9 Proofs

9.1 Preliminary

Notations. Vectors (*e.g.*, $\mathbf{a}$) and matrices (*e.g.*, $\mathbf{A}$) marked with a "hat" (*e.g.*, $\hat{\mathbf{a}}$ and $\hat{\mathbf{A}}$) signify estimators of their corresponding population quantities, which are represented with a superscript asterisk ($\mathbf{a}^\star$ and $\mathbf{A}^\star$). Furthermore, the symbol $\mathcal{F}$ references a class of functions that perform task-specific mappings from $\mathbb{R}^r$ to $\mathbb{R}^c$, while $\mathcal{H}$ designates a class of feature mapping functions from $\mathbb{R}^d$ to $\mathbb{R}^r$. For the notation of integer sets, we adopt the convention $[n] = \{1, \ldots, n\}$. The notation $\tilde{O}$ is employed to represent an expression that hides polylogarithmic factors across all problem parameters.

Model Complexity. The measurement of complexity for a function class is commonly evaluated by its Gaussian complexity, as shown in popular work [6,21, 24]. The definition of empirical and population Gaussian Complexity for a vector-valued function class $\mathcal{Q}$, comprising functions $\mathbf{q}(\cdot) : \mathbb{R}^d \to \mathbb{R}^r$, and associated with a data matrix $\mathbf{X}$ containing N datapoints, is presented as follows:

$$\hat{\mathfrak{G}}_{\mathbf{X}}(\mathcal{Q}) = \mathbb{E}_{\mathbf{g}}\left[\sup_{\mathbf{q}\in\mathcal{Q}} \frac{1}{N}\sum_{i=1}^{N} g_i \mathbf{q}(\mathbf{x}_i)\right], \tag{13}$$

where $\mathbf{g} = \{g_i\}_{i\in[N]}$ is a matrix of Gaussian random variables, each g_i following an independent and identically distributed normal distribution $\mathcal{N}(0, 1)$. The population Gaussian complexity is estimated through the expectation $\mathfrak{G}_N(\mathcal{Q}) = \mathbb{E}_{\mathbf{X}}[\hat{\mathfrak{G}}_{\mathbf{X}}(\mathcal{Q})]$, which aggregates the empirical complexities over the dataset $\mathbf{X}$.

In parallel, the Rademacher Complexities for the same function class $\mathcal{Q}$ can be defined as:

$$\hat{\mathfrak{R}}_{\mathbf{X}}(\mathcal{Q}) = \mathbb{E}_{\epsilon}\left[\sup_{\mathbf{q}\in\mathcal{Q}} \frac{1}{N}\sum_{i=1}^{N} \epsilon_i \mathbf{q}(\mathbf{x}_i)\right], \tag{14}$$

where $\epsilon = \{\epsilon_i\}_{i\in[N]}$ is a matrix of random variables with each ϵ_i independently and identically distributed according to the Rademacher distribution, which equally assigns the values -1 and 1. The population Rademacher complexity, $\mathfrak{R}_N(\mathcal{Q})$, is inferred through the expectation $\mathbb{E}_{\mathbf{X}}[\hat{\mathfrak{R}}_{\mathbf{X}}(\mathcal{Q})]$, averaging the empirical complexities across the dataset $\mathbf{X}$.

Referencing [14], we find the inequality:

$$\mathfrak{G}_{\mathbf{X}}(\mathcal{H}) \leq 2\sqrt{\log N} \cdot \mathfrak{R}_{\mathbf{X}}(\mathcal{H}), \tag{15}$$

indicating a direct relationship between the Gaussian Complexity and the Rademacher Complexity.

9.2 Proof of Theorem 1

Proof. In Eq. (6), the transfer risk Δ_{transfer} quantifies the expected prediction risk relative to the optimal counterpart. This risk is exclusively concerned with

the shared representation and the task-specific mapping of the incremental task, an aspect rigorously explored in [24]. Adapting Theorem 3 from [24] to the FSCIL yields the following expression:

$$\Delta_{\text{transfer}} \leq O\Big(\Gamma \log(N) \cdot \Big[\frac{\Gamma(\mathcal{F}) \cdot \mathfrak{G}_N(\mathcal{H}) + \mathfrak{G}_N(\mathcal{F})}{\rho}\Big] \tag{16}$$

$$+ \Gamma \mathfrak{G}_M(\mathcal{F}) + \frac{\Gamma D_{\mathcal{X}^0}}{\rho N^2} + B\Big[\frac{1}{\rho} \cdot \sqrt{\frac{\log(2/\delta)}{N}} + \sqrt{\frac{\log(2/\delta)}{M}}\Big] + \epsilon\Big). \tag{17}$$

With this tailored formulation, we proceed to bound each of the complexity terms detailed in this expression, elucidating their roles and upper bounds within the FSCIL framework.

– For the calculation of $\mathfrak{G}_N(\mathcal{H})$, we consider a layer within an architecture-agnostic neural network, expressed as:

$$h^{(l)} = g^{(l)}\left(h^{(l-1)}\right), \tag{18}$$

where $h^{(l)}$ denotes the output at the l-th layer, and $g^{(l)}$ is the corresponding layer function. On this basis, we apply the established result $\mathfrak{G}_{\mathbf{X}}(\mathcal{H}) \leq 2\sqrt{\log N} \cdot \mathfrak{R}_{\mathbf{X}}(\mathcal{H})$ to determine the upper bound for $\mathfrak{G}_N(\mathcal{H})$:

$$N\hat{\mathfrak{R}}_{\mathbf{X}}(h(\mathbf{x}_i)) = \mathbb{E}_\epsilon\big[\sup_{h\in\mathcal{H}} \sum_{i=1}^{N} \epsilon_i h(\mathbf{x}_i)\big] \tag{19}$$

$$\leq \frac{1}{\lambda} \log \mathbb{E}_\epsilon\big[\sup_{h\in\mathcal{H}} \exp\big(\lambda \sum_{i=1}^{N} \epsilon_i h^{(L)}(\mathbf{x}_i)\big)\big] \tag{20}$$

$$\leq \frac{1}{\lambda} \log \mathbb{E}_\epsilon\big[\sup_{h\in\mathcal{H}} \exp\big(\lambda \sum_{i=1}^{N} \|\epsilon_i h^{(L)}(\mathbf{x}_i)\|_\infty\big)\big] \tag{21}$$

$$= \frac{1}{\lambda} \log \mathbb{E}_\epsilon\big[\sup_{h\in\mathcal{H}} \exp\big(\lambda \sum_{i=1}^{N} \|\epsilon_i g^{(L)}\left(h^{(L-1)}(\mathbf{x}_i)\right)\|_\infty\big)\big], \tag{22}$$

where $\lambda > 0$ is identified as a tuning factor, and L represents the total number of layers in the neural network. Assuming $Z(l) = \frac{\|h^{(l)}\|_\infty}{\|h^{(l-1)}\|_\infty}$, we can further refine the upper bound as follows:

$$N\hat{\mathfrak{R}}_{\mathbf{X}}(h(\mathbf{x}_i)) \leq \frac{1}{\lambda} \log \mathbb{E}_\epsilon\big[\sup_{h\in\mathcal{H}} \exp\big(\lambda \sum_{i=1}^{N} Z(L)\|\epsilon_i h^{(L-1)}(\mathbf{x}_i)\|_\infty\big)\big] \tag{23}$$

$$\leq \frac{1}{\lambda} \log \mathbb{E}_\epsilon\big[\exp\big(Z \cdot \lambda \sum_{i=1}^{N} \|\epsilon_i \mathbf{x}_i\|_\infty\big)\big], \tag{24}$$

where $Z = \prod_{l=1}^{L} Z(l)$. Additionally, let $x_{i,j}$ denote the j-th coordinate of $\mathbf{x}_i$, where j ranges from 1 to J. Exploiting the symmetry, we can elegantly

reformulate the expectation within the log as follows:

$$\mathbb{E}_\epsilon\left[\exp\left(Z\lambda\sum_{i=1}^{N}\|\epsilon_i\mathbf{x}_i\|_\infty\right)\right]=\mathbb{E}_\epsilon\left[\exp\left(Z\lambda\cdot\max_j\left|\sum_{i=1}^{N}\epsilon_i x_{i,j}\right|\right)\right] \tag{25}$$

$$\leq\sum_{j=1}^{J}\mathbb{E}_\epsilon\left[\exp\left(Z\lambda\cdot\left|\sum_{i=1}^{N}\epsilon_i x_{i,j}\right|\right)\right], \tag{26}$$

where $|\cdot|$ signifies the absolute value operation. By harnessing the principles of symmetry and the linearity of expectation, the aforementioned expression can be effectively upper bounded as follows:

$$\sum_{j=1}^{J}\mathbb{E}_\epsilon\left[\exp\left(Z\lambda\sum_{i=1}^{N}\epsilon_i x_{i,j}\right)+\exp\left(-Z\lambda\sum_{i=1}^{N}\epsilon_i x_{i,j}\right)\right] \tag{27}$$

$$=2\sum_{j=1}^{J}\mathbb{E}_\epsilon\left[\exp\left(Z\lambda\sum_{i=1}^{N}\epsilon_i x_{i,j}\right)\right] \tag{28}$$

$$=2\sum_{j=1}^{J}\prod_{i=1}^{N}\mathbb{E}_\epsilon\left[\exp\left(Z\lambda\epsilon_i x_{i,j}\right)\right] \tag{29}$$

$$=2\sum_{j=1}^{J}\prod_{i=1}^{N}\frac{\exp\left(Z\lambda x_{i,j}\right)+\exp\left(-Z\lambda x_{i,j}\right)}{2} \tag{30}$$

$$\leq 2\sum_{j=1}^{J}\exp\left(Z^2\lambda^2\sum_{i=1}^{N}x_{i,j}^2\right) \tag{31}$$

$$\leq 2J\max_j\exp\left(Z^2\lambda^2\sum_{i=1}^{N}x_{i,j}^2\right), \tag{32}$$

we then reintroduce the derived term into Equation (24), completing the analytical process:

$$\frac{1}{\lambda}\log\mathbb{E}_\epsilon\left[\exp\left(Z\cdot\lambda\sum_{i=1}^{N}\|\epsilon_i\mathbf{x}_i\|_\infty\right)\right]\leq\frac{1}{\lambda}\log\left(2J\max_j\exp\left(Z^2\lambda^2\sum_{i=1}^{N}x_{i,j}^2\right)\right) \tag{33}$$

$$=\frac{\log(2J)}{\lambda}+Z^2\lambda\max_j\sum_{i=1}^{N}x_{i,j}^2. \tag{34}$$

By selecting $\lambda=\sqrt{\frac{\log(2J)}{Z^2\max_j\sum_{i=1}^{N}x_{i,j}^2}}$, we are able to establish an upper bound for the preceding expression:

$$N\hat{\mathfrak{R}}_{\mathbf{X}}(\mathcal{H})\leq 2Z\sqrt{\log(2J)\max_j\sum_{i=1}^{N}x_{i,j}^2}, \tag{35}$$

consequently, this yields the following result:

$$\hat{\mathfrak{G}}_{\mathbf{X}}(\mathcal{H}) \leq 2\sqrt{\log N} \cdot \hat{\mathfrak{R}}_{\mathbf{X}}(\mathcal{H}) \tag{36}$$

$$\leq \frac{4Z\sqrt{\log N}}{N} \sqrt{\log(2J) \max_j \sum_{i=1}^{N} x_{i,j}^2} \tag{37}$$

$$\leq 4D \prod_{l=1}^{L} Z(l) \sqrt{\frac{\log N \log 2J}{N}}, \tag{38}$$

where $\|\mathbf{x}\|_\infty \leq D$.

– For the calculation of $\mathfrak{G}_N(\mathcal{F})$, we derive:

$$\mathfrak{G}_N(\mathcal{F}) = \frac{1}{N} \mathbb{E}_{\mathbf{g}} \big[\sup_{f \in \mathcal{F}} \sum_{i=1}^{N} g_i f(\mathbf{x}_i) \big] \tag{39}$$

$$\leq \frac{1}{N} \mathbb{E}_{\mathbf{g}} \big[\sum_{i=1}^{N} g_i \big] = 0. \tag{40}$$

Similarly, for $\mathfrak{G}_M(\mathcal{F})$, it follows that:

$$\mathfrak{G}_M(\mathcal{F}) \leq \frac{1}{M} \mathbb{E}_{\mathbf{g}} \big[\sup_{f \in \mathcal{F}} \sum_{i=1}^{M} g_i f(\mathbf{x}_i) \big] \leq \frac{1}{M} \mathbb{E}_{\mathbf{g}} \big[\sum_{i=1}^{M} g_i \big] = 0. \tag{41}$$

In conclusion, after incorporating all Gaussian complexity terms into Eq. (16), setting $\epsilon = 0$, and omitting the polylogarithmic factors, the upper bound for Δ_{transfer} can be expressed as:

$$\tilde{O}\Big(\frac{\Gamma}{\rho} \cdot \Gamma(\mathcal{F}) \cdot 4D \prod_{l=1}^{L} Z(l) \sqrt{\frac{\log 2J}{N}} + \frac{\Gamma D_{\mathcal{X}^0}}{\rho N^2} + B \big[\frac{1}{\rho} \cdot \sqrt{\frac{\log(2/\delta)}{N}} + \sqrt{\frac{\log(2/\delta)}{M}} \big] \Big). \tag{42}$$

The determinant of Δ_{transfer} is:

$$\eta \cdot \frac{1}{\sqrt{N}}, \tag{43}$$

where $\eta = \frac{\Gamma}{\rho} \cdot \Gamma(\mathcal{F}) \cdot 4D \prod_{l=1}^{L} Z(l) \sqrt{\log 2J}$.

9.3 Proof of the Corollary 1

According to Theorem 1, the transfer risk is encapsulated within the boundary:

$$\tilde{O}\Big(\frac{\Gamma}{\rho} \cdot \Gamma(\mathcal{F}) \cdot 4D \prod_{l=1}^{L} Z(l) \sqrt{\frac{\log 2J}{N}} + \frac{\Gamma D_{\mathcal{X}^0}}{\rho N^2} + B \big[\frac{1}{\rho} \cdot \sqrt{\frac{\log(2/\delta)}{N}} + \sqrt{\frac{\log(2/\delta)}{M}} \big] \Big). \tag{44}$$

Notably, the term $\prod_{l=1}^{L} Z(l)$ only involves the architecture of the backbones. Considering the two dominant types of backbone, Vision Transformers (ViTs) and Convolutional Neural Networks (CNNs), these architectures often incorporate layer or batch normalizations, thereby ensuring $Z(l) \approx 1$. This implies that increasing the number of layers does not necessarily increase the transfer risk, as long as the infinity norm of the output of each layer remains stable.

Particularly in ViTs, there exist serveral self-attention layers, defined as:

$$h^{(l+1)} = \text{softmax}\left(h^{(l)} W_a W_a^T h^{(l)}\right) h^{(l)}, \tag{45}$$

where W_a is the parameter of the self-attention layer.

The softmax function applied to $h^{(l)} W_a W_a^T h^{(l)}$ produces a normalized, non-negative weight matrix. Thus, $h^{(l+1)}$ is a convex combination of rows in $h^{(l)}$. A key property of convex combinations is that the infinity norm of the resultant vector will not exceed that of the individual vectors being combined. Therefore, in the self-attention layer:

$$Z(l) \leq 1. \tag{46}$$

This analysis suggests that ViTs, equipped with the self-attention mechanism, potentially exhibit better transferability compared to other architectures.

9.4 Proof of the Eq. (5)

Proof.

$$\begin{aligned}
&\Delta_{\text{AR}} = R_{\text{novel}}([\hat{f}_0, \hat{f}_t], \hat{\mathbf{h}}) - R_{\text{novel}}([f_0^\star, f_t^\star], \mathbf{h}^\star) \\
=&R_{\text{novel}}([\hat{f}_0, \hat{f}_t], \hat{\mathbf{h}}) - \big(R_{\text{novel}}(\hat{f}_t, \hat{\mathbf{h}}) - R_{\text{novel}}(\hat{f}_t, \hat{\mathbf{h}})\big) \\
&- R_{\text{novel}}([f_0^\star, f_t^\star], \mathbf{h}^\star) - \big(R_{\text{novel}}(f_t^\star, \mathbf{h}^\star) - R_{\text{novel}}(f_t^\star, \mathbf{h}^\star)\big) \\
=&\underbrace{R_{\text{novel}}(\hat{f}_t, \hat{\mathbf{h}}) - R_{\text{novel}}(f_t^\star, \mathbf{h}^\star)}_{\Delta_{\text{transfer}}} \\
&+ \underbrace{R_{\text{novel}}([\hat{f}_0, \hat{f}_t], \hat{\mathbf{h}}) - R_{\text{novel}}(\hat{f}_t, \hat{\mathbf{h}})}_{\Delta R_{\text{novel}}} \\
&- \underbrace{\big(R_{\text{novel}}([f_0^\star, f_t^\star], \mathbf{h}^\star) - R_{\text{novel}}(f_t^\star, \mathbf{h}^\star)\big)}_{0} \\
=&\Delta_{\text{transfer}} + \Delta R_{\text{novel}}.
\end{aligned} \tag{47}$$

9.5 Proofs in the Theorem 2

Proof. ΔR_{novel} encapsulates the adaptation risk, quantifying the divergence between the composite classifier and the task-specific classifier:

$$\begin{aligned}
\Delta R_{\text{novel}} =& R_{\text{novel}}([\hat{f}_0, \hat{f}_t], \hat{\mathbf{h}}) - R_{\text{novel}}(\hat{f}_t, \hat{\mathbf{h}}) && (48)\\
=& \mathbb{E}\left[\frac{1}{M}\sum_{m=1}^{M} \ell([\hat{f}_0, \hat{f}_t] \circ \hat{\mathbf{h}}(\mathbf{x}_{t,m}), y_{t,m})\right] \\
&- \mathbb{E}\left[\frac{1}{M}\sum_{m=1}^{M} \ell(\hat{f}_t \circ \hat{\mathbf{h}}(\mathbf{x}_{t,m}), y_{t,m})\right] && (49)\\
=& \mathbb{E}\left[\frac{1}{M}\sum_{m=1}^{M} \ell([\hat{f}_0(\hat{\mathbf{h}}(\mathbf{x}_{t,m})), \hat{f}_t(\hat{\mathbf{h}}(\mathbf{x}_{t,m}))], y_{t,m})\right] \\
&- \mathbb{E}\left[\frac{1}{M}\sum_{m=1}^{M} \ell(\hat{f}_t(\hat{\mathbf{h}}(\mathbf{x}_{t,m})), y_{t,m})\right]. && (50)
\end{aligned}$$

Defining W_0 and W_t as the classifier parameters for the base task and the incremental task t respectively, the aforementioned expression can be reformulated as follows:

$$\begin{aligned}
&\mathbb{E}\left[\frac{1}{M}\sum_{m=1}^{M} \ell([\hat{W}_0\hat{\mathbf{h}}(\mathbf{x}_{t,m}), \hat{W}_t\hat{\mathbf{h}}(\mathbf{x}_{t,m})], y_{t,m})\right] \\
&- \mathbb{E}\left[\frac{1}{M}\sum_{m=1}^{M} \ell([\hat{W}_t\hat{\mathbf{h}}(\mathbf{x}_{t,m})], y_{t,m})\right] && (51)\\
=& \mathbb{E}\left[\frac{1}{M}\sum_{m=1}^{M}\sum_{i=1}^{K_0+K_t} -y_{t,m,i}\log(\sigma_i[\hat{W}_0\hat{\mathbf{h}}(\mathbf{x}_{t,m}), \hat{W}_t\hat{\mathbf{h}}(\mathbf{x}_{t,m})])\right] \\
&- \mathbb{E}\left[\frac{1}{M}\sum_{m=1}^{M}\sum_{i=1}^{K_t} -y_{t,m,i}\log(\sigma_i[\hat{W}_t\hat{\mathbf{h}}(\mathbf{x}_{t,m})])\right], && (52)
\end{aligned}$$

where K_0 and K_t represent the number of classes in the base and incremental task respectively, and $\sigma_i[\cdot]$ denotes the i-th coordinate of the softmax function σ applied to $[\cdot]$. Define $z_{t\to 0} = \sum_{i=1}^{K_0}\exp(\phi_{0,i})$ for the base task with K_0 base classes, and $z_{t\to t} = \sum_{i=1}^{K_t}\exp(\phi_{t,i})$ for the incremental task with K_t novel classes. Additionally, set $z_j = \exp(\phi_{t,j})$. Here, ϕ signifies the logits of the classifier, t indicates that the sample is associated with the task t, and i or j specify the respective coordinate of the logits. If the correct label is j, then the above can be reformulated:

$$\begin{aligned}
\Delta R_{\text{novel}} &= \mathbb{E}\left[\frac{1}{M}\sum_{m=1}^{M}\left(\log(\frac{z_{t\to 0}+z_{t\to t}}{z_j}) - \log(\frac{z_{t\to t}}{z_j})\right)\right] && (53)\\
&= \mathbb{E}\left[\frac{1}{M}\sum_{m=1}^{M}\log(1+\frac{z_{t\to 0}}{z_{t\to t}})\right]. && (54)
\end{aligned}$$

Similarly, we can derive the following:

$$\Delta R_{\text{base}} = \mathbb{E}\left[\frac{1}{N}\sum_{n=1}^{N}\log(1+\frac{z_{0\to t}}{z_{0\to 0}})\right]. \tag{55}$$

Thus, the consistency risk can be expressed as:

$$\mathbb{E}\left[\frac{1}{M}\sum_{m=1}^{M}\log(1+\frac{z_{t\to 0}}{z_{t\to t}})\right]+\gamma\mathbb{E}\left[\frac{1}{N}\sum_{n=1}^{N}\log(1+\frac{z_{0\to t}}{z_{0\to 0}})\right]. \tag{56}$$

Utilizing Jensen's inequality, the preceding expression can be upper bounded as follows:

$$\frac{1}{M}\sum_{m=1}^{M}\log(1+\mathbb{E}\left[\frac{z_{t\to 0}}{z_{t\to t}}\right])+\gamma\frac{1}{N}\sum_{n=1}^{N}\log(1+\mathbb{E}\left[\frac{z_{0\to t}}{z_{0\to 0}}\right]). \tag{57}$$

Substituting the definition of z into the above equation, we replace the expectations with their approximations:

$$\frac{1}{M}\sum_{m=1}^{M}\log\left(1+\frac{K_0\exp(\phi_{t\to 0})}{K_t\exp(\phi_{t\to t})}\right)+\gamma\frac{1}{N}\sum_{n=1}^{N}\log\left(1+\frac{K_t\exp(\phi_{0\to t})}{K_0\exp(\phi_{0\to 0})}\right), \tag{58}$$

where $\phi_{t\to 0}$ represents the average logit for samples from incremental task t when evaluated for the base task 0; $\phi_{t\to t}$ denotes the average logit for samples within task t, reflecting its internal classification performance; $\phi_{0\to t}$ indicates the average logit for base task samples classified in the context of incremental task t; and $\phi_{0\to 0}$ signifies the average logit for samples from the base task when classified within the base task. The preceding expression can be upper bounded by:

$$\log\left(1+\frac{K_0}{K_t}v_t\right)\left(1+\frac{K_t}{K_0}v_0\right)^{\gamma} \tag{59}$$

where $v_t = \exp\left(\sup(\phi_{t\to 0})-\inf(\phi_{t\to t})\right)$, and $v_0 = \exp\left(\sup(\phi_{0\to t})-\inf(\phi_{0\to 0})\right)$.

To find the determinant of the consistency risk, three reasonable assumptions were posited:

$$\gamma = 1, \tag{60}$$

$$v_t = \frac{K_t}{K_0}\exp(-\nu), \tag{61}$$

$$v_0 = \frac{K_0}{K_t}\exp(-\nu), \tag{62}$$

where ν denotes the classification margin discrepancy, quantified by the difference between the infimum of the classifier logits for correct classification and the supremum for incorrect classification.

Table 3: Classification Accuracy (%) on *mini*-ImageNet in the 5-way 5-shot FSCIL Setting. The notation ViT-B/16 denotes the ViT-Base model [5] with a designated patch size of 16. The symbols ⋄ and † signify models derived from the CLIP [19] and those pre-trained on the ImageNet-1K dataset, respectively. The symbol ↓ indicates preference for lower values.

Method	Backbone	Accuracy in each session(%)									DR↓(%)
		0	1	2	3	4	5	6	7	8	
CEC [32]	ResNet18	72.00	66.83	62.97	59.43	56.70	53.73	51.19	49.24	47.63	33.85
MCNet [10]	ResNet18	72.33	67.70	63.50	60.34	57.59	54.70	52.13	50.41	49.08	32.14
FACT [38]	ResNet18	72.56	69.63	66.38	62.77	60.6	57.33	54.34	52.16	50.49	30.42
ALICE [17]	ResNet18	80.6	70.6	67.4	64.5	62.5	60.0	57.8	56.8	55.7	30.89
CABD [37]	ResNet18	74.65	70.70	66.81	63.63	61.36	58.14	55.59	54.23	53.39	28.48
SAVC [22]	ResNet18	81.12	76.14	72.43	68.92	66.48	62.95	59.92	58.39	57.11	29.60
NC-FSCIL [30]	ResNet12	84.02	76.80	72.00	67.83	66.35	64.04	61.46	59.54	58.31	30.60
CoCoOp$^{\diamond}$ [39]	ViT-B/16	94.2	93.4	90.2	86.6	85.2	84.1	83.8	83.6	82.7	12.21
Prompt$^{\diamond}$ [31]	ViT-B/16	**95.4**	**94.4**	**93.4**	**93.1**	**92.1**	**91.4**	**91.4**	**90.7**	90.0	5.66
ours	ResNet18	84.23	81.33	77.01	74.62	71.66	68.74	65.97	63.83	62.17	26.19
ours$^{\diamond}$	ViT-B/16	93.16	93.15	91.17	90.54	90.53	90.04	89.05	89.13	89.04	4.42
ours†	ViT-B/16	93.73	92.75	91.82	91.22	90.87	91.25	90.51	90.52	**90.48**	**3.47**

To simplify Eq. (59), and incorporate the classification bias α into ρ, the determinant of the consistency risk can be expressed as:

$$\log(1 + \frac{\kappa}{\exp(\nu)}), \tag{63}$$

where κ is given by $\exp(\alpha) + \frac{1}{\exp(\alpha)}$.

Notably, when $\alpha = 0$, which implies no bias between the base and novel classes, κ reaches its minimum.

10 Supplementary Experiments

10.1 Implementation Details

Building on prior research [5], our scratch-trained ViT models utilize masked patch prediction for self-supervision to boost performance. Inspired by [33], we implement vanilla mixup in our training. Additionally, following [22], auto augmentation is applied to refine our training process. For label smoothing, we adjust the smoothing factor to 0.5.

10.2 Comparative Experiments

As shown in Table 3 and 4, the performance of our models mirrors trends with those observed on the CUB200 dataset. This indicates the adaptability and efficacy of our approach in enhancing the training process, even amidst the distinct characteristics and challenges presented by each dataset. Notably, our models demonstrate their flexibility and robustness in addressing dataset-specific issues such as category overlap, small image sizes, and lack of background information.

Table 4: Classification Accuracy (%) on CIFAR100 in the 5-way 5-shot FSCIL Setting. The notation ViT-B/16 specifically denotes the ViT-Base model [5] with a designated patch size of 16.

Method	Backbone	Accuracy in each session(%)									DR↓(%)
		0	1	2	3	4	5	6	7	8	
CEC [32]	ResNet20	73.07	68.88	65.26	61.19	58.09	55.57	53.22	51.34	49.14	32.75
MCNet [10]	ResNet20	73.30	69.34	65.72	61.70	58.75	56.44	54.59	53.01	50.72	30.80
FACT [38]	ResNet20	74.60	72.09	67.56	63.52	61.38	58.36	56.28	54.24	52.10	30.16
SAVC [22]	ResNet20	78.77	73.31	69.31	64.93	61.70	59.25	57.13	55.19	53.12	32.56
ALICE [17]	ResNet18	79.0	70.5	67.1	63.4	61.2	59.2	58.1	56.3	54.1	31.52
CABD [37]	ResNet18	79.45	75.63	72.00	68.09	65.54	62.59	60.76	58.35	56.56	28.81
NC-FSCIL [30]	ResNet12	82.52	76.82	73.34	69.68	66.19	62.85	60.96	59.02	56.11	32.00
CoCoOp$^{\diamond}$ [39]	ViT-B/16	82.2	78.5	76.7	74.7	73.2	71.7	71.0	70.3	69.9	14.96
Prompt$^{\diamond}$ [31]	ViT-B/16	86.5	84.2	81.8	79.73	78.0	77.1	76.0	74.7	74.4	13.99
ours	ResNet18	82.56	78.44	75.18	69.61	67.33	64.70	63.64	62.23	60.50	26.72
ours$^{\diamond}$	ViT-B/16	89.46	86.41	84.04	82.14	80.73	79.17	78.02	77.12	75.50	15.60
ours†	ViT-B/16	**89.55**	**86.96**	**85.60**	**83.20**	**83.00**	**82.40**	**81.93**	**81.02**	**79.24**	**11.51**

10.3 Ablation Study

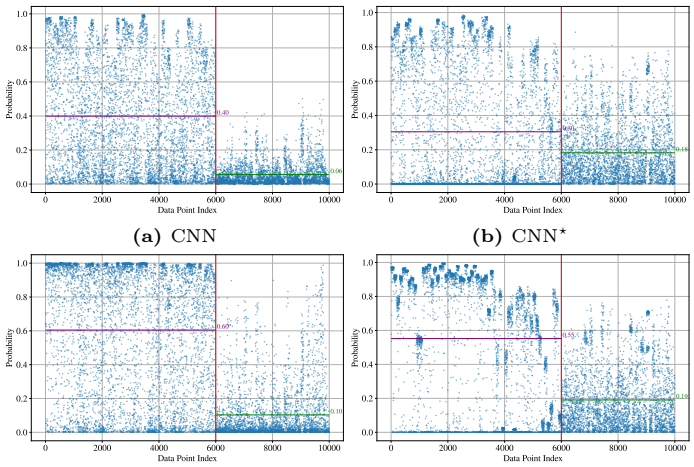
In this section, we evaluate the influence of four components of our ResNet18 model on the *mini*-ImageNet dataset, as detailed in Table 5. These components demonstrate diverse impacts on performance. Specifically, expanding the base class training dataset enhances accuracy by approximately 10% per session and decreases the DR metric by around 7%. This finding underscores the effectiveness of dataset expansion for addressing the FSCIL challenge, aligning with our theoretical insights. Moreover, stabilization, constraint application, and label smoothing each contribute positively to FSCIL performance to different extents, underscoring their significance and reaffirming our theoretical analysis.

10.4 Model Overconfidence

Observations of Fig. 7 (a) and (b) reveal a notable shift in the prediction accuracy of our model: a decrease of approximately 10% in the accuracy for base class predictions and an increase of around 12% in the accuracy for new class predictions. This shift highlights the enhanced transferability of our model, which is primarily attributed to the designed moderation in confidence levels. Similar trends are evident in Fig. 7 (c) and (d).

Table 5: Ablation Study on *mini*-ImageNet in the 5-Way 5-Shot FSCIL Setting. With ResNet18 as the backbone, our study encompasses: "Stabilize", enhacing output stability via *tanh* function after normalization; "Constrain", limiting representation values using layer normalization and *tanh* function; "LS" (Label Smoothing) to reduce model overconfidence; "Expand", expanding the dataset through the use of mixup techniques [18, 33] and various data augmentation methods [1, 7].

Stabilize	Constrain	LS	Expand	Accuracy in each session(%)									DR↓(%)
				0	1	2	3	4	5	6	7	8	
				70.70	65.75	61.90	58.42	55.30	52.37	49.88	47.68	45.95	35.01
✓				69.41	64.63	60.71	57.14	54.73	51.88	49.25	47.10	46.32	33.27
	✓			69.60	64.92	60.98	57.62	54.65	51.80	49.42	47.50	46.34	33.42
		✓		68.53	63.55	59.68	56.42	53.45	50.77	48.37	46.62	45.16	34.10
✓	✓			72.73	67.33	63.11	59.33	56.36	53.30	50.63	48.41	46.93	35.47
✓		✓		72.41	67.63	63.71	60.14	56.73	53.88	51.25	49.10	47.32	34.65
	✓	✓		72.11	67.16	63.04	58.80	55.57	53.56	50.98	48.97	47.30	34.41
✓	✓	✓		73.23	68.33	64.01	60.62	57.66	54.74	51.97	49.83	48.17	34.22
			✓	80.02	77.44	72.89	70.35	67.10	64.41	61.40	59.26	57.20	28.52
✓			✓	81.22	78.52	73.42	71.57	68.35	65.94	62.98	60.09	58.70	27.73
	✓		✓	79.80	77.70	73.35	71.28	68.20	65.45	62.38	59.78	57.88	27.47
		✓	✓	81.42	78.89	74.71	72.05	69.68	66.44	63.74	61.83	60.06	26.23
✓	✓	✓	✓	**84.23**	**81.33**	**77.01**	**74.62**	**71.66**	**68.74**	**65.97**	**63.83**	**62.17**	**26.19**



(a) CNN **(b)** CNN⋆

(c) ViT **(d)** ViT⋆

Fig. 7: Probability Distribution of Correct Predictions with ResNet18 and ViT-S/6 Backbones on *mini*-ImageNet. The horizontal axis represents the index of each test sample, while the vertical axis denotes the probability of correct prediction. Each subfigure is divided by a red line; the left segment displays base class test samples, and the right segment displays novel class samples. The average accuracy for base class samples is marked by a purple line, and that for novel class samples is marked by a green line. Besides, the symbol ⋆ denotes the model in our implementation.