A. Appendix

A.1. Training Details

We implement our approach LASER in three different state-of-the-art ASD frameworks, LoCoNet [29], TalkNet [26], and Light-ASD [16]. We closely follow the training details for fair comparison. Specifically, for LoCoNet-based training, we use batch size of 4 and sample 200 frames per training example. Each model is trained with 25 epochs on 4 RTX 2080 GPUs. For Light-ASD and TalkNet, we use a batch size that contains at most 2000 frames and each model is trained on one A40 GPU. We use Adam [6] as our optimizer with learning rate of 5×10^{-5} that reduces by 0.5% per epoch. For LoCoNet and TalkNet, we set $\lambda_{av} = 1$, $\lambda_a = 0.4$, and $\lambda_v = 0.4$; and for Light-ASD, we set $\lambda_{av} = 1$ and $\lambda_v = 0.5$, following the original work. Random resizing, cropping, horizontal flipping, and rotations are used as visual data augmentation operations and a randomly selected audio signal from the training set is added as background noise to the target audio [26].

A.2. Implementation of Landmark Pooling and 2D Lip Encoding methods

In this section, we introduce implementation details of the Landmark Pooling method [28] and Landmark as Discrete Inputs [9]. Since these methods were originally proposed to work with simple CNN architectures, we adapt them to the state-of-the-art LoCoNet framework for fair comparison.

A.2.1. Landmark Pooling

Given a face track and the lips landmark detection results, we first obtain feature map $\mathcal{F}_v \in \mathbb{R}^{T \times C \times H \times W}$ after the first layer the ResNet, where T denotes the number of frames, H and W are the spatial dimension of f_v, and C is the number of channels, we follow [28] to pool the feature with normalized lip landmark coordinates $\{(x_i, y_i)\}_{i=1}^{N}$ by $\mathcal{F}_v[x_i, y_i] \in \mathbb{R}^{T \times C}$. Then, we concatenate all N landmarks into $\mathcal{F}_{ld} \in \mathbb{R}^{T \times NC}$ and feed $\mathcal{F}_{ld}$ through a fully connected layer to get $\mathcal{F}'_{ld} \in \mathbb{R}^{T \times D}$ where D is the hidden dimension after the average pooling layer of ResNet. Finally, $\mathcal{F}'_{ld}$ is concatenated to $\mathcal{F}'_v$, the semantic feature output by ResNet and reduce the dimension with another fully-connected layer to obtain the final frame-level representation $\mathcal{F} \in \mathbb{R}^{T \times D}$. $\mathcal{F}$ is further processed by V-TCN and Context Modeling models along with audio feature to make prediction. The visualization of the method is presented in Figure A.

A.2.2. Landmark as Discrete Inputs

Given a face track and the lips landmark detection results $f \in \mathbb{R}^{T \times N \times 2}$ where T is the number of frames, N is the number of landmarks, and 2 is the coordinate (x, y). We adopt the architecture in [9] to process f with 3x3 convolution layers and convert the raw landmark coordinates into embeddings. The resulting embedding is flattened and concatednated to the visual feature in the same way of Landmark Pooling. The visualization of the architecture is presented in Figure B.

A.3. Parameter Effiency of LASER

As illustrated in Figure C, our proposed LASER approach not only achieves better performance but also only requires the negligible additional parameters compared to Landmark Pooling and Landmark as Discrete Inputs. This indicates that LASER's design is significantly more parameter-efficient, which translates to reduced memory usage and faster training/inference. By minimizing the number of additional parameters without compromising accuracy, LASER offers a compelling balance between computational cost and performance gains.

A.4. Hyperparameters Selection

We present our ablation study on hyperparamete selection, including the ablation on consistency loss weight and the Lip Track Encoding injection layer. The hyperparameter selection process is conducted through a holdout evaluation protocol (different from all protocols in main evaluation) by reversing the audio in original video clip to establish a non-speaking scenario. This avoids the risk of tuning on the test set and ensures generalizability.

A.4.1. ResNet vs 3D CNN

As discussed in Section 3.2, we integrate the aggregated lip track encoding (LTE) into the visual features produced by the 3D-CNN. We compare this approach with integrating LTE directly with the RGB face frames before the 3D-CNN. As shown in Table B, integrating LTE before the 3D-CNN leads to a significant performance drop. We speculate that this is due to the limited number of channels in the RGB image, where integrating LTE with additional channels may interfere with the feature extraction process of the original face frame. Thus, we opt to integrate LTE into the visual features output by the 3D-CNN.

A.4.2. Integrating Lip Track Encoding to Visual Features

Stage	# Channels S				
	1	2	4	8	16
1	84.9	84.3	**86.9**	85.6	84.7
2	84.3	84.7	86.2	83.6	85.0
3	85.5	84.2	83.8	86.1	82.9
4	85.3	86.5	85.1	84.9	82.4

Table A. **Ablation study on lip track integration with different stages of ResNet.** Setting $S = 4$ and integrating lip track encoding before the first stage of ResNet results in the best performance.

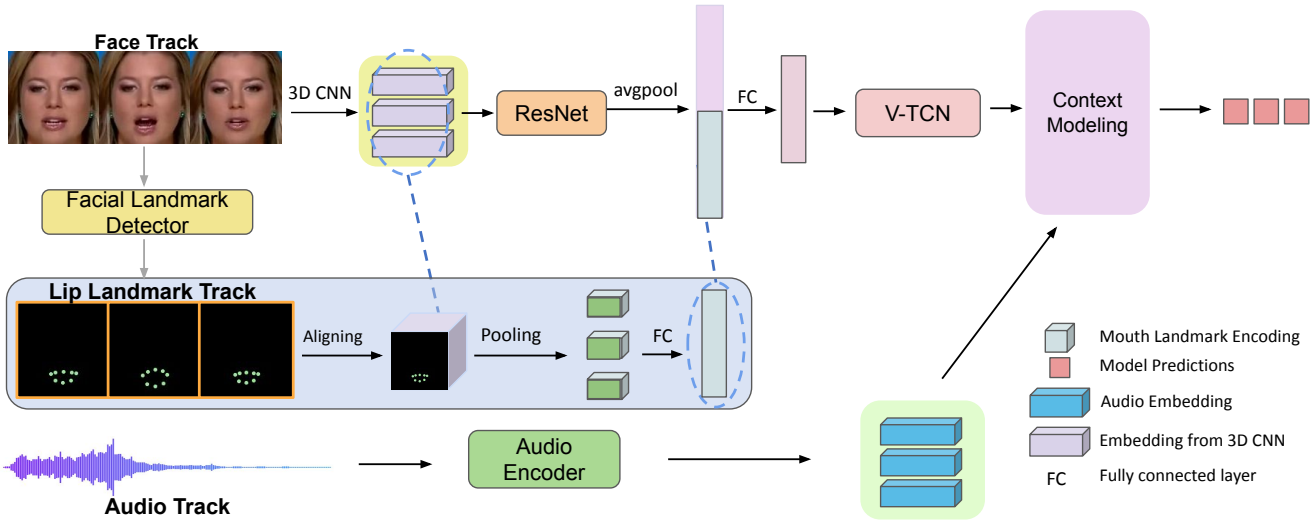

Figure A. **Our implementation of Landmark Pooling [28].**

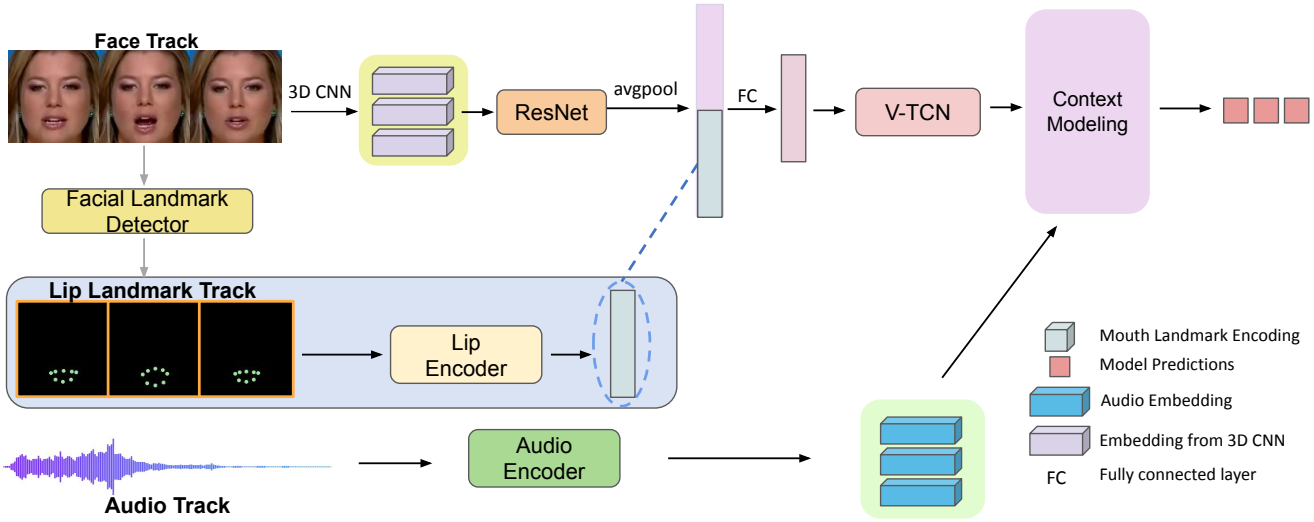

Figure B. **Our implementation of 2D Lip Encoding method [9].**

Layer	Regular Audio	Noise Audio
3D CNN	92.6	86.6
ResNet	**95.1**	**86.7**

Table B. **Ablation study on lip track encoding integration**. Integrating lip track encoding before 3D-CNN layer (i.e, on RGB frames) results in suboptimal performance.

We further investigate integrating lip landmark encoding (LTE) at different stages within ResNet, which has four internal stages with varying downsampling strides. Our ablation study (Table A) shows that integrating LTE before the first stage—using features from the 3D-CNN output—yields the best performance under noisy audio conditions, while later-stage integration degrades performance. Additionally, we examine the sensitivity of the aggregation factor S when condensing sparse lip track encodings from $K = 82$ to S dense feature maps. We find that $S = 4$ achieves the best balance, avoiding both excessive compression and sparsity.

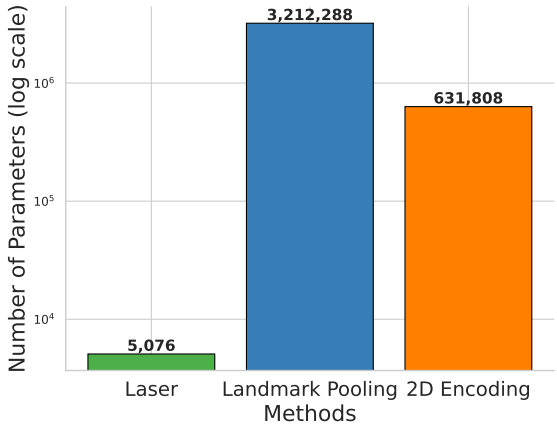

Figure C. **Number of additional parameters of each method**. The y-axis is in log scale, and the number on each bar represents the true additional number of parameters. The x-axis contains 3 methods: LASER, Landmark Pooling [28], and Landmark as Discrete Inputs [9].

Method	λ_c				
	0.2	0.4	0.6	0.8	1.0
Acccuracy	85.7	85.9	86.0	85.9	**87.5**

Table C. **Ablation study on consistency loss weight.** Setting $\lambda_c = 1$ achieves the best results while decreasing λ_c consistently degrades the performance.

A.4.3. Weight of Consistency Loss

We also study the sensitivity of the weight of consistency loss λ_c where we range the value from 0.2 to 1.0. As shown in Table C, $\lambda_c = 1$ achieves the best performance whereas decreasing the loss weight consistently hurts the model performance. Thus, we set $\lambda_c = 1$ in all of our experiments.