\mathcal{A}_T) \\ &= \text{LayerNorm}(\mathcal{A}_T) \in \mathbb{R}^T \end{aligned} \quad (8)$$

WaveNorm could be either layer normalization LayerNorm [3] or batch normalization BatchNorm [39].

**Feature Encoder:** The feature encoder is a multi-layer convolutional neural network (CNN) that processes raw audio input  $\mathcal{A}_T \in \mathbb{R}^T$  to produce a sequence of latent audio representations  $z_a \in \mathbb{R}^{T'_a \times d_a}$ , where  $T'_a < T$  due to down-sampling and  $d_a$  is the feature dimension. Typically, the encoder consists of 7 convolutional layers with GELU activations [31] and layer normalization [3].

$$z_a = \text{FeatureEncoder}(\mathcal{A}_T) \quad (9)$$

To be specific:

$$\begin{aligned} z_a &= \text{FeatureEncoder}(\mathcal{A}_T) \\ &= \text{FFW} \circ \text{CNNs} \circ \text{WaveNorm}(\mathcal{A}_T) \end{aligned} \quad (10)$$



**Context Network:** The context network comprises a stack of Transformer encoder layers that model temporal dependencies in the latent feature sequence. These layers generate context-aware representations  $c_a \in \mathbb{R}^{T'_a \times d_a}$  by applying self-attention and feedforward (FFW) operations.

$$c_a := \text{Transformer}(z_a) \quad (11)$$

Each Transformer block includes multi-head self-attention (MHSA), FFW sublayers, residual connections, and layer normalization. These allow the model to capture long-range dependencies in speech signals.

In an arbitrary  $l$ -th transformer layer, the output  $c_{al}$  is briefly defined as:

$$\begin{aligned} c_{al} &= \text{Transformer}(c_{al-1}) \\ &= \text{FFW} \circ \text{MHSA}(c_{al-1}) \end{aligned} \quad (12)$$

where MHSA is multi-head attention which is a function defined by self-attention functions SA:

$$\text{MHSA}(c_{al-1}) = \text{SA}(c_{al-1}) + c_{al-1} \quad (13)$$

Then, we have a full equation for an arbitrary  $l$ -th Transformer layer:

$$\begin{aligned} c_{al} &= \text{FFW}(\text{MHSA}(c_{al-1})) + \text{MHSA}(c_{al-1}) \\ &= \text{FFW}(\text{SA}(c_{al-1}) + c_{al-1}) + [\text{SA}(c_{al-1}) + c_{al-1}] \end{aligned} \quad (14)$$

For layer-wise formulation, the 0-th Transformer layer (the first layer) is connected to the feature encoder, which is defined as:

$$c_{a0} = \text{Transformer}(z_a) \quad (15)$$

Given an  $L$ -Transformer-layer wav2vec 2.0 architecture, the  $L - 1$ -th Transformer layer (the final layer) is defined as a chain function as:

$$\begin{aligned} c_{aL-1} &= \text{Transformer}(c_{aL-2}) \\ &= \text{Transformer} \circ \text{Transformer} \circ \dots \circ \text{Transformer}(x_0^T) \\ &= \text{Transformer} \circ \text{Transformer} \circ \dots \circ \text{Transformer} \circ \text{Transformer}(z_a) \end{aligned} \quad (16)$$

where  $L$  is the total number of Transformer layers in the encoder, layer indices start from 0 to  $L - 1$ .

**Quantization Module:** To formulate a contrastive learning task, the model discretizes the latent features  $z_a$  into quantized targets  $q_{t_a} \in \mathbb{R}_a^d$  using a quantization module. This module employs Gumbel-softmax-based vector quantization with multiple codebooks.

Let  $G_a$  be the number of codebooks and  $V_a$  the number of entries per codebook. Each quantized representation is obtained as:

$$q_{t_a} = \text{Concat}(e_{g_{1a}}, e_{g_{2a}}, \dots, e_{g_{G_a}}) \quad (17)$$

where each  $e_{g_{ia}} \in \mathbb{R}^{d_a/G_a}$  is a learned embedding vector selected from the  $i$ -th codebook.

**Pre-training Objective:** During pre-training, the model masks a subset of the latent features and uses the context representations to identify the corresponding quantized targets from a pool of negatives. The primary learning signal is a contrastive loss defined as:

$$\mathcal{L}_{\text{contrastive}} = -\log \frac{\exp(\text{sim}(c_{t_a}, q_{t_a})/\kappa_a)}{\sum_{q'_a \in \mathcal{Q}_{t_a}} \exp(\text{sim}(c_{t_a}, q'_a)/\kappa_a)} \quad (18)$$

where  $\text{sim}(\cdot, \cdot)$  is cosine similarity,  $\kappa_a$  is a temperature hyperparameter, and  $\mathcal{Q}_{t_a}$  contains one true quantized vector and multiple negatives.

To encourage diversity across codebook entries, a diversity loss is added:

$$\mathcal{L}_{\text{diversity}} = \frac{G_a}{V_a} \sum_{g_a=1}^{G_a} \sum_{v_a=1}^{V_a} p_{g_a, v_a} \log p_{g_a, v_a} \quad (19)$$

where  $p_{g,v}$  is the average selection probability of the  $v$ -th entry in the  $g$ -th codebook. The total pre-training loss is:

$$\mathcal{L}_a = \mathcal{L}_{\text{contrastive}} + \alpha_a \cdot \mathcal{L}_{\text{diversity}} \quad (20)$$

with  $\alpha_a$  being a tunable weight.

### A.1.2 Supervised Fine-tuning

Once pre-trained, the context representations  $c_a$  are used for supervised ASR by appending a randomly initialized linear projection layer and training the model with a Connectionist Temporal Classification (CTC) loss [27, 25]. The entire model is fine-tuned end-to-end using a small amount of labeled data, enabling high accuracy even with limited supervision.

See Section D for details of supervised fine-tuning ASR.

## A.2 Rationale

The rapid advancement of deep learning has revolutionized the field of speech processing, enabling significant improvements in tasks such as ASR, speaker identification, emotion recognition, and speech synthesis. Traditionally, these tasks have relied heavily on supervised learning [80, 57, 93, 94], which requires large volumes of labeled data. However, obtaining high-quality labeled speech data is both expensive and time-consuming, especially when considering the wide variability in languages, accents, recording conditions, and speaker characteristics. This has led to a growing interest in SSL, an approach that leverages vast amounts of unlabeled data to learn meaningful representations without explicit annotations [4, 20, 62, 82].

Self-supervised speech representation learning aims to extract high-level, informative features from raw audio signals by solving pretext tasks derived from the inherent structure of the data. These pretext tasks are designed such that solving them requires understanding relevant patterns in the speech signal, such as phonetic content, prosody, or speaker identity [76, 88, 5]. Once trained, the resulting representations can be fine-tuned or directly applied to downstream tasks with minimal supervision, significantly reducing the dependence on labeled data [21, 52, 49].

Recent breakthroughs in SSL, particularly inspired by advances in natural language processing (e.g., BERT [19], GPT [65, 9, 72]) and computer vision (e.g., SimCLR [14], MoCo [30]), have led to the development of powerful speech models such as wav2vec [76], HuBERT [34], and WavLM [13]. These models have demonstrated state-of-the-art performance across a wide range of speech-related benchmarks, often outperforming fully supervised counterparts when only limited labeled data is available [6, 76]. Moreover, SSL has opened new avenues for learning more generalizable, robust, and multilingual representations [7, 22, 75].

The wav2vec 2.0 transformer exhibits an autoencoder-like behavior [81, 16]: the representations start deviating from the input speech features followed by a reverse trend where even deeper layers become more similar to the input, as if reconstructing the input.

1. The layer-wise progression of representations exhibits an acoustic-to-linguistic hierarchy: lower layers encode acoustic features, followed sequentially by phonetic, word identity, and semantic information, before reversing this trend in the upper layers, as shown in Figure 5.
2. ASR fine-tuning disrupts this autoencoder-like behavior in the upper layers, enhancing their capacity to encode lexical information.
3. The initial transformer and final CNN layers show high correlation with mel spectrograms, indicating convergence toward human-engineered features.
4. The SSL model encodes some semantic content.
5. The final two layers often deviate from preceding patterns

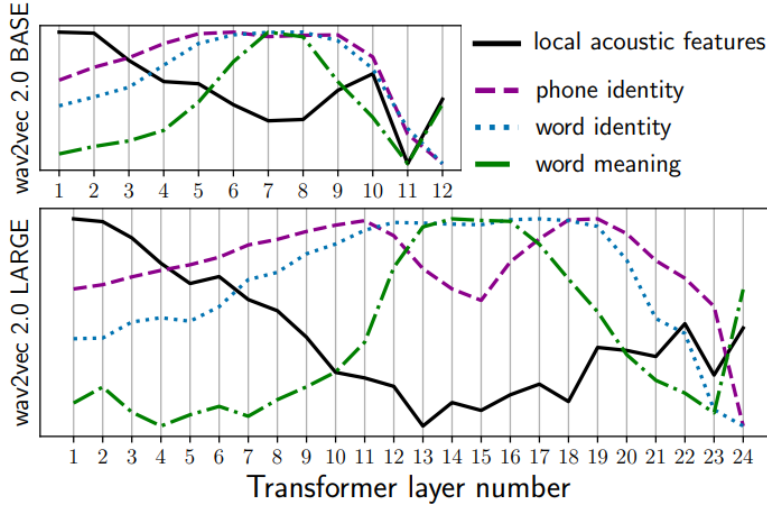


Figure 5: Visualization of properties encoded at different wav2vec 2.0 layers.

### A.3 Analysis Methods

#### A.3.1 Canonical Correlation Analysis

Canonical Correlation Analysis (CCA) [33] is a classical statistical technique designed to quantify the linear relationships between two multivariate random variables. Given two sets of continuous-valued random vectors, CCA identifies pairs of canonical directions—one for each set—such that the correlation between the projections of the vectors onto these directions is maximized. This results in a sequence of canonical correlation coefficients that capture the degree of linear alignment between the two representational spaces.

In the context of SSL models such as wav2vec 2.0, CCA has proven to be a valuable tool for analyzing the internal structure of learned representations. wav2vec 2.0 encodes raw audio waveforms into hierarchical feature representations through a series of convolutional and Transformer layers. By applying CCA, we can quantify the representational similarity across layers of the model, offering insight into how acoustic and linguistic information is progressively abstracted.

Pasad et al. [66] employ CCA in two complementary ways. First, they compute pairwise CCA scores between different layers of the wav2vec 2.0 Transformer encoder to investigate the evolution and redundancy of learned features. This helps assess whether certain layers exhibit similar information encoding patterns, or whether deeper layers introduce significant representational shifts.

Second, Pasad et al. [66] apply CCA to measure the similarity between the internal layer representations of wav2vec 2.0 and external reference vectors. These reference vectors include pre-trained word embeddings (e.g., Word2Vec [61] or GloVe [68]) and low-level acoustic features (e.g., Mel-frequency cepstral coefficients or log-Mel spectrograms). This cross-modal comparison enables us to determine the extent to which specific Transformer layers align with either phonetic-level acoustic information or semantically-rich linguistic abstractions. Through this analysis, we gain deeper interpretability into how wav2vec 2.0 encodes and transitions between speech and language representations [63, 48, 74].

#### A.3.2 Mutual Information Estimation

While CCA is a natural choice for quantifying relationships between pairs of continuous-valued vector representations, it is limited to capturing linear correlations and does not generalize well to the dependence between learned representations and categorical linguistic units such as phones or words. Instead, Pasad et al. [66] adopt *mutual information* (MI) as a more general measure of statistical dependence between the latent representations  $\mathbf{y}_{\text{phn}}$  or  $\mathbf{y}_{\text{wrd}}$ —which are extracted from intermediate layers of the wav2vec 2.0 model—and their corresponding ground-truth phoneme or word labels.

Since the model outputs continuous-valued representations, Pasad et al. [66] follow prior work [6, 2] and discretize them using clustering (e.g.,  $k$ -means), thereby enabling estimation of mutual information via co-occurrence statistics.

The resulting MI metrics, denoted as **MI-phone** and **MI-word**, quantify the amount of phonetic or lexical information preserved in the internal feature representations. Higher MI indicates a stronger correlation between learned representations and linguistic targets, providing insight into the degree of linguistic abstraction encoded by the model during SSL.

#### A.4 Findings of Self-Supervised Representation Learning

##### A.4.1 Reconstruction Behavior

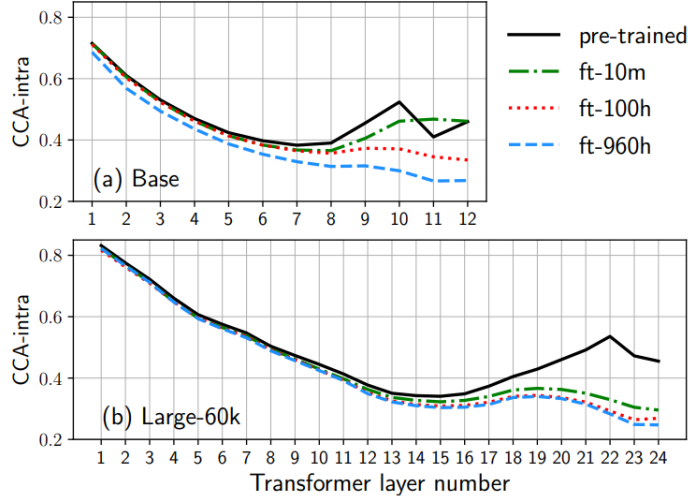


Figure 6: CCA similarity with local features

Figure 6 presents a comparison of transformer layer representations with the local features extracted by the CNN module (layer 0), using CCA similarity. The pre-trained model (solid black curve) exhibits an autoencoder-like pattern: representations initially diverge from the input features with increasing depth, but subsequently reconverge in deeper layers, indicating a reconstruction-like behavior. This trend is disrupted in the final two layers (see Section below). Given that the training objective involves distinguishing a masked input segment from distractors, it is expected that the final layers reencode representations similar to the input. A comparable pattern—termed context encoding and reconstruction—has been previously observed in BERT for masked language modeling objectives [83].

##### A.4.2 Encoded Acoustic-Linguistic Information

Pasad et al. [66] analyzed how specific properties are encoded across different model layers. It is important to note that all experiments are conducted using features extracted from short temporal spans, corresponding to frame-, phone-, or word-level segments. Any observed increase in the amount of encoded "information" across layers for these local representations can be attributed to the contextualization enabled by the self-attention mechanism, which allows each frame-level output to incorporate information from the entire utterance. Conversely, a reduction in localized "information" across layers may result from de-localization, wherein the representation becomes increasingly distributed and less confined to the original temporal segment.

**Frame-level acoustic information:** Figure 7 presents the layer-wise CCA similarity between filterbank (fbank) features and the representations from the wav2vec 2.0 Base model. In the initial layers, the correlation increases progressively with depth. A similar trend is observed for the Large models, which exhibit high CCA values ( $> 0.75$ ) between layers C4 and T2. These results suggest that the model implicitly learns representations analogous to fbank features, indicating the potential

for simplifying wav2Vec 2.0 by directly using fbank inputs. However, to our best knowledge, the potential suggested by Pasad et al. [66] has not been empirically proven yet.

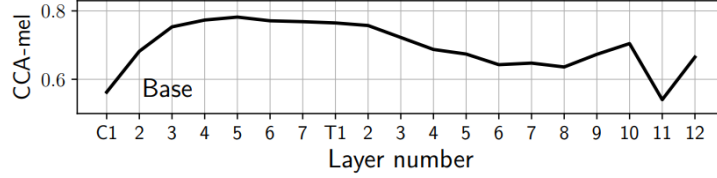


Figure 7: CCA similarity between layer representations and fbanks; Ci: CNN layer i, Tj: transformer layer j.

**Phonetic information:** Pasad et al. [66] quantify the phonetic information encoded in the pre-trained model using two metrics: mutual information with phone labels (MI-phone) and canonical correlation analysis with AGWEs (CCA-agwe), as visualized in Figure 8. Given that AGWEs are designed to represent phonetic content, the similarity in trends between the MI-phone and AGWE curves supports this expectation. In the wav2vec 2.0 Base model, phonetic information peaks around layers 6–7. We, to the best of our knowledge, found this behavior consistent with prior findings [35] which analyzed the behavior of HuBERT [34]. In contrast, the Large-60k model exhibits prominent phonetic encoding at layers 11 and 18/19, with a notable decline in intermediate layers.

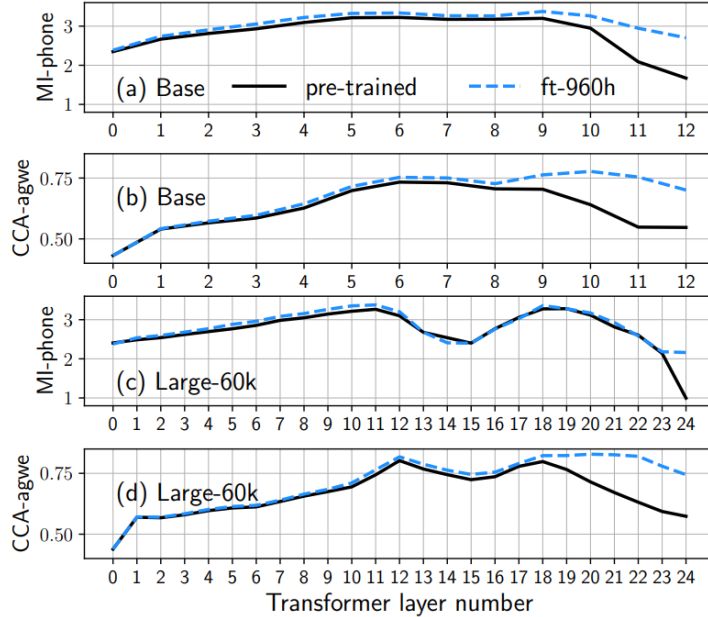


Figure 8: MI with phone labels (max: 3.6) and CCA similarity with AGWE.

**Word identity:** Figure 9 presents the MI between layer representations and word labels. For the wav2vec 2.0 Base model, the observed trends resemble those of MI with phone labels (Figure 8). In the Large-60k model (Figure 9), word identity is consistently encoded across layers 12 to 18, without the decline observed in the MI-phone curve. This behavior shows that, to the best understanding of Pasad et al. [66]’s work, MI-word and word discrimination are always highly correlated.

#### A.4.3 Word Meaning Representation

Although certain linguistic features appear critical for the model to solve the SSL objective, it remains unclear whether semantic content—specifically word meaning—is among them. To investigate this, Pasad et al. [66] assess the encoding of word meaning in wav2Vec 2.0 by computing the CCA

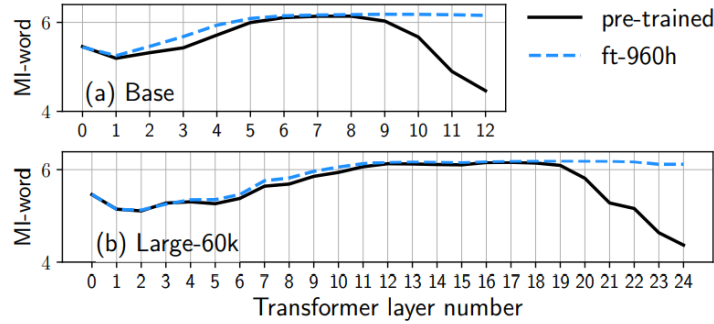


Figure 9: MI with word labels (max: 6.2).

similarity between word segment representations and GloVe embeddings [68], as illustrated in Figure 10. The results indicate that the middle layers—layers 7–8 in the Base model and 14–16 in the Large-60k model—encode the richest contextual information. Notably, the narrower plateau of peak performance in these curves compared to the MI curves in Figure 9 suggests that central layers are more specialized in capturing semantic content, whereas peripheral layers primarily encode lower-level linguistic features without semantic abstraction.

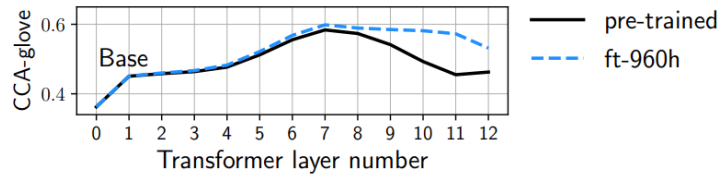


Figure 10: CCA similarity with GloVe embeddings [68].

#### A.4.4 Fine-tuning Effect

As shown in Figure 6 (CCA-intra), fine-tuning disrupts the autoencoder-like behavior of the model. Post fine-tuning for ASR, the deeper layers, which previously aimed to reconstruct the input, increasingly diverge from it, indicating a shift toward learning task-specific representations. Additionally, Figure 11 reveals that the upper layers undergo the most significant changes during fine-tuning, implying that the pre-trained model may provide suboptimal initialization for these layers in ASR tasks. This observation, to the best of our knowledge, aligns with findings in BERT language modelling [98], where re-initialization of top layers prior to fine-tuning improves performance.

The results also suggest that fine-tuning with character-level CTC loss [27] is more strongly associated with encoding word identity than phone identity, as anticipated.

We observed that the final layers of wav2Vec 2.0 undergo the most substantial modifications during fine-tuning (Figure 11) and exhibit reduced encoding of linguistic information relevant to ASR. These findings suggest that certain upper layers may offer suboptimal initialization for downstream ASR tasks.



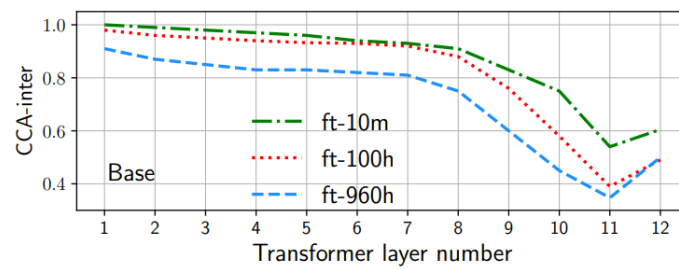


Figure 11: CCA similarity between each layer of a pre-trained model and the same layer of fine-tuned models.

## B Weakly Supervised Speech Representation Learning

### B.1 Attention Encoder Decoder (AED)

As for AED models, Whisper architecture is shown in Figure 12, and Deepgram architecture is shown in Figure 13.

#### B.1.1 Whisper Architecture

#### B.1.2 Deepgram Nova-2 Architecture

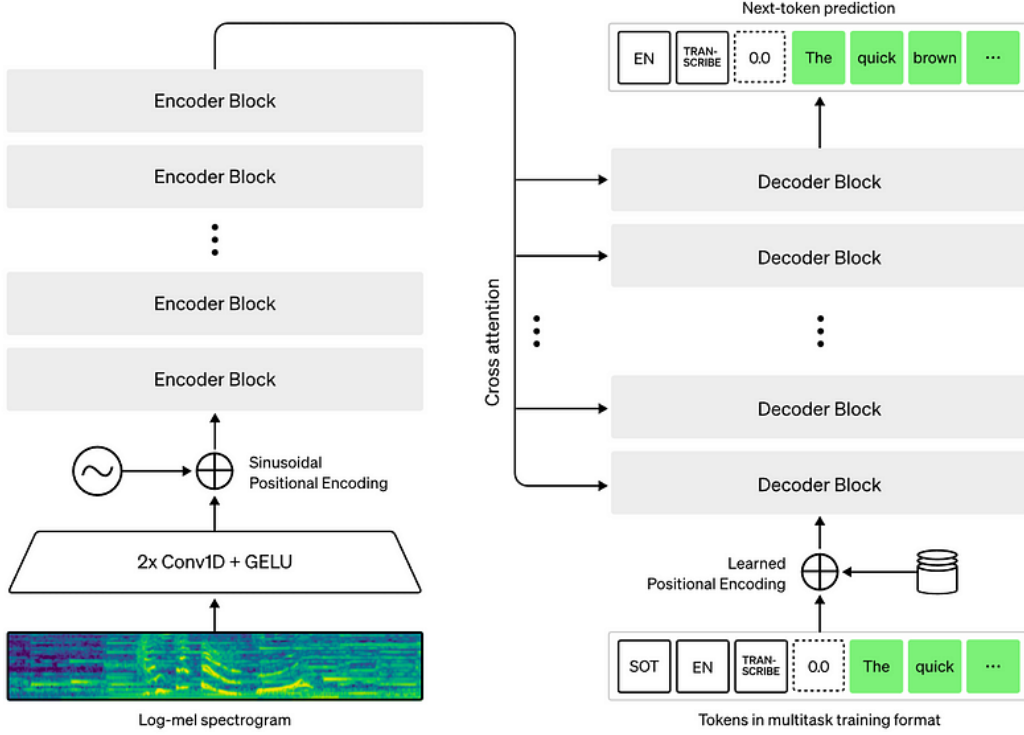


Figure 12: **OpenAI's Whisper architecture.** Whisper is a Transformer-based AED architecture, using MFCC features as input.

An ASR model is used to transcribe speech into text by mapping an audio signal  $x_1^T := x_1, x_2, \dots, x_T$  of length  $T$  to the most likely word sequence  $w_1^N$  of length  $N$ . The word sequence probability is described as:

$$p(w_1^N | x_1^T) = \prod_{n=1}^N p(w_n | w_1^{n-1}, x_1^T). \quad (21)$$

In the ASR encoder-decoder architecture, given  $D$  as the feature dimension size, the input audio signal matrix could be described as  $x_1^T \in \mathbb{R}^{T \times D_{input}}$ . When simplified, downsampling before or inside the encoder - conducted by a fixed factor, such as striding in a Convolutional Neural Network (CNN) - is removed. Thus, the encoder output sequence is as follows:

$$h_1^T = \text{Encoder}(x_1^T) \in \mathbb{R}^{T \times D_{encoder}}. \quad (22)$$

Using a stack of Transformer ( $\mathcal{T}$ ) blocks [? ], the encoder output sequence is described as function composition:

$$h_1^T = \mathcal{T}_0 \circ \dots \circ \mathcal{T}_{N_{encLayers}}(x_1^T). \quad (23)$$

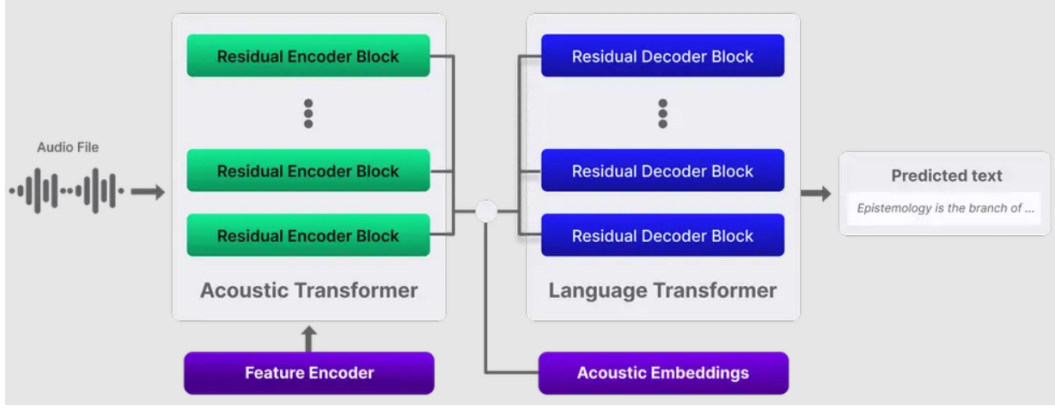


Figure 13: **Deepgram's Nova-2 architecture.** To our best understanding of Deepgram's documentation, Deepgram's Nova-2 is a Transformer-based AED architecture, using raw waveform as input instead of MFCC like Whisper. Feature extraction from raw waveform is probably conducted by a learnable feature encoder, e.g. a block of CNNs like wav2vec 2.0. Between encoder-decoder space, (unknown) acoustic embeddings are probably added as cross-attention.

In the decoder, the probability for each single word is defined as:

$$\begin{aligned} p(w_n | w_1^{n-1}, x_1^T) &= p(w_n | w_1^{n-1}, h_1^T(x_1^T)) \\ &= p(w_n | w_1^{n-1}, h_1^T). \end{aligned} \quad (24)$$

Based on Equation 21, the word sequence probability given the output of encoder is described as:

$$p(w_1^N | x_1^T) = \prod_{n=1}^N p(w_n | w_1^{n-1}, h_1^T). \quad (25)$$

Then, decoder hidden state is formulated as:

$$g_n = \mathcal{F}(g_{n-1}, w_{n-1}, c_n) \in \mathbb{R}^{D_g}, \quad (26)$$

where  $\mathcal{F}$  is neural network;  $D_g$  is hidden state dimension; and  $c_n$  is context vector, e.g. weighted sum of encoder outputs via attention mechanism.

The attention mechanism in the decoder is described via 3 components: context vector  $c_n$ , attention weights  $\alpha_{n,t}$ , and attention energy  $e_{n,t}$ :

$$\begin{aligned} c_n &= \sum_{t=1}^T \alpha_{n,t} h_t \in \mathbb{R}^{D_{encoder}}, \\ \alpha_{n,t} &= \frac{\exp(e_{n,t})}{\sum_{t'=1}^T \exp(e_{n,t'})} \\ &= \text{Softmax}_T(\exp(e_{n,t})) \in \mathbb{R}, \\ e_{n,t} &= \text{Align}(g_{n-1}, h_t) \in \mathbb{R} \\ &= W_2 \cdot \tanh(W_1 \cdot [g_{n-1}, h_t]), \end{aligned} \quad (27)$$

where  $n$  is decoder step;  $t$  is encoder frame;  $\alpha \in \mathbb{R}^{T \times N}$  is attention weight matrix;  $\alpha_n \in \mathbb{R}^T$  is normalized probability distribution over  $t$ ;  $\text{Softmax}_T$  is Softmax function over spatial dimension  $T$ , not feature dimension;  $W_1 \in \mathbb{R}^{(D_g + D_{encoder}) \times D_{key}}$ ;  $W_2 \in \mathbb{R}^{D_{key}}$ .

In the decoding, the output probability distribution over vocabulary is defined as:

$$\begin{aligned} p(w_n = * | w_1^{n-1}, h_1^T) \\ = \text{Softmax}(\text{MLP}(w_{n-1}, g_n, c_n)) \in \mathbb{R}^N, \end{aligned} \quad (28)$$

where *MLP* is Multi-layer Perceptron.

To train an AED model, sequence-level frame-wise cross-entropy loss is employed:

$$\begin{aligned}\mathcal{L}_{AED} &= - \sum_{(x_1^T, w_1^N)} \log p(w_1^N | x_1^T) \\ &= - \sum_{(x_1^T, w_1^N)} \sum_{n=1}^N \log p(w_n | w_1^{n-1}, x_1^T).\end{aligned}\tag{29}$$

During beam search, the auxiliary quantity for each unknown partial string (tree of partial hypotheses)  $w_1^n$  is defined as:

$$\begin{aligned}Q(n; w_1^n) &:= \prod_{n'=1}^n p(w_{n'} | w_0^{n'-1}, x_1^T) \\ &= p(w_n | w_0^{n-1}, x_1^T) \cdot Q(n-1, w_1^{n-1}).\end{aligned}\tag{30}$$

After discarding the less likely hypotheses in the beam search, the word sequence probability is calculated by the best hypothesis:

$$p(w_1^N | x_1^T) = Q(N; w_1^N).\tag{31}$$

## B.2 Rationale

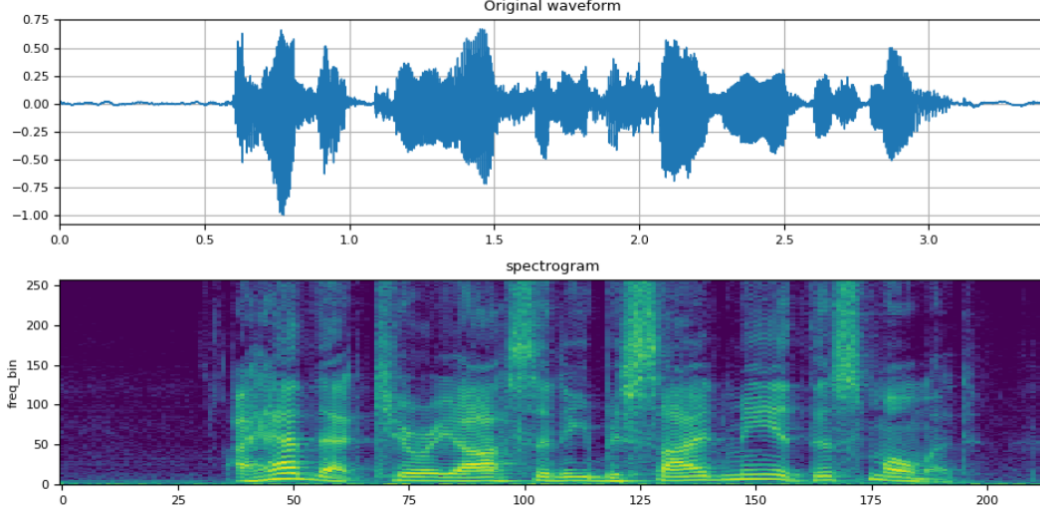


Figure 14: **MFCC visualization.** The computation of MFCCs begins by dividing the original waveform into overlapping 20ms frames.

## C Raw waveform vs MFCC

### C.1 Mel-Frequency Cepstral Coefficients (MFCCs)

MFCC serves as a compact representation of the audio signal’s spectral properties. The computation of MFCCs begins by dividing the input signal  $x_1^T := x_1, x_2, \dots, x_T$  into overlapping frames, as visualized in Figure 14<sup>1</sup>.

**Pre-emphasis:** The audio signal, sampled at 16 kHz with a step size of 10 ms, is processed by extracting 160 consecutive samples from the Pulse Code Modulation (PCM) waveform for each frame. These 10 ms frames are non-overlapping, ensuring that stacking adjacent vectors avoids discontinuities. The 16-bit quantized samples, which span the integer range from  $-2^{15}$  to  $+2^{15}$ , must be normalized to a numerically stable range. This normalization is achieved by applying mean and variance normalization, either globally across the entire training dataset or on a per-utterance basis. A commonly employed processing technique, known as high-frequency pre-emphasis, can be implemented by computing the differences between adjacent samples, as illustrated below:

$$x'_t = x_t - x_{t-1} \in \mathbb{R} \quad (32)$$

A sequence of  $16 \text{ kHz} \times 10 \text{ ms} = 160$  pre-emphasized waveform samples can then be considered a feature vector:

$$\hat{x}_t = x'^t_{t-160+1} \in \mathbb{R}^{160} \quad (33)$$

**Amplitude spectrum - FFT:** The short-time Fourier transform (STFT) is applied to overlapping windows with a duration of 25 ms. Given a sampling rate of 16 kHz, this window length corresponds to  $25 \text{ ms} \times 16 \text{ kHz} = 400$  samples. To facilitate computation using the fast Fourier transform (FFT), the sample count is zero-padded to the next power of two, resulting in  $2^9 = 512$ .

$$\begin{aligned} \mathcal{Z}_t &\in \mathbb{R}^{512} \\ &= \begin{bmatrix} x'^{t'}_{t-400+1} & x'^{t'}_{t-400+2} & \dots & x'^{t'} & \underbrace{0 \dots 0}_{\text{zero-padding}} \end{bmatrix} \end{aligned} \quad (34)$$

<sup>1</sup>? J’s Dissertation at RWTH Aachen University described MFCC more comprehensively. MFCC visualization image is retrieved from Pytorch library.

The extended sample vector is weighted using a Hann window, which exhibits smaller side lobes in the amplitude spectrum compared to a rectangular window:

$$\begin{aligned} \mathbf{w}^{(n)} &= 0.5 - 0.5 \cos \left( \frac{2\pi(n-1)}{512-1} \right), \\ 1 &\leq n \leq 512 \end{aligned} \quad (35)$$

$$\mathbf{d}_t^{(n)} = \mathbf{z}_t^{(n)} \cdot \mathbf{w}^{(n)} \quad (36)$$

While the discrete STFT could be done directly by evaluating the sum

$$\begin{aligned} \mathcal{S}_t^{(\mathbb{F})} &= \sum_{n=0}^{512-1} \mathbf{d}_t^{(n)} \cdot \exp \left( -j \frac{2\pi}{512} \mathbb{F} n \right), \\ 1 &\leq \mathbb{F} \leq 512 \end{aligned} \quad (37)$$

the complexity can be reduced from  $\mathcal{O}(N^2)$  to  $\mathcal{O}(N \log N)$  by applying the fast Fourier transform.

The 512-FFT results in a 257-dimensional vector because of the symmetry of the amplitude spectrum of a real-valued signal. The phase spectrum is removed.

$$\begin{aligned} \hat{\mathbf{x}}_t &= \begin{bmatrix} |\mathcal{S}_t^{(0)}| & |\mathcal{S}_t^{(1)}| & \dots & |\mathcal{S}_t^{(512/2)}| \end{bmatrix} \\ &\in \mathbb{R}^{512/2+1} \end{aligned} \quad (38)$$

**MFCC:** The MFCC feature extraction is based on the STFT of the pre-emphasized speech signal [?]. It considers the nonlinear sensitivity of human auditory perception to variations in frequency. This is evidenced that the filter bank used to integrate the magnitude spectrum  $|\mathcal{S}_t^{(\mathbb{F})}|$  consists of  $\mathbb{I}$  filters equidistantly spaced on the mel scale. The mel scale is a logarithmically scaled frequency axis. The  $k$ -th frequency bin of the FFT centered around  $\mathbb{F}_k$  Hz is then mapped to  $\tilde{\mathbb{F}}_k$  on the mel scale:

$$\mathbb{F}_k = \frac{k}{512} \cdot \mathbb{F}_s \quad (39)$$

$$\tilde{\mathbb{F}}_k = 2595 \cdot \log_{10} \left( 1 + \frac{\mathbb{F}_k}{700 \text{ Hz}} \right) \quad (40)$$

The filter center  $\tilde{\mathbb{F}}_c^{(i)}$  of the  $i$ -th triangular filter is then placed at  $i \cdot \tilde{\mathbb{F}}_b$ , where the bandwidth  $\tilde{\mathbb{F}}_b$  corresponds to  $\tilde{\mathbb{F}}_{512}/\mathbb{I}$ . With these parameters, the coefficients of the  $i$ -th triangular filter can be calculated explicitly as a piecewise linear function and stored in a weight vector  $\mathbf{v}_i \in \mathbb{R}^{N/2+1}$ .

By applying discrete cosine transform (DCT), the MFCC features are extracted from the logarithm filter outputs:

$$\mathcal{X}_t^{(i)} = \log_{10} \left( \sum_{\mathbb{F}=0}^{512} |\mathcal{S}_t^{(\mathbb{F})}| \mathbf{v}_i^{(\mathbb{F})} \right) \quad (41)$$

$$\mathbf{c}_{m,i} = \cos \left( \frac{\pi m(i+0.5)}{\mathbb{I}} \right) \quad (42)$$

$$\mathbf{c}_t^{(m)} = \sum_{i=0}^{\mathbb{I}-1} \mathbf{c}_{m,i} \mathcal{X}_t^{(i)} \quad (43)$$

$$\hat{\mathbf{x}}_t = \begin{bmatrix} \mathbf{c}_t^{(0)} & \mathbf{c}_t^{(1)} & \dots & \mathbf{c}_t^{(\mathbb{I}-1)} \end{bmatrix} \in \mathbb{R}^{\mathbb{I}} \quad (44)$$



## C.2 SpecAugment

SpecAugment [?] is a data augmentation technique for ASR that manipulates spectrograms to improve model robustness by randomly applying masking in consecutive frames in the time axis as well as consecutive dimensions in the feature axis. It performs three main transformations<sup>2</sup>: time warping, frequency masking, and time masking.

Figure 15 shows examples of the individual augmentations applied to a single input.

**Time Masking:** Given an audio signal  $x_1^T := x_1, x_2, \dots, x_T$  of length  $T$ . Time masking is masking of  $\tau$  successive time steps  $[t, t + \tau)$ , where we set:

$$(x_t, \dots, x_{t+\tau}) := 0 \quad (45)$$

where  $\tau$  is the masking window selected from a uniform distribution from 0 to the maximum time mask parameter  $\mathbb{T}\mathbb{M}$ . The time position  $t$  is picked from another uniform distribution over  $[0, T)$  such that the maximum sequence length  $T$  is not exceeded (i.e. if  $t + \tau > T$ , we set it to  $T$ ).

**Frequency Masking:** Frequency masking is applied such that  $\phi$  consecutive frequency channels  $[f, f + \phi)$  are masked, where  $\phi$  is selected from a uniform distribution from 0 to the frequency mask parameter  $\mathbb{F}\mathbb{M}$ , and  $f$  is chosen from  $[0, \nu)$ , where  $\nu$  is the input feature dimension, e.g. the number of MFCC channels. For raw waveform as input,  $\nu = 1$ . Similar to time masking, if  $f + \phi > \nu$ , we set it to  $f = \nu$ .

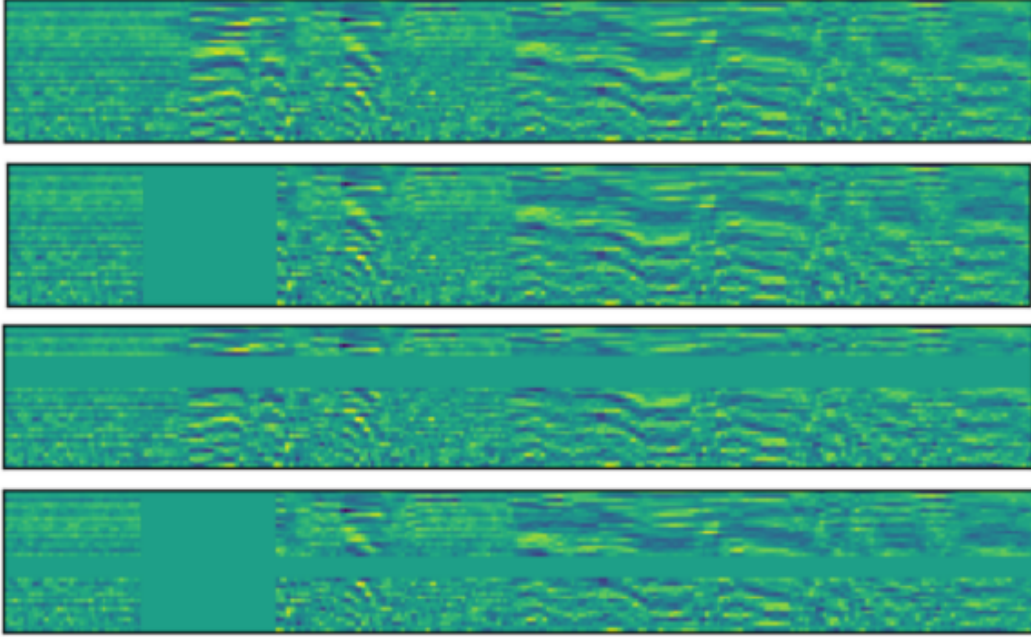


Figure 15: **SpecAugment visualization.** From top to bottom, the figures show the spectrogram of the input audio with no data augmentation, time masking, frequency masking and both masking applied.

<sup>2</sup>? ] analyzed deeply in end-to-end ST. ? ] stated that time warping is the most expensive and the least influential, we do not include it here

## D Automatic Speech Recognition

### D.1 Overview

ASR is traditionally formulated within a statistical framework. Formally, let  $x_1^T$  denote a sequence of acoustic feature vectors, where  $x_t \in \mathbb{R}^D$  for  $1 \leq t \leq T$ , extracted from the raw speech waveform via a feature extraction process (e.g. MFCC). Let  $V$  represent the vocabulary set. Typically, each vector  $x_t$  encodes information corresponding to a fixed-duration frame of the speech signal, such as 10 milliseconds.

By Bayes' decision rule [8], given the observed acoustic feature sequence  $x_1^T$ , an ASR system aims to determine the most probable word sequence  $\hat{w}_1^{\hat{N}} \in V^*$  such that:

$$\hat{w}_1^{\hat{N}} = \arg \max_{N, w_1^N} p(w_1^N | x_1^T) \quad (46)$$

$$= \arg \max_{N, w_1^N} \left[ \frac{p(x_1^T | w_1^N) \cdot p(w_1^N)}{p(x_1^T)} \right] \quad (47)$$

$$= \arg \max_{N, w_1^N} \left[ \frac{p(x_1^T | w_1^N) \cdot p(w_1^N)}{\text{const}(w_1^N)} \right] \quad (48)$$

$$= \arg \max_{N, w_1^N} [p(x_1^T | w_1^N) \cdot p(w_1^N)] \quad (49)$$

where  $p(w_1^N | x_1^T)$  denotes the posterior probability of the word sequence  $w_1^N$  of  $N$  length conditioned on the acoustic features  $x_1^T$ .

The effectiveness of an ASR system is typically quantified using the Word Error Rate (WER), defined for a reference word sequence  $\tilde{w}_1^{\tilde{N}}$  and a hypothesis  $w_1^N$  produced by the system as:

$$\text{WER} = \frac{S_w + D_w + I_w}{\tilde{N}_w}, \quad (50)$$

where  $S_w$ ,  $D_w$ , and  $I_w$  represent the minimal number of *substitution*, *deletion*, and *insertion* operations, respectively, required to transform the reference sequence into the hypothesis. The quantity  $S_w + D_w + I_w$  corresponds to the Levenshtein distance [50] between the two sequences. For an evaluation corpus containing multiple references, the numerator and denominator are computed by summing over all hypotheses and references, respectively. WER is typically reported as a percentage.

Conventional ASR architectures, as discussed in [64], employ the decision rule in Eq. 49, wherein the acoustic likelihood  $p(x_1^T | w_1^N)$  (the *acoustic model*) and the prior  $p(w_1^N)$  (the *language model*) are modeled independently. In this context, the acoustic model is instantiated by wav2vec 2.0, while the language model is often implemented using count-based methods [47].

### D.2 Language Modeling

We consider the task of language modeling due to its close relationship with ASR. A *language model* (LM) defines a probability distribution over a label sequence  $w_1^N$ , denoted as  $p_{LM}(w_1^N)$ . This probability is typically factorized in an autoregressive fashion, although alternative non-autoregressive modeling approaches have also been proposed [40, 19]:

$$p_{LM}(w_1^N) = \prod_{n=1}^N p_{LM}(w_n | w_0^{n-1}), \quad (51)$$

where the LM estimates the conditional probability  $p_{LM}(w_n | w_1^{n-1})$ . Traditional LMs rely on count-based methods under the  $k$ -th order Markov assumption, i.e.,  $p_{LM}(w_n | w_1^{n-1}) \approx p_{LM}(w_n | w_{n-k}^{n-1})$ . In contrast, contemporary neural LMs are designed to leverage the full left context to directly model  $p_{LM}(w_n | w_1^{n-1})$ . To ensure that the normalization condition  $\sum_{w_1^N} p_{LM}(w_1^N) = 1$  holds, all sequences are required to terminate with a special end-of-sequence (EOS) symbol.

The performance of an LM is commonly assessed via its perplexity (PPL) [42], which for a sequence  $w_1^N$  is defined as:

$$\text{PPL} = \left[ \prod_{n=1}^N p_{LM}(w_n | w_1^{n-1}) \right]^{-\frac{1}{N}} = \exp \left( -\frac{1}{N} \sum_{n=1}^N \log p_{LM}(w_n | w_1^{n-1}) \right). \quad (52)$$

This formulation generalizes to a corpus-level evaluation by averaging the negative log probabilities of all tokens (along with their left contexts) across the corpus. Perplexity can be interpreted as the average effective number of choices the LM considers when predicting the next token. Lower perplexity indicates a better-performing model.

In Hidden Markov Model (HMM)-based ASR systems, the LM is an integral component. Although sequence-to-sequence (seq2seq) models do not incorporate an LM explicitly, empirical results have demonstrated that incorporating an external LM during decoding can significantly reduce the WER [38, 32, 45], assuming no domain mismatch. Consequently, it is now standard practice to integrate an external LM into the decoding process of seq2seq ASR models, which is also the approach adopted in this thesis. In wav2vec 2.0 experiments, researchers usually consider three types of LMs: a count-based Kneser-Ney smoothed  $n$ -gram model [47], an LSTM-based LM [79], and a Transformer-based LM [41].

## E Connectionist Temporal Classification (CTC)

wav2vec 2.0 uses CTC to model, thus we provide an overview of CTC in this section.

### E.1 Topology

A CTC model [25] consists of an encoder network followed by a linear projection and a softmax activation layer. The encoder takes as input a sequence of acoustic feature vectors  $x_1^T$  and produces a corresponding sequence of hidden representations  $h_1^{T'}$ :

$$h_1^{T'} = \text{Encoder}(x_1^T) \quad (53)$$

where each encoding vector  $h_t \in \mathbb{R}^{D_{\text{enc}}}$  for  $1 \leq t \leq T'$ , and  $D_{\text{enc}}$  denotes the dimensionality of the encoder output. The length  $T'$  of the output sequence is typically less than or equal to  $T$ , due to potential downsampling mechanisms, i.e.,  $T' \leq T$ , and generally  $T' < T$ .

Let  $V$  denote the vocabulary of permissible labels, and let  $\varepsilon$  represent a special label not included in  $V$ . Define the extended label set as  $V' = V \cup \{\varepsilon\}$ , where  $\varepsilon$  is referred to as the *blank* label, typically interpreted as representing either silence or the absence of a label. The output of the encoder network is processed through a linear transformation followed by a softmax activation, yielding:

$$o_1^{T'} = \text{Softmax}(\text{Linear}(h_1^{T'})) \quad (54)$$

where  $o_t \in [0, 1]^{|V'|}$  for  $1 \leq t \leq T'$ . The  $k$ -th component of the output vector  $o_t$ , denoted  $o_{t,k}$ , corresponds to the probability of emitting the  $k$ -th label from  $V'$  at time step  $t$ :

$$o_{t,k} = p_t(v_k \mid h_1^{T'}), \quad (55)$$

with  $v_k \in V'$  and  $1 \leq k \leq |V'|$ . This formulation characterizes the output distribution of a CTC model, specifying a per-frame categorical distribution over the extended label set  $V'$ , including the blank label.

Given this frame-level distribution, the CTC model defines a probability distribution over all possible output label sequences  $w_1^N$  conditioned on the input  $x_1^T$ , formally expressed as  $p_{\text{CTC}}(w_1^N \mid x_1^T) := p_{\text{CTC}}(w_1^N \mid h_1^{T'})$ . To construct this distribution, define a *path* as a label sequence  $y_1^{T'}$  of length  $T'$  such that each  $y_t \in V'$  corresponds to a label emitted at time step  $t$ .

Under the CTC framework, a key assumption is that of *conditional independence* across time steps, implying that the joint probability of a path  $y_1^{T'}$  conditioned on the encoder outputs factorizes as follows:

$$p(y_1^{T'} \mid h_1^{T'}) = \prod_{t=1}^{T'} p_t(y_t \mid h_1^{T'}). \quad (56)$$

A path  $y_1^{T'}$  can be formally regarded as an *alignment* corresponding to an output label sequence. Specifically, let  $\mathcal{B} : (V')^* \rightarrow V^*$  denote the *collapse* function, which operates by first merging consecutive repeated labels and subsequently removing all blank symbols. For instance, consider the examples:

$$\mathcal{B}(\varepsilon\varepsilon c c c \varepsilon a \varepsilon a a a \varepsilon t t t t \varepsilon) = \mathcal{B}(c c c \varepsilon a \varepsilon a a a t t) = c a a t.$$

Under this definition, any path  $y_1^{T'}$  satisfying  $\mathcal{B}(y_1^{T'}) = w_1^N$  serves as a valid alignment for the label sequence  $w_1^N$ . The probability assigned to a label sequence  $w_1^N$  is obtained by marginalizing over all its possible alignments:

$$\begin{aligned}
p_{\text{CTC}}(w_1^N | h_1^{T'}) &= \sum_{y_1^{T'} : \mathcal{B}(y_1^{T'}) = w_1^N} p(y_1^{T'} | h_1^{T'}) \\
&= \sum_{y_1^{T'} : \mathcal{B}(y_1^{T'}) = w_1^N} \prod_{t=1}^{T'} p_t(y_t | h_1^{T'})
\end{aligned} \tag{57}$$

CTC loss for the input-target pair  $(x_1^T, w_1^N)$  is defined as the negative log-likelihood of the target sequence under the CTC model, i.e., the cross-entropy loss:  $-\log p_{\text{CTC}}(w_1^N | x_1^T)$ .

An illustrative example of the CTC topology is depicted in Figure 16. As shown, the corresponding lattice structure admits two valid initial nodes and two valid final nodes. This arises from the fact that a valid alignment path may begin or end with either a true label or the special blank label, reflecting the inherent flexibility of CTC in handling variable-length alignments.

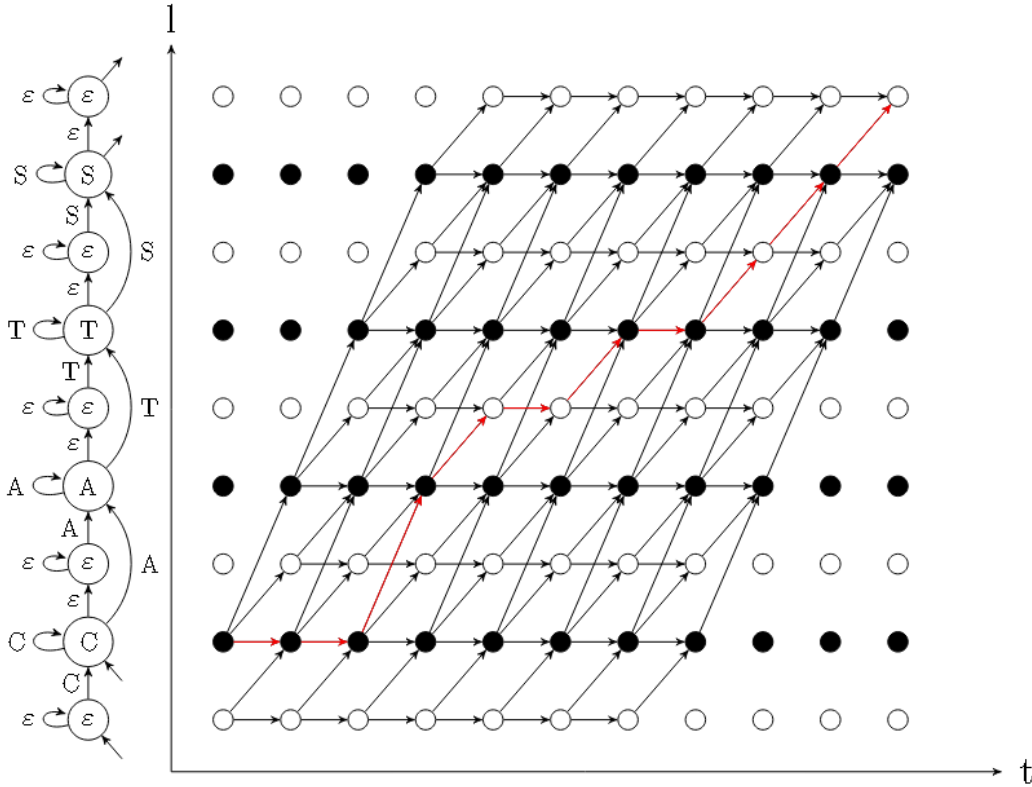


Figure 16: An illustrative CTC lattice, adapted from [27], depicts all valid alignment paths corresponding to the label sequence  $CATS$  along the  $l$ -axis, distributed over 11 time frames on the  $t$ -axis. Each node  $(t, l)$  denotes the emission of label  $l$  at time step  $t$ . Black nodes correspond to the emission of true (non-blank) labels, whereas white nodes indicate the emission of the blank symbol. The highlighted red path exemplifies a possible alignment:  $CCCA\epsilon\epsilon TT\epsilon S\epsilon$ . On the left, a finite state machine is shown, which constrains valid transitions through the lattice, subject to appropriate initial and terminal states.

We highlight two properties of CTC that ensure its consistency with the ASR task:

- The CTC alignment  $y_1^{T'}$ , as previously defined, is *strictly monotonic*.
- The conditional probability  $p_{\text{CTC}}(w_1^N | h_1^{T'})$  defines a distribution over all label sequences with  $N \leq T'$ , aligning with typical ASR scenarios where  $N < T'$ .

Additionally, CTC exhibits an empirically observed “peaky” behavior [27], wherein it predominantly emits the blank symbol with high probability, interspersed with sharp peaks corresponding to predicted labels. This behavior diverges from the intuitive expectation that a label should be strongly emitted throughout its spoken duration. A formal analysis of this phenomenon is provided in [95].

## E.2 CTC Forward-Backward Algorithm

The training objective of a CTC model is to minimize the negative log-likelihood  $-\log p_{\text{CTC}}(w_1^N | h_1^{T'})$ , which necessitates the computation of  $p_{\text{CTC}}(w_1^N | h_1^{T'})$ . A direct evaluation using the definition in Equation 57 is computationally intensive due to the exponential number of possible alignments  $y_1^{T'}$  corresponding to the target sequence  $w_1^N$ . To address this, Graves et al. [27] proposed an efficient dynamic programming (DP) algorithm, analogous to the forward-backward procedure employed in HMMs [71], to compute this quantity.

For a given label sequence  $w_1^N$ , we define the forward variables  $Q_\varepsilon(t, n)$  and  $Q_l(t, n)$  for all  $1 \leq t \leq T'$  and  $0 \leq n \leq N$  as the total probability of all valid alignments of the partial sequence  $w_1^n$  from frame 1 to frame  $t$ , where the alignment ends at frame  $t$  with either a blank symbol ( $\varepsilon$ ) or a non-blank label ( $l$ ), respectively. Formally:

$$Q_\varepsilon(t, n) = \sum_{\substack{y_1^t, \mathcal{B}(y_1^t)=w_1^n \\ y_t=\varepsilon}} \prod_{t'=1}^t p_{t'}(y_{t'} | h_1^{T'}) \quad (58)$$

$$Q_l(t, n) = \sum_{\substack{y_1^t, \mathcal{B}(y_1^t)=w_1^n \\ y_t \neq \varepsilon}} \prod_{t'=1}^t p_{t'}(y_{t'} | h_1^{T'}) \quad (59)$$

here,  $w_1^0$  denotes the empty sequence. The DP procedure is initialized using the following base cases:

$$Q_\varepsilon(t, 0) = \prod_{t'=1}^t p_{t'}(\varepsilon | h_1^{T'}) \quad \forall 1 \leq t \leq T' \quad (60)$$

$$Q_l(t, 0) = 0 \quad \forall 1 \leq t \leq T' \quad (61)$$

$$Q_\varepsilon(1, n) = 0 \quad \forall 1 \leq n \leq N \quad (62)$$

$$Q_l(1, n) = \begin{cases} p_1(w_1 | h_1^{T'}), & \text{if } n = 1 \\ 0, & \text{if } 2 \leq n \leq N \end{cases} \quad (63)$$

For all  $t \geq 2$  and  $n \geq 1$ , the values  $Q_\varepsilon(t, n)$  and  $Q_l(t, n)$  can be computed using the following DP recursion:

$$Q_\varepsilon(t, n) = p_t(\varepsilon | h_1^{T'}) \cdot [Q_\varepsilon(t-1, n) + Q_l(t-1, n)] \quad (64)$$

$$Q_l(t, n) = p_t(w_n | h_1^{T'}) \cdot [Q_l(t-1, n) + Q_\varepsilon(t-1, n-1) + \overline{Q}_l(t-1, n-1)], \quad (65)$$

where  $\overline{Q}_l(t-1, n-1)$  is defined as:

$$\overline{Q}_l(t-1, n-1) = \begin{cases} Q_l(t-1, n-1), & \text{if } w_n \neq w_{n-1} \\ 0, & \text{otherwise} \end{cases} \quad (66)$$

By the definition of the forward variables,  $p_{\text{CTC}}(w_1^N | h_1^{T'})$  could be calculated as follows:

$$p_{\text{CTC}}(w_1^N | h_1^{T'}) = Q_\varepsilon(T', N) + Q_l(T', N) \quad (67)$$

Similarly, the backward variables  $R_\varepsilon(t, n)$  and  $R_l(t, n)$ , defined for all  $1 \leq t \leq T'$ ,  $1 \leq n \leq N+1$ , represent the total alignment probabilities corresponding to the decoding of the label sequence  $w_n^N$

from frame  $t$  to frame  $T'$ , conditioned on the assumption that the label emitted at frame  $t$  is either a blank symbol ( $\varepsilon$ ) or a true label ( $l$ ), respectively.

$$R_\varepsilon(t, n) = \sum_{\substack{y_1^t, \mathcal{B}(y_1^t)=w_n^N \\ y_t=\varepsilon}} \prod_{t'=t}^{T'} p_{t'}(y_{t'}|h_1^{T'}) \quad (68)$$

$$R_l(t, n) = \sum_{\substack{y_1^t, \mathcal{B}(y_1^t)=w_n^N \\ y_t \neq \varepsilon}} \prod_{t'=t}^{T'} p_{t'}(y_{t'}|h_1^{T'}) \quad (69)$$

where  $w_{N+1}^N$  is seen as the empty sequence. The following initializations are needed for the DP:

$$R_\varepsilon(t, N+1) = \prod_{t'=t}^{T'} p_{t'}(\varepsilon|h_1^{T'}) \quad \forall 1 \leq t \leq T' \quad (70)$$

$$R_l(t, N+1) = 0 \quad \forall 1 \leq t \leq T' \quad (71)$$

$$R_\varepsilon(T', n) = 0 \quad \forall 1 \leq n \leq N \quad (72)$$

$$R_l(T', n) = \begin{cases} p_{T'}(w_N|h_1^{T'}), & \text{if } n = N \\ 0, & \text{if } 1 \leq n \leq N-1 \end{cases} \quad (73)$$

For  $t \leq T' - 1, n \leq N$ ,  $R_\varepsilon(t, n)$  and  $R_l(t, n)$  could be later computed by the DP recursion as follows:

$$R_\varepsilon(t, n) = p_t(\varepsilon|h_1^{T'}) \cdot [R_\varepsilon(t+1, n) + R_l(t+1, n)] \quad (74)$$

$$R_l(t, n) = p_t(w_n|h_1^{T'}) \cdot [R_l(t+1, n) + R_\varepsilon(t+1, n+1) + \bar{R}_l(t+1, n+1)], \quad (75)$$

where  $\bar{R}_l(t+1, n+1)$  is defined as:

$$\bar{R}_l(t+1, n+1) = \begin{cases} R_l(t+1, n+1), & \text{if } w_n \neq w_{n+1} \\ 0, & \text{otherwise} \end{cases} \quad (76)$$

By the definition of the backward variables,  $p_{\text{CTC}}(w_1^N|h_1^{T'})$  could be calculated as:

$$p_{\text{CTC}}(w_1^N|h_1^{T'}) = R_\varepsilon(1, 1) + R_l(1, 1) \quad (77)$$

Besides calculating  $p_{\text{CTC}}(w_1^N|h_1^{T'})$ , Graves et al. [27] shows that these forward and backward variables could be employed to compute the error signal of the model during training analytically.

### E.3 CTC Decoding

Decoding (also referred to as *search*, *recognition*, or *inference*) denotes the process of identifying the word sequence  $\hat{w}_1^{\hat{N}}$  that satisfies the decision rule defined in Equation 46, given a sequence of acoustic feature vectors  $x_1^T$ . In this section, we first present general forms of decision rules applicable to seq2seq models. Subsequently, we detail specific decision rules and corresponding search algorithms tailored to the CTC framework.

#### E.3.1 Decision Rules

Because seq2seq models directly calculate  $p(w_1^N|x_1^T)$ , the decision rule for the search is defined as belows:

$$\hat{w}_1^{\hat{N}} = \arg \max_{N, w_1^N} p(w_1^N|x_1^T) = \arg \max_{N, w_1^N} [\log p(w_1^N|x_1^T)]. \quad (78)$$

As for CTC, decision rule 78 could be rewritten as:

$$\hat{w}_1^{\hat{N}} = \arg \max_{N, w_1^N} \left[ \sum_{y_1^{T'}: \mathcal{B}(y_1^{T'})=w_1^N} \prod_{t=1}^{T'} p_t(y_t|h_1^{T'}) \right]. \quad (79)$$



Because of the peaky behavior [95], CTC models usually generate blanks more and true labels less. In order to minimize blanks and increase true label emission, an alternative *prior correction* could be employed [58]. The prior of a label  $y$ , which could then be either a true label or the blank label, is calculated as:

$$p_{\text{prior}}(y) = \frac{1}{\sum_{\tilde{h}_1^{\tilde{T}'}} \tilde{T}'} \sum_{\tilde{h}_1^{\tilde{T}'}} \sum_{t=1}^{\tilde{T}'} p_t(y|\tilde{h}_1^{\tilde{T}'}), \quad (80)$$

where the encoded vectors  $\tilde{h}_1^{\tilde{T}'}$  are retrieved based on the empirical distribution. The label posteriors in decision rule 79 of the search are then divided by their priors as follows:

$$\hat{w}_1^{\hat{N}} = \arg \max_{N, w_1^N} \left[ \sum_{y_1^{T'} : \mathcal{B}(y_1^{T'}) = w_1^N} \prod_{t=1}^{T'} \frac{p_t(y_t|h_1^{T'})}{p_{\text{prior}}^{\lambda_{\text{prior}}}(y_t)} \right], \quad (81)$$

where  $\lambda_{\text{prior}}$  is the prior scale, which is also a constant. In previous empirical studies, prior correction has been used in order to boost CTC performance.

As detailed in Section D above, combining an external LM  $p_{\text{LM}}$  with a seq2seq model generally leads to enhanced results. The most simple and commonly employed method is *shallow fusion* [28], which simply linearly combines decision rule in Equation 78 with the LM score as shown below:

$$\hat{w}_1^{\hat{N}} = \arg \max_{N, w_1^N} [\log p(w_1^N|x_1^T) + \lambda_{\text{LM}} \log p_{\text{LM}}(w_1^N)], \quad (82)$$

where  $\lambda_{\text{LM}}$  is the LM scale. It is important to be aware that there should be no domain mismatch between the LM and the inference domain, and the external LM should preserve the linguistic distribution in the inference domain well. For CTC, shallow fusion could be simply combined with prior correction.

### E.3.2 Greedy Search

Greedy search is a simplified decoding strategy for CTC, in which the summation over all alignment paths in the decision rule in Equation 79 is approximated by the most probable path alone. That is, it selects the single highest-probability alignment path predicted by the CTC model:

$$\begin{aligned} \hat{w}_1^{\hat{N}} &\approx \arg \max_{N, w_1^N} \left[ \max_{y_1^{T'} : \mathcal{B}(y_1^{T'}) = w_1^N} \prod_{t=1}^{T'} p_t(y_t|h_1^{T'}) \right] \\ &= \mathcal{B} \left( \arg \max_{y_1^{T'}} \left[ \prod_{t=1}^{T'} p_t(y_t|h_1^{T'}) \right] \right). \end{aligned} \quad (83)$$

This can be computed in a straightforward manner by selecting the maximum label emission at each time frame, followed by the application of the collapse function to the resulting alignment. Greedy search offers the advantage of high computational efficiency, particularly when employed with Transformer-based architectures, as the encoder computations can be parallelized across time frames. Nevertheless, greedy search typically yields lower accuracy compared to beam search methods [27], and it lacks the capability to incorporate an external LM.

### E.3.3 Time-synchronous Search

CTC is inherently a time-synchronous model, making it natural to adopt a time-synchronous decoding strategy. As proposed in [26], a time-synchronous beam search variant for CTC is introduced, which estimates the sum of *some* path probabilities corresponding to each hypothesis by maintaining two auxiliary partial probabilities during DP. Most studies adopt a variant of beam search in which the score assigned to each hypothesis corresponds to the probability of its most likely alignment path, augmented with an external LM score. This approach bears resemblance to Viterbi decoding employed in HMM-based systems [71]. Incorporating prior correction, the resulting decision rule for this decoding procedure can be expressed as:

$$\hat{w}_1^{\hat{N}} = \arg \max_{N, w_1^N} \left[ \log \left( \max_{y_1^{T'} : \mathcal{B}(y_1^{T'}) = w_1^N} \prod_{t=1}^{T'} \frac{p_t(y_t | h_1^{T'})}{p_{\text{prior}}^{\lambda_{\text{prior}}}(y_t)} \right) + \lambda_{\text{LM}} \log p_{\text{LM}}(w_1^N) \right]. \quad (84)$$

Details of the search procedure are provided in Algorithm 1. The initial hypothesis consists of the empty sequence augmented with a virtual begin-of-sequence (BOS) label. It is important to note that, in this context, the set of partial hypotheses corresponds not to partial label sequences, but to partial CTC alignments.

At each iteration over the input frames, the scores of all possible hypothesis expansions are computed. Subsequently, if multiple expansions yield the same label sequence, only the one with the highest score is retained. This operation constitutes a form of maximum recombination over all expansions that correspond to the same label sequence. Following this recombination, a subset of the expansions is selected based on the beam size constraint, thereby pruning the search space.

In the absence of an external LM and prior correction, this procedure reduces to the standard greedy search.

---

**Algorithm 1** CTC time-synchronous beam search.

---

```

1: Input: beam size  $H$ , CTC posterior  $p_t(v|h_1^{T'}) \forall v \in V', 1 \leq t \leq T'$ , language model  $p_{\text{LM}}$ , LM
   scale  $\lambda_{\text{LM}}$ , label prior  $p_{\text{prior}}(v) \forall v \in V'$ , prior scale  $\lambda_{\text{prior}}$ 
2: Initialize:
3:  $hyps \leftarrow \{ \langle \text{BOS} \rangle \}$  ▷ set of partial hypotheses
4:  $scores \leftarrow \{ \}$  ▷ hash-map storing scores of partial hypotheses
5:  $scores[\langle \text{BOS} \rangle] \leftarrow 0$ 
6: for  $t = 1 \dots T'$  do
7: /* Consider all possible hypothesis expansions */
8:    $expansions \leftarrow \{ g \cdot v | g \in hyps, v \in V' \}$ 
9:   for  $g \in hyps$  do
10:    for  $v \in V'$  do
11:       $scores[g \cdot v] \leftarrow scores[g] + \log p_t(v|h_1^{T'}) - \lambda_{\text{prior}} \log p_{\text{prior}}(v)$ 
12:      if  $v \neq \varepsilon$  and  $v \neq \text{last label of } g$  then
13:         $scores[g \cdot v] \leftarrow scores[g \cdot v] + \lambda_{\text{LM}} \log p_{\text{LM}}(v|\mathcal{B}(g))$ 
14:      end if
15:    end for
16:  end for
17: /* Recombination step */
18:  for  $g_1, g_2 \in expansions \times expansions$  do
19:    if  $g_1 \neq g_2$  and  $\mathcal{B}(g_2) = \mathcal{B}(g_1)$  then
20:      if  $scores[g_2] > scores[g_1]$  then
21:         $scores[g_1] \leftarrow -\infty$ 
22:      else
23:         $scores[g_2] \leftarrow -\infty$ 
24:      end if
25:    end if
26:  end for
27: /* Pruning step */
28:   $hyps \leftarrow \text{the } H \text{ partial hypotheses in } expansions \text{ with highest } scores$ 
29: end for
30: Return:  $\mathcal{B} \left( \arg \max_{y_1^{T'} \in hyps} scores[y_1^{T'}] \right)$ 

```

---

### E.3.4 Label-synchronous Search

Label-synchronous search was originally introduced alongside the CTC topology in [27]. This approach directly applies the decision rule given in Equation 78, wherein beam search operates at the level of label sequences. To facilitate this procedure, the *CTC prefix probability* is required, denoted

by

$$p_{\text{CTC}}(w_1^n, * \mid x_1^T) := p_{\text{CTC}}(w_1^n, * \mid h_1^{T'}) \quad (85)$$

which quantifies the probability that the CTC model emits a sequence beginning with the prefix  $w_1^n$ , followed by an arbitrary suffix (including the empty suffix).

$$p_{\text{CTC}}(w_1^n, * \mid h_1^{T'}) = \sum_{\nu \in (V')^*} p_{\text{CTC}}(w_1^n \cdot \nu \mid h_1^{T'}), \quad (86)$$

where  $(V')^*$  denotes the set of all label sequences over  $V'$ , including the empty sequence. The score assigned to a partial hypothesis  $w_1^n$  is defined as  $\log p_{\text{CTC}}(w_1^n, * \mid h_1^{T'})$ . If the hypothesis  $w_1^n$  is complete, its final score becomes  $\log p_{\text{CTC}}(w_1^n \mid h_1^{T'})$ . The label-synchronous beam search procedure, along with the DP formulation for computing the CTC prefix probability during decoding, is detailed in [25].

The label-synchronous search procedure operates at the output label level, facilitating straightforward integration of an external LM. The partial score of a hypothesis  $w_1^n$  is given by:

$$\log p_{\text{CTC}}(w_1^n, * \mid h_1^{T'}) + \lambda_{\text{LM}} \log p_{\text{LM}}(w_1^n). \quad (87)$$

This approach sums over all path probabilities corresponding to a partial label sequence, but incurs a time complexity of  $\mathcal{O}(T')$  per hypothesis extension. Consequently, the total complexity becomes  $\mathcal{O}(N_{\text{max}}T')$ , where  $N_{\text{max}}$  denotes the length of the longest hypothesis. This is substantially less efficient than the  $\mathcal{O}(T')$  complexity of time-synchronous search. Moreover, CTC is inherently time-synchronous.