Appendix A: ABLATION IN RESULTS

To discuss more about baseline as a Model-I, we also conducted the following experiments, we substituted a baseline for Model-I and evaluated its performance. By comparing coarse and fine predictions, visualizing latent features, and comparing with ground truth (Fig. A.1), we assessed the two-stage model's effectiveness. The subgraph visualizes T-SNE-processed samples, with b and d showing single-stage feature distributions. SVC classification highlights feature-sample correlations, revealing single-stage classification capabilities.

The hierarchical results of the Fig A.1 subgraph (b) show that the features of ResNet after feature extraction are too complex, resulting in the failure of SVC, which also means that ResNet relies on a more powerful classifier in the two-stage model. The comparison between the hierarchical graph and the linear fitting graph reveals that the two-stage structure can further enhance the prediction capabilities of the single-stage model.

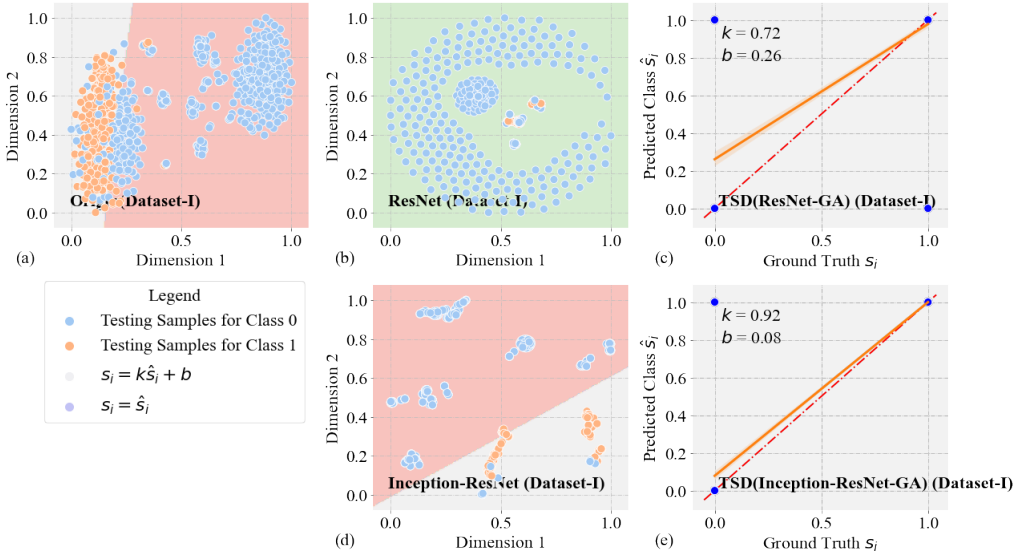

Fig. A.1: T-SNE feature distribution plots (a, b, d) and correlation comparison plots (c, e) contrasting different model and GA model on Dataset-I.

In Figure A.2, the degree of aggregation in subgraph d is more intensive, indicating that the EfficientNet model can classify FRB samples well. through a comparison of subfigures c and e, we observe that TSC(DenseNet-GA) and TSC(EfficientNet-GA) exhibit similar precision levels. However, as seen in subfigures b and d, the DenseNet model performs comparatively poorer in coarse classification capability compared to EfficientNet. It is noteworthy that, through the introduction of the two-stage cascade structure, the prediction precision of the DenseNet model can be significantly improved to match the level of EfficientNet. This result demonstrates that the proposed two-stage cascade structure possesses remarkable robustness and enhancement capabilities for single-stage models with weaker performance.

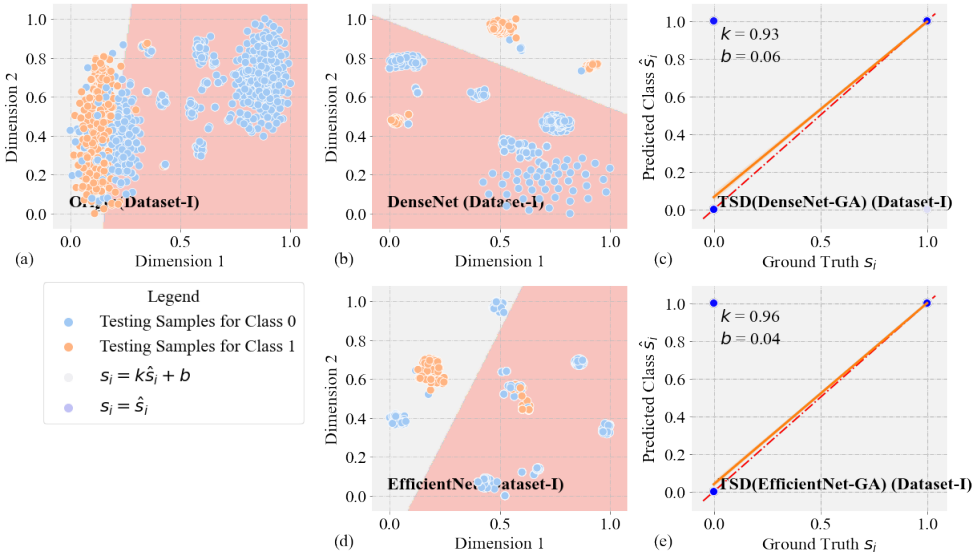

Fig. A.2: T-SNE feature distribution plots (a, b, d) and correlation comparison plots (c, e) contrasting different model %and GA model on Dataset-I.

Appendix B: TRANSFER LEARNING RESULTS

B.1. Transfer Learning

We also perform model transfer training on Dataset-III. Specifically, we randomly select half of the positive and negative samples from Dataset-III to form the training set, and use the other half as the test set. In the transfer training step, the model initializes its parameters with the optimal values obtained from training on Dataset-II. These models are trained for 4 epochs, and during this transfer training, the learning rate and other hyperparameters are kept at the original values obtained during training on Dataset-II.

B.2. Transfer Datasets

In this case, a small amount of real signal data was tagged, which was obtained from the FRB signal data volume of the FRB radio telescope, available at https:fast.cstcloud.com(**?**). A total of 56 FITS files were used, resulting in a total of 60 signal files and 23 effective FRB signal files. A total of 2,661 images of size 4096×4096 were annotated. There were 74 positive and 2,587 negative samples. A detailed table of the FAST signals is given in Table C.6.

B.3. preprocessing

Due to the presence of Gaussian noise and radio frequency interference (RFI) signals throughout the background, Model are easily affected by RFI signals. In the process of detection. DEVANSH.A(**?**) suggests that in order to generalize the model to accommodate different types of astronomical telescopes, the data can be processed with channel baseline averaging to achieve noise reduction. Zhang C proposed to use Gaussian filtering for smoothing, which can improve feature saliency and achieve noise reduction effect.in this section, we compare FETCH(Boxcar filiter), Gaussian filiter, Binarization, OTSU(Maximization of interclass variance algorithm enhanced algorithm(**?**)) four different enhancement algorithms and present our noise

reduction processing algorithm. The four algorithms are expressed as:

$$f_{fetch}(x) = x_i - E(x_i) \tag{B.1}$$

$$G_{gaussion}(x) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{B.2}$$

$$f_{binnary}(x) = \begin{cases} y_{i,j} = 1,\ y_{i,j} \geq \sigma^2 \\ y_{i,j} = 0,\ y_{i,j} < \sigma^2 \end{cases} \tag{B.3}$$

$$f\left(y,i,j\right)_{otsu} = \begin{cases} y_{i,j} = \sigma^2,\ y_{i,j} > \sigma^2, max\ \ \sigma^2 = \frac{(mG \times p_1 - m)^2}{p_1(1-p_1)} \\ y_{i,j} = \ y_{i,j},\ otherwise \end{cases} \tag{B.4}$$

Where i indicates the signal channel range, in our experiment, the range was [1,4096]. The subscript j represents j in unit sampling time, E represents its mathematical expectation, x_i represents the data sampled under the total sampling time on the i channel, p_1 represents the probability of the part greater than the threshold, m represents the cumulative mean of the gray level under the threshold, and mG represents the global mean of the image.

To visually demonstrate the differences in the enhancement of the algorithms, We add noise to the background of the simulated FRB signal using random Gaussian noise and salt-and-pepper noise to evaluate the noise reduction effect of different noise reduction methods which show in the following table:

Table B.1: MSE(error of mean square) of the four enhancement algorithms, smaller means closer to a noise-free image

Size = 100	FETCH	Gaussian	Binary	OTSU
MSE	0.07107	0.0691	0.0572	**0.0422**

We selected batch data of size $100 \times 4096 \times 4096$ for comparative analysis, tested on Intel(R) Core(TM) i7-10700 CPU @ 2.90GHz, and finally OTSU has less time cost in processing batch data than other enhanced algorithms which show in the following table:

Table B.2: Time overhead of the four enhancement algorithms

Size = 100	FETCH	Gaussian	Binary	OTSU
Time(second)	30.6266	12.3432	13.3691	**3.5693**

The comparative experiments in the table demonstrate that the OTSU method outperforms other enhancement algorithms in terms of both noise reduction performance and time efficiency.

B.4. Test Result On Dataset-III

We choose half the number of typical positive and negative sample segments for training on the real data labeled in **??**, The proportion of the training set and test set is 50% and 50% respectively.Just train 4 epoch

on the train set. The optimal parameters for training on the Dataset-I were chosen, and the models trained with transfer were tested on the test set.The test result is shown in Table B.3.

Table B.3: Dataset-III test results after transfer learning

Model Name	Accuracy	Precision	Recall	F1
TSC(EfficientNet-GA)	**0.991**	**0.946**	**0.920**	**0.924**
EfficientNet	0.940	0.932	0.865	0.881
ResNet	0.664	0.972	0.653	0.781
Inception-ResNet	0.830	0.711	0.964	0.818
DenseNet	0.900	0.847	0.980	0.908
conv17(**?**)	0.812	0.794	0.701	0.744
FETCH(**?**)	0.903	0.893	0.924	0.908

Appendix C: MODEL STRUCTURE

The model structure of our experiment is shown below:

Table C.1: GA model structure

Layers	Output Size	Kernel Size	number
Flatten	1×100		1
Linear1+BN	1×64		1
Linear2+BN	1×32		1
Linear3+BN	1×2		1
MLP	1×100		1
MLP	1×32		1
Global Attention	1×2		1

Table C.2: ResNet-conv17 Model Structure

Layers	Output Size	Kernel Size	Number	
Conv0	$512 \times 512 \times 1$	32×1 Conv	1	–
Conv1	$128 \times 128 \times 32$	7×7 Conv	1	–
Conv2	$32 \times 32 \times 32$	3×3 Conv	2	–
Conv3	$8 \times 8 \times 64$	3×3 Conv	3	–
Conv4	$4 \times 4 \times 128$	3×3 Conv	2	–
Avg-pool	$1 \times 1 \times 128$	–	–	–
fc	1×2	–	–	–

Table C.3: Inception-ResNet Model Structure

Layers	Output Size	Kernel Size	Number
Stem	$127 \times 127 \times 32$	3×3 Conv	1
	$125 \times 125 \times 32$	3×3 Conv	
	$125 \times 125 \times 32$	3×3 Conv	
Inception-ResNet-A	$62 \times 62 \times 192$	1×1 Conv	5
		$\begin{bmatrix} 1 \times 1 \\ 3 \times 3 \end{bmatrix}$ Conv	
		$\begin{bmatrix} 1 \times 1 \\ 3 \times 3 \\ 3 \times 3 \end{bmatrix}$ Conv	
Reduction-A	$62 \times 62 \times 192$	3×3 MaxPooling	3
		3×3 Conv	
		$\begin{bmatrix} 1 \times 1 \\ 3 \times 3 \\ 3 \times 3 \end{bmatrix}$ Conv	
Inception-ResNet-B	$29 \times 29 \times 320$	1×1 Conv	10
		$\begin{bmatrix} 1 \times 1 \\ 1 \times 7 \\ 7 \times 1 \end{bmatrix}$ Conv	
Reduction-B	$14 \times 14 \times 1088$	3×3 MaxPooling	1
		$\begin{bmatrix} 1 \times 1 \\ 3 \times 3 \end{bmatrix}$ Conv	
		$\begin{bmatrix} 1 \times 1 \\ 1 \times 7 \\ 7 \times 1 \\ 3 \times 3 \end{bmatrix}$ Conv	
Inception-ResNet-C	$6 \times 6 \times 2080$	1×1 Conv	1
		$\begin{bmatrix} 1 \times 1 \\ 1 \times 3 \\ 3 \times 1 \end{bmatrix}$ Conv	
Avg Pool	$1 \times 1 \times 1536$		1
Softmax	1×2		1

Table C.4: Dense121 Model

Layers	Output Size	Kernel Size	number
Convolution	$128 \times 128 \times 64$	–	1
Max Pooling	$64 \times 64 \times 64$	–	1
Dense Block	$64 \times 64 \times 512$	$\begin{bmatrix} 1 \times 1 \\ 3 \times 3 \end{bmatrix}$ Conv	6
Transition Layer	$32 \times 32 \times 512$	1×1	–
Dense Block	$32 \times 32 \times 512$	$\begin{bmatrix} 1 \times 1 \\ 3 \times 3 \end{bmatrix}$ Conv	12
Transition Layer	$16 \times 16 \times 1024$	1×1	–

Table C.5: Efficient-Net model Structure

Layers	Output Size	Kernel Size	number
Zero Padding	$257 \times 257 \times 1$		1
Zero Padding	$128 \times 128 \times 32$	3×3	1
Zero Padding	$128 \times 128 \times 16$	3×3	1
MBConv6	$128 \times 128 \times 24$	3×3	1
MBConv6	$64 \times 64 \times 40$	5×5	2
MBConv6	$32 \times 32 \times 80$	3×3	2
MBConv6	$16 \times 16 \times 112$	5×5	3
MBConv6	$16 \times 16 \times 192$	5×5	3
MBConv6	$7 \times 7 \times 320$	3×3	4
Conv	$7 \times 7 \times 1280$	3×3	1
Pooling	$1 \times 1 \times 2304$		1
Dropout	1×2304		1
FC			

Table C.6: Information about the Fast number used to migrate the dataset(please see https://fast.cstcloud.cn/datavolume/detail For more detail)

PulseID	Obs_Date(UT)	FileName	Sample_ID
P0003	20190830	FRB121102_tracking_0038.fits	61288
P0007	20190830	FRB121102_tracking_0046.fits	119757
P0012	20190830	FRB121102_tracking_0114.fits	52231
P0016	20190830	FRB121102_tracking_0143.fits	22684
P0017	20190830	FRB121102_tracking_0145.fits	48497
P0026	20190830	FRB121102_tracking_0169.fits	106047
P0030	20190830	FRB121102_tracking_0192.fits	104876
P0031	20190830	FRB121102_tracking_0198.fits	105871
P0032	20190830	FRB121102_tracking_0199.fits	12950
P0037	20190830	FRB121102_tracking_0215.fits	104796
P0038	20190830	FRB121102_tracking_0218.fits	97375
P0046	20190830	FRB121102_tracking_0253.fits	66187
P0047	20190830	FRB121102_tracking_0265.fits	59224
P0050	20190830	FRB121102_tracking_0266.fits	105009
P0051	20190830	FRB121102_tracking_0268.fits	83271
P0052	20190830	FRB121102_tracking_0272.fits	108154
P0053	20190830	FRB121102_tracking_0276.fits	78979
P0054	20190830	FRB121102_tracking_0277.fits	85466
P0057	20190830	FRB121102_tracking_0291.fits	101189
P0060	20190830	FRB121102_tracking_0324.fits	102213
P0061	20190830	FRB121102_tracking_0325.fits	28241
P0063	20190830	FRB121102_tracking_0330.fits	60065
P0065	20190830	FRB121102_tracking_0333.fits	197784
P0069	20190830	FRB121102_tracking_0338.fits	54234
P0071	20190830	FRB121102_tracking_0342.fits	64103
P0074	20190830	FRB121102_tracking_0361.fits	82400
P0076	20190830	FRB121102_tracking_0368.fits	149320
P0079	20190830	FRB121102_tracking_0381.fits	70530
P0081	20190830	FRB121102_tracking_0388.fits	23278
P0082	20190830	FRB121102_tracking_0391.fits	86534
P0083	20190830	FRB121102_tracking_0397.fits	19325
P0084	20190830	FRB121102_tracking_0401.fits	81044
P0088	20190831	FRB121102_tracking-M01_0013.fits	73046

Table C.7: Information about the Fast number used to migrate the dataset(please see https://fast.cstcloud.cn/datavolume/detail For more detail)

PulseID	Obs_Date(UT)	FileName	Sample_ID
P0091	20190831	FRB121102_tracking-M01_0041.fits	94068
P0216	20190901	FRB121102_tracking-M01_0078.fits	90330
P0096	20190831	FRB121102_tracking-M01_0091.fits	88490
P0666	20190908	FRB121102_tracking-M01_0112.fits	31233
P0103	20190831	FRB121102_tracking-M01_0128.fits	104223
P0108	20190831	FRB121102_tracking-M01_0181.fits	30784
P0774	20190911	FRB121102_tracking-M01_0185.fits	33594
P0113	20190831	FRB121102_tracking-M01_0190.fits	101721
P0115	20190831	FRB121102_tracking-M01_0194.fits	120779
P0121	20190831	FRB121102_tracking-M01_0301.fits	34979
P0138	20190831	FRB121102_tracking-M01_0429.fits	17395
P0139	20190831	FRB121102_tracking-M01_0437.fits	49076
P0140	20190831	FRB121102_tracking-M01_0446.fits	6066
P0247	20190901	FRB121102_tracking-M01_0448.fits	62294
P0142	20190831	FRB121102_tracking-M01_0451.fits	72437
P0143	20190831	FRB121102_tracking-M01_0457.fits	92249
P0144	20190831	FRB121102_tracking-M01_0464.fits	55886
P0146	20190831	FRB121102_tracking-M01_0479.fits	103115
P0147	20190831	FRB121102_tracking-M01_0482.fits	57711
P0148	20190831	FRB121102_tracking-M01_0484.fits	100489
P0158	20190831	FRB121102_tracking-M01_0541.fits	14666
P0254	20190901	FRB121102_tracking-M01_0549.fits	41765
P0165	20190831	FRB121102_tracking-M01_0556.fits	45720