System Description for T290 Team System

Yang Hu and Haowei Li

CZUR

{huyang, lihaowei}@czur.com

Abstract

This report describes our submission to the fixed track of task 1 in the first Chinese Continuous Visual Speech Recognition Challenge (CNVSRC 2023). Our system exploits audio data in the training stage to improve the accuracy of visual speech recognition (VSR). Specifically, we firstly train an audio speech recognition (ASR) network. Then, we train a video to audio converter (VAC) which aims to convert video data to the latent features of the ASR network. Finally, we combine VAC and ASR networks and finetune both networks jointly on video data only. The resulted network only needs video data for inference. We also explore three ensemble methods to further enhance the performance. Our best system achieves an Character Error Rate (CER) of 39.4707% on the official *CNVSRC-Single.Eval* split.

1. Data

We use *CN-CVS* and *CNVSRC-Single.Dev* in our system development. We use two split strategies. The first strategy follows the official split of the two datasets. The second strategy uses the training/validation/test splits of *CN-CVS* and the training split of *CNVSRC-Single.Dev* as the training data, and it uses the validation split of *CNVSRC-Single.Dev* as the validation data. We choose one of the two splits for different training stages of our system. Please see section 2.4 for more details.

2. Models

The intuition of our system is to utilize audio data to train an ASR system as the core for VSR. We expect this leads to better performance compared to training on video data only, since audio data generally achieves far better speech recognition accuracy [1, 2] due to its less ambiguity. We design our system based on the method proposed in [3]. It consists of an ASR system and a video to audio converter (VAC). The ASR system performs speech recognition using audio, and VAC converts video data to latent features of the ASR system. By connecting VAC to the ASR system, we can use video data only for inference. Fig. 1 shows an illustration of our system.

2.1. Pre-processing

For video data, we use the code of the CNVSRC 2023 baseline [4] for pre-processing. Briefly speaking, it detects facial landmarks for each frame, aligns the facial images and crops the lip region. For audio data, we use raw audio waveforms without further processing.

2.2. ASR System

Our ASR system adopts the same end-to-end network structure as the CNVSRC 2023 baseline [4]. The whole model includes an audio frontend, an transformer encoder and a transformer decoder. The audio frontend extracts features frame by frame from raw audio waveforms. The transformer encoder exploits the global relations between audio features at different frames. The transformer decoder predicts the probability of the next token given the output features of the encoder and the predicted tokens of previous frames. For detailed network structure we use the audio backbone defined in the CNVSRC 2023 baseline config files `train_cncvs_4s.yaml`[1] and `train_cncvs_4s_30s.yaml`[2].

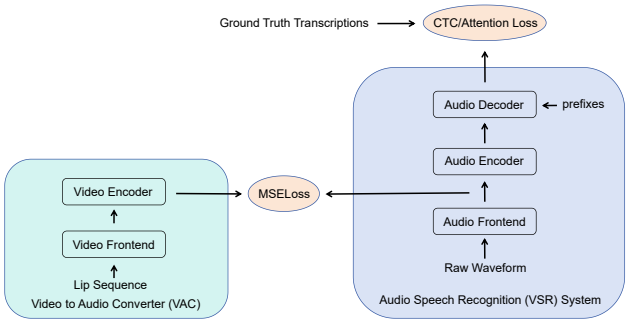

Figure 1: The CZUR system for CNVSRC 2023.

2.3. Video to Audio Converter (VAC)

Since ASR generally achieves better performance than VSR, we aim to utilize the well-trained ASR system for VSR. To achieve this, we use a video to audio converter (VAC) to transfer video frames to latent ASR features, so that we can feed the transferred features through the ASR system to get token predictions. Our VAC is designed based on the method proposed in [3]. It employs the video frontend and the VSR encoder in the CNVSRC 2023 baseline [4] for feature conversion. The video frontend extracts features from video frames. The extracted features are fed to the VSR encoder which aims to transfer videos features to the features output by the audio frontend of the ASR system, as shown in Fig. 1.

Moreover, we note that [3] suggests video features representing lip movements can be better transferred to latent audio features, in comparison to the video features aligned with the text decoding task. Therefore, we initialize the video frontend and the first four encoder layers of VAC with the pre-trained

[1] https://github.com/MKT-Dataoceanai/CNVSRC2023Baseline/blob/master/conf/train_cncvs_4s.yaml

[2] https://github.com/MKT-Dataoceanai/CNVSRC2023Baseline/blob/master/conf/train_cncvs_4s_30s.yaml

weights of the CNVSRC 2023 baseline on the *CN-CVS* dataset[3]. We freeze these layers and train the rest encoder layers with random initialization. We expect that the shallow layers we freeze provide features more related to lip movements than the text decoding task. At the same time this strategy avoids training from scratch which could take longer to converge.

Last but not least, we use the masking scheme in [3] to facilitate the transfer of video features to audio features. Specifically, audio features (*i.e.* output features of the audio frontend in the VSR system) are masked and added to the output features of the frozen VAC layers. Then the summed features are fed through the rest VAC encoder layers to predict the audio features before the masking. The initial masking ratio is small (0.3 in our setting). The ratio increases along with training epochs and eventually reaches 1.0 at the end of the training. Such a scheme eases the initial stage of the training by providing the network with more information on audio features. And it gradually removes the audio information as the training proceeds, pushing the network to smoothly learn to convert video features to audio features.

2.4. Model training

We design a multi-stage training scheme for our model. The ASR system is trained on audio data at first. Then, we train VAC with both audio and video data. Finally, trained VAC and ASR system are connected and finetuned on video data only.

Training of ASR system. Following the curriculum learning scheme in the CNVSRC 2023 baseline [4], we firstly train the ASR system on utterances less than 4 seconds on *CN-CVS*, using the split strategy 1 mentioned in Section 1. Then, we train the ASR system on utterances between 4 seconds and 30 seconds, using the split strategy 2 in Section 1 on combined *CN-CVS* and *CNVSRC-Single.Dev* datasets. We adopt the CTC/Attention loss used in the CNVSRC 2023 baseline for ASR system training.

Training of VAC. We adopts the above curriculum learning scheme to train VAC as well. VAC is trained on utterances less than 4 seconds on *CN-CVS* with the split strategy 1, before it is trained on utterances between 4 seconds and 30 seconds on combined *CN-CVS* and *CNVSRC-Single.Dev* datasets with the split strategy 2. For VAC training, we use mean square error between the predicted features (*i.e.* output of VAC) and the target features (*i.e.* output of the audio frontend of the ASR system) as the loss. Recall that some VAC layers are frozen as mentioned in Section 2.3.

Joint finetuning of ASR and VAC. After the ASR system and VAC are separated trained, we simply connect the two systems by feeding the VAC output through the ASR system to predict tokens. At this stage, we firstly finetune VAC with the ASR system frozen for a few epochs. Then, we unfreeze all layers and train the whole network. All training at this stage is performed on combined *CN-CVS* and *CNVSRC-Single.Dev* datasets with the split strategy 2.

Additionally, instead of selecting checkpoints based on validation metrics, we track the inference CERs of the models on a randomly selected 200-utterance subset of the validation split of *CNVSRC-Single.Dev*. We select the epochs with top performance for model ensemble and submission (see below). This is because we have found mismatch between the training/validation metrics and actual inference CERs in the joint finetuning stage. Specifically, although the validation decoding accuracy kept rising throughout the training, the inference CER actually increases rather than decreases after a certain number of epochs. We think this may due to the discrepancy between the training and inference process. The decoder learns to predict next-token probabilities with ground truth tokens as prefixes during training, but it has to predict the next token based on its own previous predictions in inference. For the VSR task, the CER is high, so the prefixes predicted by the model in inference may be very different from the ground truth. As a result, the inference data distribution deviates from the training data distribution, and it may lead to performance mismatch between training/validation and inference.

[3] model_avg_last10_cncvs_4s_30s.pth in [4]

2.5. Model Ensemble

We exploit three model ensemble methods: intra-model fusion, inter-model weight fusion and decoding score fusion. In terms of intra-model fusion, we average the weights of different checkpoints produced during the training of a model. For inter-model weight fusion, we attempt to average the weights of models trained with different settings. As for decoding score fusion, we use mean fusion of CTC and attention probabilities produced by different models to calculate next-token probabilities in the decoding stage. We train several models with different hyper-parameter settings. For each model, we obtain a set of candidates using intra-model fusion. After that, we perform inter-model weight fusion with candidates from different models. Finally, models with top validation performance in the intra-model fusion and inter-model weight fusion stages are selected for decoding score fusion.

2.6. Analysis

At the first glance, our system seems to be a CNVSRC 2023 baseline with doubled encoder depth. However, the whole training process makes our system more than a deeper baseline. In this subsection, we analyze the differences between our system and the baseline, together with the possible advantages brought by our training process.

First, the baseline trains the whole network using video data only. With limited data, a deeper baseline is prone to overfitting. In contrast, we train different parts of our network with different data, before we finetune on video data only. We suppose this helps to better avoid overfitting when training a deeper network.

Second, we think the data variability in our training process might also lead to better performance. Specifically, we can view the separate training of the ASR system and VAC as pre-training before the final joint finetuning. Since different data modalities and targets are used for the two pre-training tasks, our model might better exploit the underlying data distribution in training. So it might attain better performance.

Third, we set more specified targets for our network. The baseline simply takes video frames as input and let the network output token probabilities. It is well-known that this is hard task. Actually, viewing the whole task as a blackbox may increase the difficulty of optimization, and it could lead to suboptimal solutions. Differently, we explicitly specify two different targets for the two parts of our system, respectively. The deeper part of our system (*i.e.* the ASR system) is initially trained to perform speech recognition using audio signal, while the shallower part of our system (*i.e.* VAC) transfers video features to audio features. The former task is well-studied and promising performance can be guaranteed [5, 6, 7], while the latter task is also explored in recent research and good performance is reported [3]. Such manually-designed tasks provides

the network with more specified and more achievable optimization targets, and hence it might produce better models.

Table 1: CER of our submissions on *CNVSRC-Single.Eval*. Intra: intra-model suion; Inter: inter-model weight fusion; score: decoding score fusion.

Submission ID	Ensemble			CER
	Intra	Inter	Score	
1	✓			41.3193%
2	✓		✓	39.9156%
3		✓		40.3947%
4	✓		✓	39.4707%
5	✓	✓	✓	39.7367%

3. Results

We have made five submissions to the submission system. Tab. 1 shows CERs of our submissions on the official *CNVSRC-Single.Eval* evaluation set. Note that we list ensemble choices to provide brief information on each submission. The ensemble choice in the table does not form a controlled ablation study, as the models, checkpoints and the number of models used for ensemble are different in each submission.

4. Resource

We train our models on single GPUs with gradient accumulation and mixed-precision training. We use two GeForce RTX 3090s and one A100 to train our system.

5. References

[1] Pingchuan Ma, Stavros Petridis, and Maja Pantic, "End-to-end audio-visual speech recognition with conformers," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 1–5.

[2] Pingchuan Ma, Alexandros Haliassos, Adriana Fernandez-Lopez, and Honglie Chen, "Auto-AVSR: Audio-visual speech recognition with automatic labels," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[3] Yasser Abdelaziz Dahou Djilali, Sanath Narayan, Haithem Boussaid, Ebtessam Almazrouei, and Merouane Debbah, "Lip2Vec: Efficient and robust visual speech recognition via latent-to-latent visual to audio representation mapping," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 2023, pp. 13790–13801.

[4] `https://github.com/MKT-Dataoceanai/CNVSRC2023Baseline`.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. 2017, vol. 30, Curran Associates, Inc.

[6] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang, "Conformer: Convolution-augmented Transformer for Speech Recognition," in *Proceedings of Interspeech 2020*, 2020, pp. 5036–5040.

[7] Sehoon Kim, Amir Gholami, Albert Shaw, Nicholas Lee, Karttikeya Mangalam, Jitendra Malik, Michael W Mahoney, and Kurt Keutzer, "Squeezeformer: An efficient transformer for automatic speech recognition," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds. 2022, vol. 35, pp. 9361–9373, Curran Associates, Inc.