# Provable Benefit of Cutout and CutMix for Feature Learning

Chulhee “Charlie” Yun (KAIST AI)

Based on joint work with Junsoo Oh (KAIST AI)

OptiML Optimization & Machine Learning Laboratory
KAIST AI
Kim Jaechul Graduate School

# Data Augmentation Zoo



(cat,dog) = (0,1)

Original

# Data Augmentation Zoo



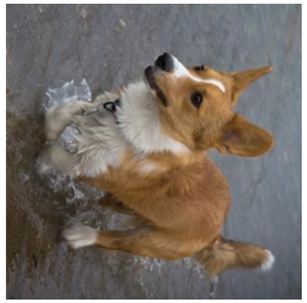
(cat,dog) = (0,1)

Original



(cat,dog) = (0,1)

Flipping



(cat,dog) = (0,1)

Rotation

# Data Augmentation Zoo



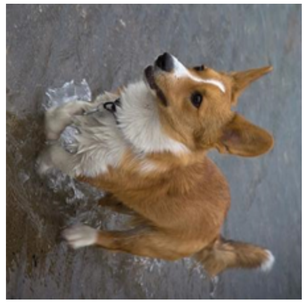
(cat,dog) = (0,1)

Original



(cat,dog) = (0,1)

Flipping



(cat,dog) = (0,1)

Rotation



(cat,dog) = (0.4,0.6)

Mixup

[Zhang et al. (2017); DeVries and Taylor (2017); Yun et al. (2019); ...]

# Data Augmentation Zoo



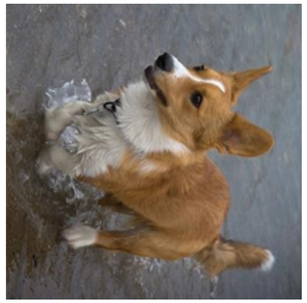
(cat,dog) = (0,1)

Original



(cat,dog) = (0,1)

Flipping



(cat,dog) = (0,1)

Rotation



(cat,dog) = (0.4,0.6)

Mixup



(cat,dog) = (0,1)

Cutout

[Zhang et al. (2017); DeVries and Taylor (2017); Yun et al. (2019); ...]

# Data Augmentation Zoo



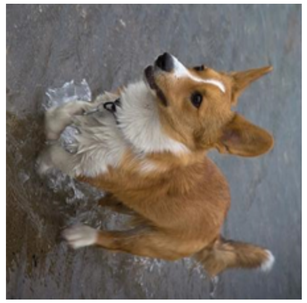
(cat,dog) = (0,1)

Original



(cat,dog) = (0,1)

Flipping



(cat,dog) = (0,1)

Rotation



(cat,dog) = (0.4,0.6)

Mixup



(cat,dog) = (0.4,0.6)

CutMix



(cat,dog) = (0,1)

Cutout

[Zhang et al. (2017); DeVries and Taylor (2017); Yun et al. (2019); ...]

# Data Augmentation Zoo



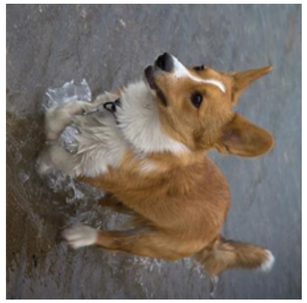
(cat,dog) = (0,1)

Original



(cat,dog) = (0,1)

Flipping



(cat,dog) = (0,1)

Rotation



(cat,dog) = (0.4,0.6)

Mixup

**Patch-level Data Augmentation**



(cat,dog) = (0.4,0.6)

CutMix



(cat,dog) = (0,1)

Cutout

**Oh & Yun. Provable Benefit of Cutout and CutMix for Feature Learning. NeurIPS 2024 (Spotlight)**

# TL;DR

We investigate the benefit of Cutout and CutMix for learning “features” from data, and show ERM < Cutout < CutMix in extracting rare features

# Characteristics of Images



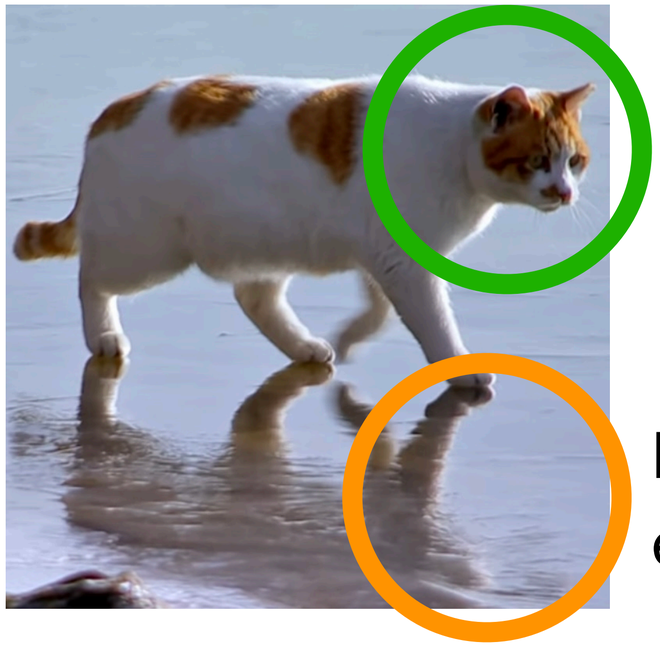
# Characteristics of Images



Label-dependent **feature**

e.g. Cat's face

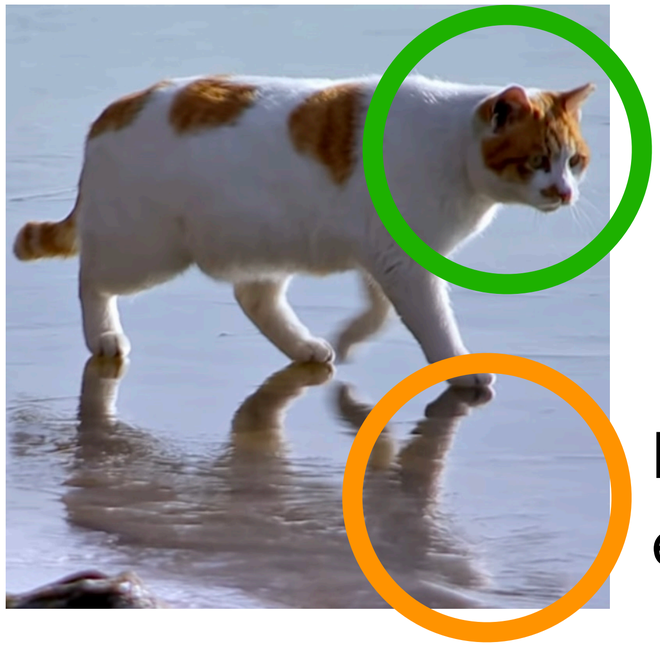
# Characteristics of Images



Label-dependent **feature**
e.g. Cat's face

Label-independent **noise**
e.g. background

# Characteristics of Images

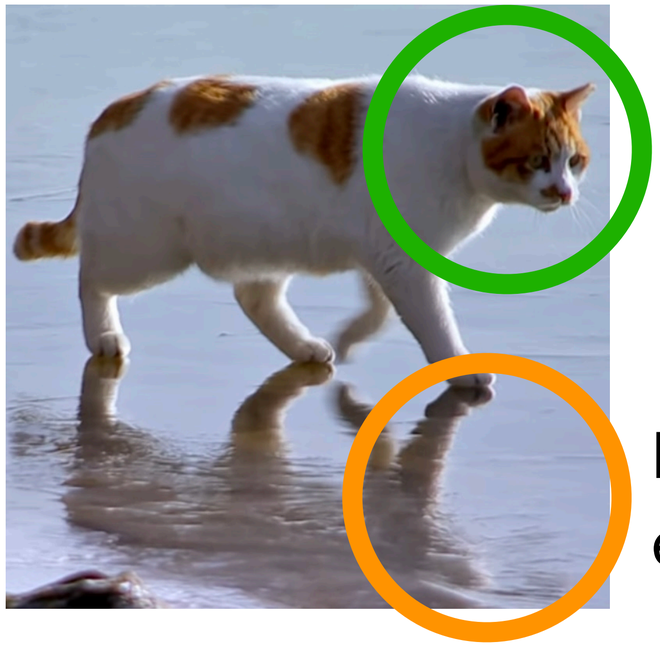


Label-dependent **feature**
e.g. Cat's face

Label-independent **noise**
e.g. background

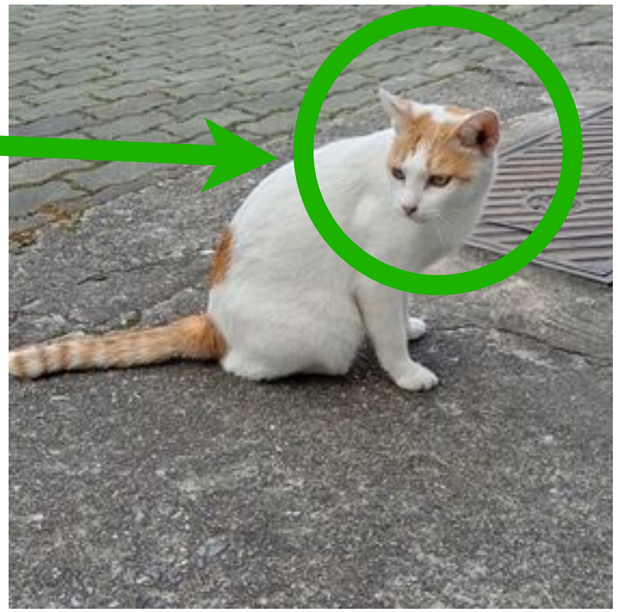


# Characteristics of Images

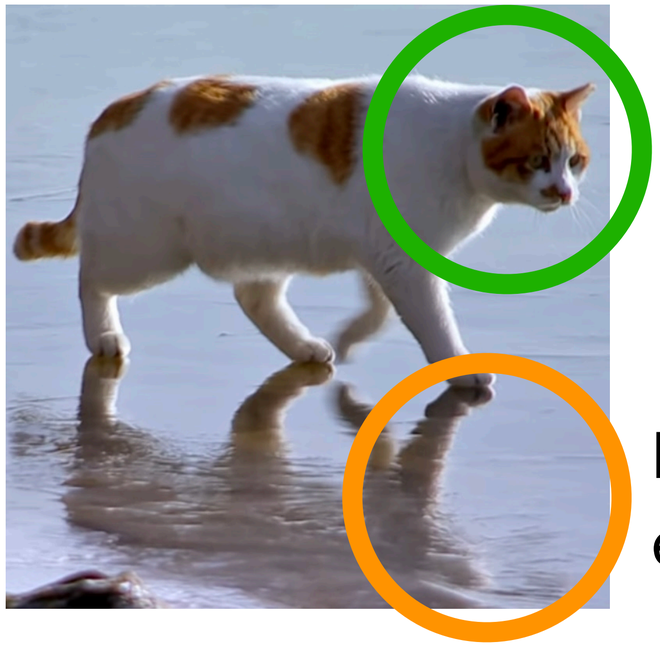


Label-dependent **feature**
e.g. Cat's face

Label-independent **noise**
e.g. background



# Characteristics of Images



Label-dependent **feature**
e.g. Cat's face

Label-independent **noise**
e.g. background





# Data Distribution

A popular approach for identifying the mechanism of feature learning is through the assumption of feature-noise data distribution.

# Data Distribution

A popular approach for identifying the mechanism of feature learning is through the assumption of feature-noise data distribution.

Initially proposed by `[Allen-Zhu & Li (2023)]`, first arXived in 2020, the framework has been widely adopted, helping us understand

# Data Distribution

A popular approach for identifying the mechanism of feature learning is through the assumption of feature-noise data distribution.

Initially proposed by [Allen-Zhu & Li (2023)], first arXived in 2020, the framework has been widely adopted, helping us understand

- Vanilla training vs ensemble vs knowledge distillation [Allen-Zhu & Li (2023)]

# Data Distribution

A popular approach for identifying the mechanism of feature learning is through the assumption of feature-noise data distribution.

Initially proposed by [Allen-Zhu & Li (2023)], first arXived in 2020, the framework has been widely adopted, helping us understand

- Vanilla training vs ensemble vs knowledge distillation [Allen-Zhu & Li (2023)]
- Benign overfitting in 2-layer CNNs [Cao et al. (2022); Kou et al. (2023)]

# Data Distribution

A popular approach for identifying the mechanism of feature learning is through the assumption of feature-noise data distribution.

Initially proposed by [Allen-Zhu & Li (2023)], first arXived in 2020, the framework has been widely adopted, helping us understand

- Vanilla training vs ensemble vs knowledge distillation [Allen-Zhu & Li (2023)]
- Benign overfitting in 2-layer CNNs [Cao et al. (2022); Kou et al. (2023)]
- GD vs Adam vs momentum GD [Zou et al. (2023a); Jelassi & Li (2022)]

# Data Distribution

A popular approach for identifying the mechanism of feature learning is through the assumption of feature-noise data distribution.

Initially proposed by [Allen-Zhu & Li (2023)], first arXived in 2020, the framework has been widely adopted, helping us understand

- Vanilla training vs ensemble vs knowledge distillation [Allen-Zhu & Li (2023)]
- Benign overfitting in 2-layer CNNs [Cao et al. (2022); Kou et al. (2023)]
- GD vs Adam vs momentum GD [Zou et al. (2023a); Jelassi & Li (2022)]
- Vanilla training vs Mixup [Zou et al. (2023b); Chidambaram et al. (2023)]

# Data Distribution

A popular approach for identifying the mechanism of feature learning is through the assumption of feature-noise data distribution.

Initially proposed by [Allen-Zhu & Li (2023)], first arXived in 2020, the framework has been widely adopted, helping us understand

- Vanilla training vs ensemble vs knowledge distillation [Allen-Zhu & Li (2023)]
- Benign overfitting in 2-layer CNNs [Cao et al. (2022); Kou et al. (2023)]
- GD vs Adam vs momentum GD [Zou et al. (2023a); Jelassi & Li (2022)]
- Vanilla training vs Mixup [Zou et al. (2023b); Chidambaram et al. (2023)]
- and many more...

# Data Distribution

We now define our feature-noise data distribution $(\mathbf{X}, y) \sim \mathscr{D}$.

# Data Distribution

We now define our feature-noise data distribution $(\mathbf{X}, y) \sim \mathscr{D}$.

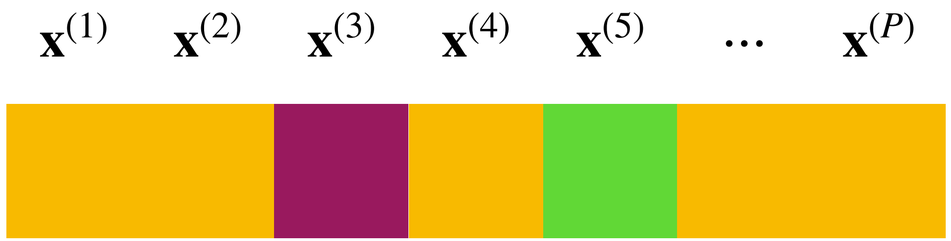
Label $y \in \{\pm 1\}$ is sampled uniformly at random, and
data point $\mathbf{X} = (\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(P)})$ consists of $P$ "patches" of three different kinds:

# Data Distribution

We now define our feature-noise data distribution $(\mathbf{X}, y) \sim \mathscr{D}$.

Label $y \in \{\pm 1\}$ is sampled uniformly at random, and
data point $\mathbf{X} = (\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(P)})$ consists of $P$ "patches" of three different kinds:

$$\mathbf{X} \in \mathbb{R}^{d \times P}$$



# Data Distribution

We now define our feature-noise data distribution $(\mathbf{X}, y) \sim \mathscr{D}$.

Label $y \in \{\pm 1\}$ is sampled uniformly at random, and
data point $\mathbf{X} = (\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(P)})$ consists of $P$ "patches" of three different kinds:

$$\mathbf{X} \in \mathbb{R}^{d \times P}$$

$\mathbf{x}^{(1)}$ $\mathbf{x}^{(2)}$ $\mathbf{x}^{(3)}$ $\mathbf{x}^{(4)}$ $\mathbf{x}^{(5)}$ $\cdots$ $\mathbf{x}^{(P)}$



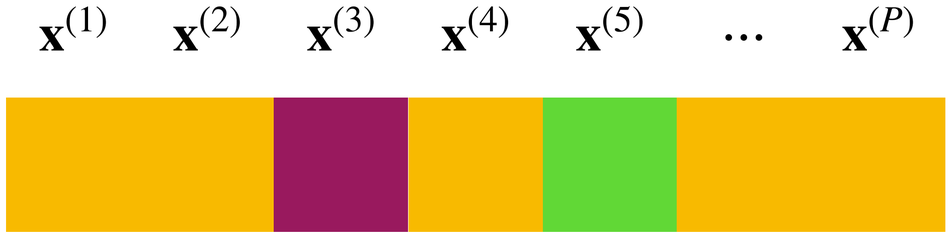
One **Feature** Patch

One **Dominant Noise** Patch

$P - 2$ **Background Noise** Patches

# Data Distribution

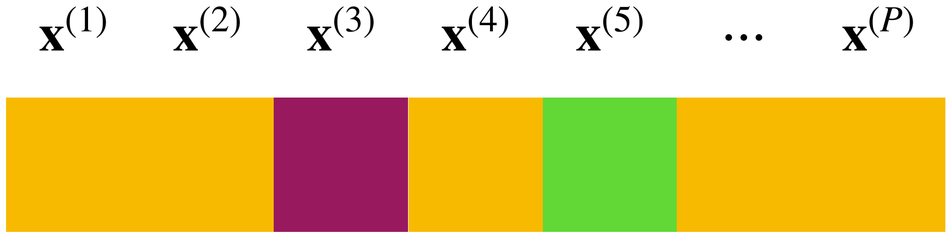
**Feature Patch**. For each given label $y \in \{\pm 1\}$, there are $K$ feature vectors $\{\mathbf{v}_{y,k}\}_{k\in[K]}$ which occur with conditional probabilities $\{\rho_k\}_{k\in[K]}$.




**Here, $\{\mathbf{v}_{s,k}\}_{s\in\{\pm 1\},k\in[K]}$ is orthonormal, $\rho_1 \geq \rho_2 \geq \ldots \geq \rho_K$ , and $\sum_{k=1}^{K} \rho_k = 1$.**

# Data Distribution

**Feature Patch**. For each given label $y \in \{\pm 1\}$, there are $K$ feature vectors $\{\mathbf{v}_{y,k}\}_{k\in[K]}$ which occur with conditional probabilities $\{\rho_k\}_{k\in[K]}$.

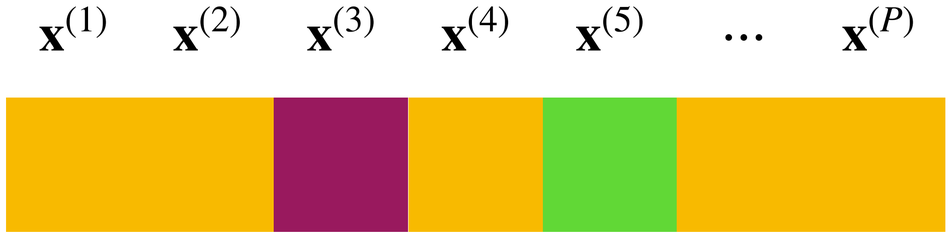



There are three kinds of features, with different levels of **rarity** (small $\rho_k$ means rare)

**Here,** $\{\mathbf{v}_{s,k}\}_{s\in\{\pm 1\},k\in[K]}$ **is orthonormal,** $\rho_1 \geq \rho_2 \geq \ldots \geq \rho_K$ **, and** $\sum_{k=1}^{K} \rho_k = 1$**.**

# Data Distribution

**Feature Patch**. For each given label $y \in \{\pm 1\}$, there are $K$ feature vectors $\{\mathbf{v}_{y,k}\}_{k\in[K]}$ which occur with conditional probabilities $\{\rho_k\}_{k\in[K]}$.



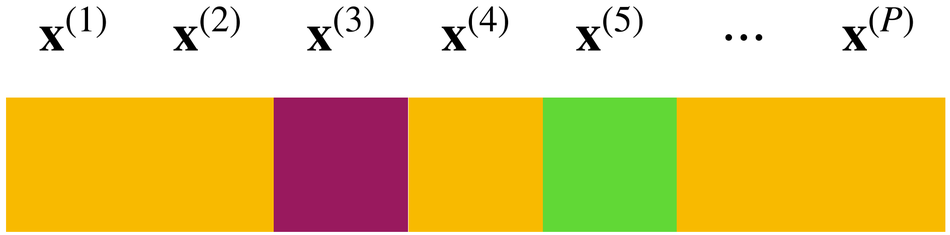

There are three kinds of features, with different levels of **rarity** (small $\rho_k$ means rare)

- **Common** $\mathcal{K}_C \subset [K]$, with frequency $\rho_k = \Theta(1)$ for $k \in \mathcal{K}_C$

**Here,** $\{\mathbf{v}_{s,k}\}_{s\in\{\pm 1\},k\in[K]}$ **is orthonormal,** $\rho_1 \geq \rho_2 \geq \ldots \geq \rho_K$ **, and** $\sum_{k=1}^{K} \rho_k = 1$**.**

# Data Distribution

**Feature Patch**. For each given label $y \in \{\pm 1\}$, there are $K$ feature vectors $\{\mathbf{v}_{y,k}\}_{k\in[K]}$ which occur with conditional probabilities $\{\rho_k\}_{k\in[K]}$.



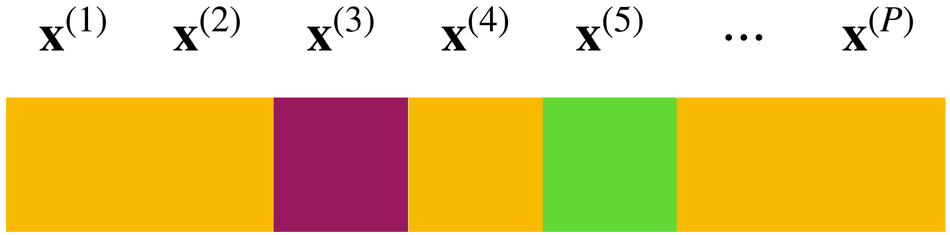

There are three kinds of features, with different levels of **rarity** (small $\rho_k$ means rare)

- **Common** $\mathcal{K}_C \subset [K]$, with frequency $\rho_k = \Theta(1)$ for $k \in \mathcal{K}_C$
- **Rare** $\mathcal{K}_R \subset [K]$, with frequency $\rho_k = \Theta(\rho_R) = o(1)$ for $k \in \mathcal{K}_R$

**Here,** $\{\mathbf{v}_{s,k}\}_{s\in\{\pm 1\},k\in[K]}$ **is orthonormal,** $\rho_1 \geq \rho_2 \geq \ldots \geq \rho_K$ **, and** $\sum_{k=1}^{K} \rho_k = 1$**.**

# Data Distribution

**Feature Patch**. For each given label $y \in \{\pm 1\}$, there are $K$ feature vectors $\{\mathbf{v}_{y,k}\}_{k\in[K]}$ which occur with conditional probabilities $\{\rho_k\}_{k\in[K]}$.



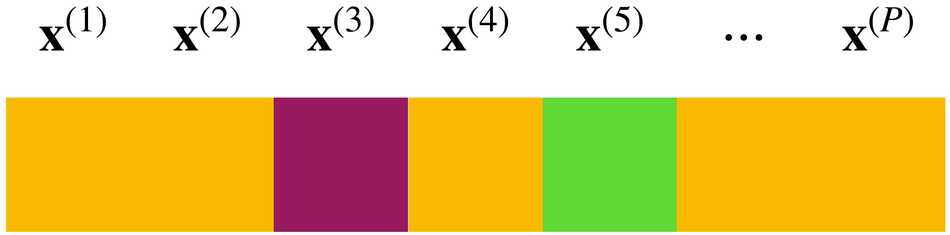

There are three kinds of features, with different levels of **rarity** (small $\rho_k$ means rare)

- **Common** $\mathcal{K}_C \subset [K]$, with frequency $\rho_k = \Theta(1)$ for $k \in \mathcal{K}_C$
- **Rare** $\mathcal{K}_R \subset [K]$, with frequency $\rho_k = \Theta(\rho_R) = o(1)$ for $k \in \mathcal{K}_R$
- **Extremely Rare** $\mathcal{K}_E \subset [K]$, with frequency $\rho_k = \Theta(\rho_E) = o(\rho_R)$ for $k \in \mathcal{K}_E$

**Here,** $\{\mathbf{v}_{s,k}\}_{s\in\{\pm1\},k\in[K]}$ **is orthonormal,** $\rho_1 \geq \rho_2 \geq \ldots \geq \rho_K$ **, and** $\sum_{k=1}^{K} \rho_k = 1$**.**

# Data Distribution

**Feature Patch**. For each given label $y \in \{\pm 1\}$, there are $K$ feature vectors $\{\mathbf{v}_{y,k}\}_{k\in[K]}$ which occur with conditional probabilities $\{\rho_k\}_{k\in[K]}$.




There are three kinds of features, with different levels of **rarity** (small $\rho_k$ means rare)

- **Common** $\mathscr{K}_C \subset [K]$, with frequency $\rho_k = \Theta(1)$ for $k \in \mathscr{K}_C$
- **Rare** $\mathscr{K}_R \subset [K]$, with frequency $\rho_k = \Theta(\rho_R) = o(1)$ for $k \in \mathscr{K}_R$
- **Extremely Rare** $\mathscr{K}_E \subset [K]$, with frequency $\rho_k = \Theta(\rho_E) = o(\rho_R)$ for $k \in \mathscr{K}_E$

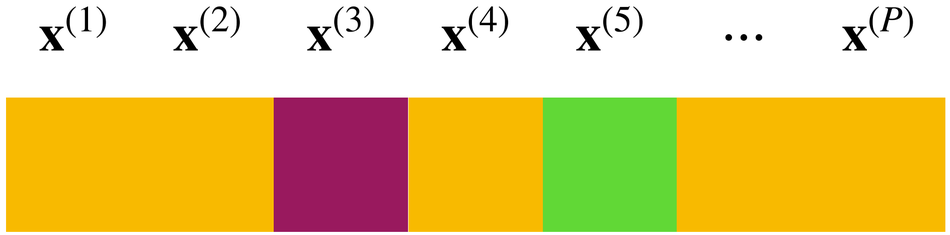
Given the choice of $y$, choose $\mathbf{v}$ from $\{\mathbf{v}_{y,k}\}_{k\in[K]}$ with probability $\{\rho_k\}_{k\in[K]}$ and position $p^* \in [P]$ uniformly at random, set $\mathbf{x}^{(p^*)} = \mathbf{v}$.

**Here,** $\{\mathbf{v}_{s,k}\}_{s\in\{\pm 1\},k\in[K]}$ **is orthonormal,** $\rho_1 \geq \rho_2 \geq \ldots \geq \rho_K$ **, and** $\sum_{k=1}^{K} \rho_k = 1$**.**

# Data Distribution

**Dominant Noise Patch.**

Sample patch index $\tilde{p} \neq p^*$. Set



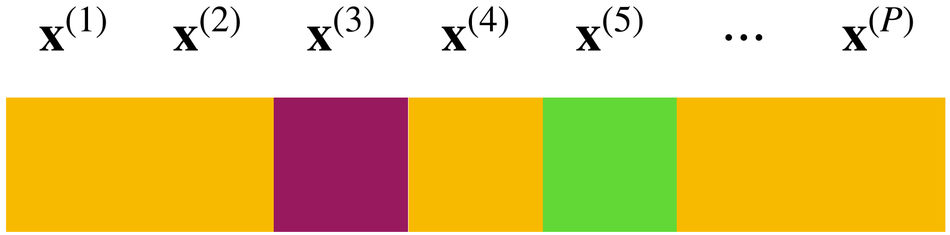

**Here,** $\mathbf{\Lambda} = \mathbf{I} - \sum \mathbf{v}_{s,k}\mathbf{v}_{s,k}^{\top}$ **and** $\sigma_{\mathrm{d}} \gg \sigma_{\mathrm{b}}$.

# Data Distribution

**Dominant Noise Patch.**

Sample patch index $\tilde{p} \neq p^*$. Set

$$\mathbf{x}^{(\tilde{p})} = \alpha\mathbf{u} + \xi^{(\tilde{p})},$$

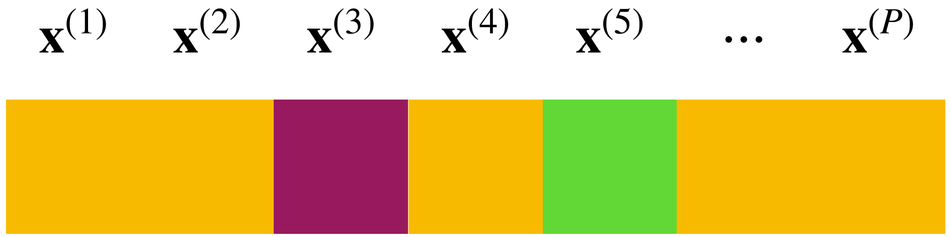



where $\alpha\mathbf{u}$ is "feature noise" and $\xi^{(\tilde{p})} \sim N(\mathbf{0}, \sigma_{\mathrm{d}}^2\mathbf{\Lambda})$.

The feature noise is drawn $\mathbf{u} \sim \mathrm{Unif}\{\mathbf{v}_{+1,1}, \mathbf{v}_{-1,1}\}$ to model "confusing" features.

**Here, $\mathbf{\Lambda} = \mathbf{I} - \sum \mathbf{v}_{s,k}\mathbf{v}_{s,k}^\top$ and $\sigma_{\mathrm{d}} \gg \sigma_{\mathrm{b}}$.**

# Data Distribution




**Dominant Noise Patch**.
Sample patch index $\tilde{p} \neq p^*$. Set

$$\mathbf{x}^{(\tilde{p})} = \alpha\mathbf{u} + \xi^{(\tilde{p})},$$

where $\alpha\mathbf{u}$ is "feature noise" and $\xi^{(\tilde{p})} \sim N(\mathbf{0}, \sigma_{\mathrm{d}}^2\mathbf{\Lambda})$.

The feature noise is drawn $\mathbf{u} \sim \mathrm{Unif}\{\mathbf{v}_{+1,1}, \mathbf{v}_{-1,1}\}$ to model "confusing" features.

**Background Noise Patch**. The remaining $P-2$ patches $p \in [P]\backslash\{p^*, \tilde{p}\}$ are filled with independent and identically distributed Gaussian noise $\mathbf{x}^{(p)} = \xi^{(p)} \sim N(\mathbf{0}, \sigma_{\mathrm{b}}^2\mathbf{\Lambda})$.

**Here, $\mathbf{\Lambda} = \mathbf{I} - \sum \mathbf{v}_{s,k}\mathbf{v}_{s,k}^\top$ and $\sigma_{\mathrm{d}} \gg \sigma_{\mathrm{b}}$.**

# Prediction Model

We define **2-Layer CNN** $f_{\mathbf{W}} : \mathbb{R}^{d \times P} \to \mathbb{R}$**,**
parameterized by $\mathbf{W} = \{\mathbf{w}_1, \mathbf{w}_{-1}\} \in \mathbb{R}^{d \times 2}$.

# Prediction Model

We define **2-Layer CNN** $f_{\mathbf{W}} : \mathbb{R}^{d \times P} \to \mathbb{R}$,
parameterized by $\mathbf{W} = \{\mathbf{w}_1, \mathbf{w}_{-1}\} \in \mathbb{R}^{d \times 2}$.

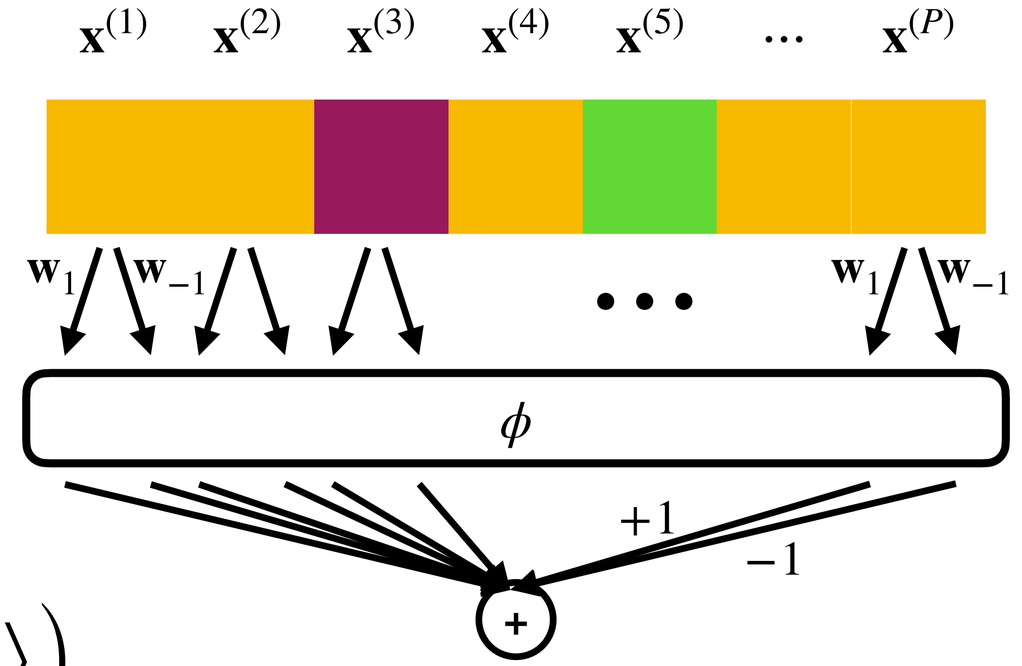
For input $\mathbf{X} = (\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(P)}) \in \mathbb{R}^{d \times P}$,
we define

$$f_{\mathbf{W}}(\mathbf{X}) = \sum_{p \in [P]} \phi\left(\left\langle \mathbf{w}_1, \mathbf{x}^{(p)} \right\rangle\right) - \sum_{p \in [P]} \phi\left(\left\langle \mathbf{w}_{-1}, \mathbf{x}^{(p)} \right\rangle\right).$$

# Prediction Model

We define **2-Layer CNN** $f_{\mathbf{W}} : \mathbb{R}^{d \times P} \to \mathbb{R}$,
parameterized by $\mathbf{W} = \{\mathbf{w}_1, \mathbf{w}_{-1}\} \in \mathbb{R}^{d \times 2}$.

For input $\mathbf{X} = (\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(P)}) \in \mathbb{R}^{d \times P}$,
we define




$$f_{\mathbf{W}}(\mathbf{X}) = \sum_{p \in [P]} \phi\left(\left\langle \mathbf{w}_1, \mathbf{x}^{(p)} \right\rangle\right) - \sum_{p \in [P]} \phi\left(\left\langle \mathbf{w}_{-1}, \mathbf{x}^{(p)} \right\rangle\right).$$

# Prediction Model

$$f_{\mathbf{W}}(\mathbf{X}) = \sum_{p \in [P]} \phi\left(\left\langle \mathbf{w}_1, \mathbf{x}^{(p)} \right\rangle\right) - \sum_{p \in [P]} \phi\left(\left\langle \mathbf{w}_{-1}, \mathbf{x}^{(p)} \right\rangle\right)$$

If $f_{\mathbf{W}}(\mathbf{X}) \geq 0$, predict $y = +1$, and vice versa.

# Prediction Model

$$f_{\mathbf{W}}(\mathbf{X}) = \sum_{p\in[P]} \phi\left(\left\langle \mathbf{w}_1, \mathbf{x}^{(p)} \right\rangle\right) - \sum_{p\in[P]} \phi\left(\left\langle \mathbf{w}_{-1}, \mathbf{x}^{(p)} \right\rangle\right)$$

If $f_{\mathbf{W}}(\mathbf{X}) \geq 0$, predict $y = +1$, and vice versa.

The activation function $\phi$ is a smoothed leaky ReLU activation, defined as

$$\phi(z) = \begin{cases} z - \frac{(1-\beta)r}{2} & z \geq r \\ \frac{1-\beta}{2r} z^2 + \beta z & 0 \leq z \leq r \\ \beta z & z \leq 0 \end{cases},$$

where $0 < \beta \leq 1$ and $r > 0$.

# Prediction Model

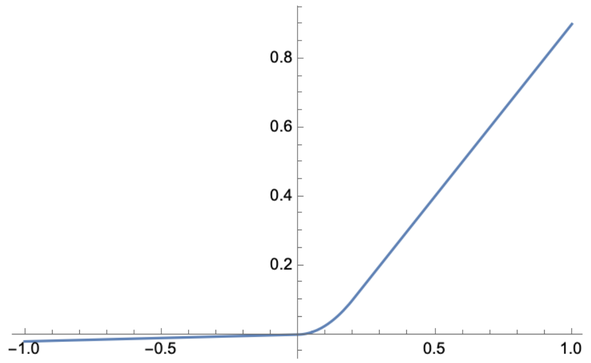
$$f_{\mathbf{W}}(\mathbf{X}) = \sum_{p\in[P]} \phi\left(\left\langle \mathbf{w}_1, \mathbf{x}^{(p)} \right\rangle\right) - \sum_{p\in[P]} \phi\left(\left\langle \mathbf{w}_{-1}, \mathbf{x}^{(p)} \right\rangle\right)$$

If $f_{\mathbf{W}}(\mathbf{X}) \geq 0$, predict $y = +1$, and vice versa.

The activation function $\phi$ is a smoothed leaky ReLU activation, defined as

$$\phi(z) = \begin{cases} z - \dfrac{(1-\beta)r}{2} & z \geq r \\ \dfrac{1-\beta}{2r} z^2 + \beta z & 0 \leq z \leq r \\ \beta z & z \leq 0 \end{cases},$$

where $0 < \beta \leq 1$ and $r > 0$.




Graph of $\phi$ for $\beta = 0.02,\ r = 0.2$

# Assumptions…

# Assumptions…

**Key** to the proofs + main **limitation** of this data distribution:

# Assumptions…

**Key** to the proofs + main **limitation** of this data distribution:

We focus on **high-dimensional** patch vectors (i.e., require large enough $d$) to use **near-orthogonality** of isotropic Gaussian vectors.

# Assumptions…

**Key** to the proofs + main **limitation** of this data distribution:

We focus on **high-dimensional** patch vectors (i.e., require large enough $d$) to use **near-orthogonality** of isotropic Gaussian vectors.

A set of assumptions on the problem parameters needs to be placed; example configuration of parameters that works is:

$$P = 8, C = 2, n = \Theta\left(d^{0.4}\right), \alpha = \Theta\left(d^{-0.02}\right), \beta = \frac{1}{\text{polylog}(d)}, \sigma_0 = \Theta(d^{-0.2}), r = \Theta(d^{-0.2}),$$
$$\sigma_{\text{d}} = \Theta\left(d^{-0.305}\right), \sigma_{\text{b}} = \Theta\left(d^{-0.375}\right), \rho_R = \Theta\left(d^{-0.1}\right), \rho_E = \Theta\left(d^{-0.195}\right), \eta = \Theta(d^{-1}).$$

# Assumptions…

**Assumption B.1.** The following conditions hold.

A1. (The number of patches) $P = \Theta(1)$ and $P \geq 8$.

A2. (Overparameterized regime): $n = o\left(\alpha\beta\sigma_{\mathrm{d}}^{-1}\sigma_{\mathrm{b}}d^{\frac{1}{2}}/\mathrm{polylog}(d)\right)$.

A3. (Sufficient feature data): For all $k \in [K]$, $\rho_k n = \omega\left(n^{\frac{1}{2}}\log d\right)$.

A4. (Common feature vs dominant noise): For all $k \in \mathcal{K}_C$, $\rho_k = \Theta(1)$ and $\sigma_{\mathrm{d}}^2 d = o(\beta n)$.

A5. (Rare feature vs noise): For all $k \in \mathcal{K}_R$, $\rho_k = \Theta(\rho_R)$ with $\rho_R n = o\left(\alpha^2\sigma_{\mathrm{d}}^2 d/\mathrm{polylog}(d)\right)$ and $\sigma_{\mathrm{b}}^2 d = o(\beta\rho_R n)$.

A6. (Extremely rare feature vs background noise) For all $k \in \mathcal{K}_E, \rho_k = \Theta(\rho_E)$ with $\rho_E n = o\left(\alpha^2\sigma_{\mathrm{b}}^2 d/\mathrm{polylog}(d)\right)$.

A7. (Strength of feature noise) $\alpha = o\left(n^{-1}\beta\sigma_{\mathrm{d}}^2 d/\mathrm{polylog}(d)\right)$.

A8. $\sigma_0\sigma_{\mathrm{d}}^2 d, r = o\left(\alpha/\mathrm{polylog}(d)\right), \eta = o\left(r\sigma_{\mathrm{d}}^{-2}d^{-1}/\mathrm{polylog}(d)\right)$

# Training Procedure 1: ERM



**Training Data:** $\{(\mathbf{X}_i, y_i)\}_{i \in [n]} \overset{\text{i.i.d}}{\sim} \mathscr{D}$

# Training Procedure 1: ERM



**Training Data:** $\{(\mathbf{X}_i, y_i)\}_{i\in[n]} \overset{\text{i.i.d}}{\sim} \mathscr{D}$

We define **ERM loss** as

$$\mathscr{L}_{\text{ERM}}(\mathbf{W}) := \frac{1}{n}\sum_{i\in[n]} \ell(y_i f_{\mathbf{W}}(\mathbf{X}_i)),$$

# Training Procedure 1: ERM



**Training Data:** $\{(\mathbf{X}_i, y_i)\}_{i\in[n]} \overset{\text{i.i.d}}{\sim} \mathscr{D}$

We define **ERM loss** as

$$\mathscr{L}_{\text{ERM}}(\mathbf{W}) := \frac{1}{n}\sum_{i\in[n]} \ell(y_i f_{\mathbf{W}}(\mathbf{X}_i)),$$

where $\ell(\,\cdot\,)$ is the logistic loss $\ell(z) = \log(1+\exp(-z))$.

# Training Procedure 1: ERM

**Training Data:** $\{(\mathbf{X}_i, y_i)\}_{i\in[n]} \overset{\text{i.i.d}}{\sim} \mathscr{D}$

We define **ERM loss** as

$$\mathscr{L}_{\text{ERM}}(\mathbf{W}) := \frac{1}{n}\sum_{i\in[n]} \ell(y_i f_{\mathbf{W}}(\mathbf{X}_i)),$$

where $\ell(\,\cdot\,)$ is the logistic loss $\ell(z) = \log(1+\exp(-z))$.

We consider GD on ERM loss $\mathscr{L}_{\text{ERM}}(\mathbf{W})$ with learning rate $\eta$.

# Training Procedure 2: Cutout

**Augmented Data**: For each $i \in [n]$ and $\mathscr{C} \in \binom{[P]}{C}$



# Training Procedure 2: Cutout

**Augmented Data**: For each $i \in [n]$ and $\mathscr{C} \in \binom{[P]}{C}$



$$\mathbf{X}_{i,\mathscr{C}} = (\mathbf{x}_{i,\mathscr{C}}^{(1)}, \ldots, \mathbf{x}_{i,\mathscr{C}}^{(P)}) \quad \text{where} \quad \mathbf{x}_{i,\mathscr{C}}^{(p)} = \begin{cases} \mathbf{x}_i^{(p)} & \text{if } p \notin \mathscr{C} \\ \mathbf{0} & \text{otherwise} \end{cases}.$$

# Training Procedure 2: Cutout

**Augmented Data**: For each $i \in [n]$ and $\mathscr{C} \in \binom{[P]}{C}$



$$\mathbf{X}_{i,\mathscr{C}} = (\mathbf{x}_{i,\mathscr{C}}^{(1)}, \ldots, \mathbf{x}_{i,\mathscr{C}}^{(P)}) \quad \text{where} \quad \mathbf{x}_{i,\mathscr{C}}^{(p)} = \begin{cases} \mathbf{x}_i^{(p)} & \text{if } p \notin \mathscr{C} \\ \mathbf{0} & \text{otherwise} \end{cases}.$$

We define **Cutout loss** as

$$\mathscr{L}_{\text{Cutout}}(\mathbf{W}) := \frac{1}{n} \sum_{i \in [n]} \mathbb{E}_{\mathscr{C} \sim \mathscr{D}_{\mathscr{C}}} \ell(y_i f_{\mathbf{W}}(\mathbf{X}_{i,\mathscr{C}})) \,.$$

**We fix** $1 \leq C < P/2$**.** $\mathscr{D}_{\mathscr{C}}$ **is a uniform distribution on** $\binom{[P]}{C}$**.**

# Training Procedure 2: Cutout

**Augmented Data**: For each $i \in [n]$ and $\mathscr{C} \in \binom{[P]}{C}$



$$\mathbf{X}_{i,\mathscr{C}} = (\mathbf{x}_{i,\mathscr{C}}^{(1)}, \ldots, \mathbf{x}_{i,\mathscr{C}}^{(P)}) \quad \text{where} \quad \mathbf{x}_{i,\mathscr{C}}^{(p)} = \begin{cases} \mathbf{x}_i^{(p)} & \text{if } p \notin \mathscr{C} \\ \mathbf{0} & \text{otherwise} \end{cases}.$$

We define **Cutout loss** as

$$\mathscr{L}_{\text{Cutout}}(\mathbf{W}) := \frac{1}{n} \sum_{i \in [n]} \mathbb{E}_{\mathscr{C} \sim \mathscr{D}_{\mathscr{C}}} \ell(y_i f_{\mathbf{W}}(\mathbf{X}_{i,\mathscr{C}})) \,.$$

We consider GD on Cutout loss $\mathscr{L}_{\text{Cutout}}(\mathbf{W})$ with learning rate $\eta$.

**We fix $1 \le C < P/2$. $\mathscr{D}_{\mathscr{C}}$ is a uniform distribution on $\binom{[P]}{C}$.**

# Training Procedure 3: CutMix

**Augmented Data**: For each $i, j \in [n]$ and $\mathcal{S} \subset [P]$.



# Training Procedure 3: CutMix




**Augmented Data**: For each $i, j \in [n]$ and $\mathcal{S} \subset [P]$.

$$\mathbf{X}_{i,j,\mathcal{S}} = (\mathbf{x}^{(1)}_{i,j,\mathcal{S}}, \ldots, \mathbf{x}^{(P)}_{i,j,\mathcal{S}}) \quad \text{where} \quad \mathbf{x}^{(p)}_{i,j,\mathcal{S}} = \begin{cases} \mathbf{x}^{(p)}_i & \text{if } p \in \mathcal{S} \\ \mathbf{x}^{(p)}_j & \text{otherwise} \end{cases}.$$

# Training Procedure 3: CutMix




**Augmented Data**: For each $i, j \in [n]$ and $\mathcal{S} \subset [P]$.

$$\mathbf{X}_{i,j,\mathcal{S}} = (\mathbf{x}^{(1)}_{i,j,\mathcal{S}}, \ldots, \mathbf{x}^{(P)}_{i,j,\mathcal{S}}) \quad \text{where} \quad \mathbf{x}^{(p)}_{i,j,\mathcal{S}} = \begin{cases} \mathbf{x}^{(p)}_i & \text{if } p \in \mathcal{S} \\ \mathbf{x}^{(p)}_j & \text{otherwise} \end{cases}.$$

We define **CutMix loss** as

$$\mathscr{L}_{\text{CutMix}}(\mathbf{W}) := \frac{1}{n^2} \sum_{i,j \in [n]} \mathbb{E}_{\mathcal{S} \sim \mathscr{D}_{\mathcal{S}}} \left[ \frac{|\mathcal{S}|}{P} \ell(y_i f_{\mathbf{W}}(\mathbf{X}_{i,j,\mathcal{S}})) + \left(1 - \frac{|\mathcal{S}|}{P}\right) \ell(y_j f_{\mathbf{W}}(\mathbf{X}_{i,,j,\mathcal{S}})) \right].$$

**$\mathscr{D}_{\mathcal{S}}$ is a distribution such that: 1. uniformly choose size $s \in \{0,1,\ldots,P\}$ and 2. uniformly choose $\mathcal{S}$ from $\binom{[P]}{s}$.**

# Training Procedure 3: CutMix




**Augmented Data**: For each $i, j \in [n]$ and $\mathcal{S} \subset [P]$.

$$\mathbf{X}_{i,j,\mathcal{S}} = (\mathbf{x}^{(1)}_{i,j,\mathcal{S}}, \ldots, \mathbf{x}^{(P)}_{i,j,\mathcal{S}}) \quad \text{where} \quad \mathbf{x}^{(p)}_{i,j,\mathcal{S}} = \begin{cases} \mathbf{x}^{(p)}_i & \text{if } p \in \mathcal{S} \\ \mathbf{x}^{(p)}_j & \text{otherwise} \end{cases}.$$

We define **CutMix loss** as

$$\mathscr{L}_{\text{CutMix}}(\mathbf{W}) := \frac{1}{n^2} \sum_{i,j \in [n]} \mathbb{E}_{\mathcal{S} \sim \mathscr{D}_{\mathcal{S}}} \left[ \frac{|\mathcal{S}|}{P} \ell(y_i f_{\mathbf{W}}(\mathbf{X}_{i,j,\mathcal{S}})) + \left(1 - \frac{|\mathcal{S}|}{P}\right) \ell(y_j f_{\mathbf{W}}(\mathbf{X}_{i,,j,\mathcal{S}})) \right].$$

We consider GD on CutMix loss $\mathscr{L}_{\text{CutMix}}(\mathbf{W})$ with learning rate $\eta$.

**$\mathscr{D}_{\mathcal{S}}$ is a distribution such that: 1. uniformly choose size $s \in \{0,1,\ldots,P\}$ and 2. uniformly choose $\mathcal{S}$ from $\binom{[P]}{s}$.**

# Main Results - ERM

**Theorem 3.1 (Vanilla Training)**

Let $\mathbf{W}^{(t)}$ be iterates of ERM training. Then with high probability, there exists $T_{\mathrm{ERM}}$ such that any $T \in [T_{\mathrm{ERM}}, T^*]$ satisfies the following:

**Here, $T^*$ is any large enough (polynomial in $d$) admissible training iterations**

# Main Results - ERM

**Theorem 3.1 (Vanilla Training)**

Let $\mathbf{W}^{(t)}$ be iterates of ERM training. Then with high probability, there exists $T_{\mathrm{ERM}}$ such that any $T \in [T_{\mathrm{ERM}}, T^*]$ satisfies the following:

1. **(Perfectly fits training set)**: For all $i \in [n]$, $y_i f_{\mathbf{W}^{(T)}}(\mathbf{X}_i) > 0$.

**Here, $T^*$ is any large enough (polynomial in $d$) admissible training iterations**

# Main Results - ERM

**Theorem 3.1 (Vanilla Training)**

Let $\mathbf{W}^{(t)}$ be iterates of ERM training. Then with high probability, there exists $T_{\mathrm{ERM}}$ such that any $T \in [T_{\mathrm{ERM}}, T^*]$ satisfies the following:

1. **(Perfectly fits training set)**: For all $i \in [n]$, $y_i f_{\mathbf{W}^{(T)}}(\mathbf{X}_i) > 0$.

2. **(Random guess on new data with rare and extremely rare features)**:

$$\mathbb{P}_{(\mathbf{X},y)\sim\mathcal{D}}[y f_{\mathbf{W}^{(T)}}(\mathbf{X}) > 0] \approx 1 - \frac{1}{2} \sum_{k \in \mathcal{K}_R \cup \mathcal{K}_E} \rho_k$$

**Here, $T^*$ is any large enough (polynomial in $d$) admissible training iterations**

# Main Results - Cutout

**Theorem 3.2 (Cutout Training)**

Let $\mathbf{W}^{(t)}$ be iterates of Cutout training. Then with high probability, there exists $T_{\text{Cutout}}$ such that any $T \in [T_{\text{Cutout}}, T^*]$ satisfies the following:

# Main Results - Cutout

**Theorem 3.2 (Cutout Training)**

Let $\mathbf{W}^{(t)}$ be iterates of Cutout training. Then with high probability, there exists $T_{\text{Cutout}}$ such that any $T \in [T_{\text{Cutout}}, T^*]$ satisfies the following:

1. **(Perfectly fits augmented data)**: For all $i \in [n]$ & $\mathscr{C} \in \binom{[P]}{C}$, $y_i f_{\mathbf{W}^{(T)}}(\mathbf{X}_{i,\mathscr{C}}) > 0$.

# Main Results - Cutout

**Theorem 3.2 (Cutout Training)**

Let $\mathbf{W}^{(t)}$ be iterates of Cutout training. Then with high probability, there exists $T_{\text{Cutout}}$ such that any $T \in [T_{\text{Cutout}}, T^*]$ satisfies the following:

1. **(Perfectly fits augmented data)**: For all $i \in [n]$ & $\mathscr{C} \in \binom{[P]}{C}$, $y_i f_{\mathbf{W}^{(T)}}(\mathbf{X}_{i,\mathscr{C}}) > 0$.

2. **(Perfectly fits original training data)**: For all $i \in [n]$, $y_i f_{\mathbf{W}^{(T)}}(\mathbf{X}_i) > 0$.

# Main Results - Cutout

**Theorem 3.2 (Cutout Training)**

Let $\mathbf{W}^{(t)}$ be iterates of Cutout training. Then with high probability, there exists $T_{\text{Cutout}}$ such that any $T \in [T_{\text{Cutout}}, T^*]$ satisfies the following:

1. **(Perfectly fits augmented data)**: For all $i \in [n]$ & $\mathscr{C} \in \binom{[P]}{C}$, $y_i f_{\mathbf{W}^{(T)}}(\mathbf{X}_{i,\mathscr{C}}) > 0$.

2. **(Perfectly fits original training data)**: For all $i \in [n]$, $y_i f_{\mathbf{W}^{(T)}}(\mathbf{X}_i) > 0$.

3. **(Random guess on new data with extremely rare features)**:

$$\mathbb{P}_{(\mathbf{X},y)\sim\mathscr{D}}[y f_{\mathbf{W}^{(T)}}(\mathbf{X}) > 0] \approx 1 - \frac{1}{2}\sum_{k\in\mathscr{K}_E} \rho_k$$

# Main Results - CutMix

**Theorem 3.3 (CutMix Training)**

Let $\mathbf{W}^{(t)}$ be iterates of CutMix training. Then with high probability, there exists some $T \in [0, T^*]$ that satisfies the following:

# Main Results - CutMix

**Theorem 3.3 (CutMix Training)**

Let $\mathbf{W}^{(t)}$ be iterates of CutMix training. Then with high probability, there exists some $T \in [0, T^*]$ that satisfies the following:

1. **(Near Convergence)**: $\left\| \nabla_{\mathbf{W}} \mathscr{L}_{\text{CutMix}} \left( \mathbf{W}^{(T)} \right) \right\| \approx 0$

# Main Results - CutMix

**Theorem 3.3 (CutMix Training)**

Let $\mathbf{W}^{(t)}$ be iterates of CutMix training. Then with high probability, there exists some $T \in [0, T^*]$ that satisfies the following:

1. **(Near Convergence)**: $\left\| \nabla_{\mathbf{W}} \mathscr{L}_{\text{CutMix}} \left( \mathbf{W}^{(T)} \right) \right\| \approx 0$

2. **(Perfectly fits original training data)**: For all $i \in [n]$, $y_i f_{\mathbf{W}^{(T)}}(\mathbf{X}_i) > 0$.

# Main Results - CutMix

**Theorem 3.3 (CutMix Training)**

Let $\mathbf{W}^{(t)}$ be iterates of CutMix training. Then with high probability, there exists some $T \in [0, T^*]$ that satisfies the following:

1. **(Near Convergence)**: $\left\| \nabla_{\mathbf{W}} \mathscr{L}_{\text{CutMix}} \left( \mathbf{W}^{(T)} \right) \right\| \approx 0$

2. **(Perfectly fits original training data)**: For all $i \in [n]$, $y_i f_{\mathbf{W}^{(T)}}(\mathbf{X}_i) > 0$.

3. **(Almost perfectly classifies test data)**: $\mathbb{P}_{(\mathbf{X},y)\sim\mathscr{D}}[y f_{\mathbf{W}^{(T)}}(\mathbf{X}) > 0] \approx 1$.

# Main Results - Summary

| Training Method \ Rarity | Common | Rare | Extremely Rare |
|---|---|---|---|
| ERM (Vanilla Training) | ✓ | ✗ | ✗ |
| Cutout | ✓ | ✓ | ✗ |
| Cutmix | ✓ | ✓ | ✓ |

# Proof Overview



# Proof Overview

There are two ways for the model to “fit” training data



# Proof Overview

There are two ways for the model to “fit” training data



1. It could extract the **feature vector** $\mathbf{v}_{s,k}$

# Proof Overview

There are two ways for the model to “fit” training data



1. It could extract the **feature vector** $\mathbf{v}_{s,k}$
    - The model will **generalize** to test examples having the same feature $\mathbf{v}_{s,k}$

# Proof Overview

There are two ways for the model to “fit” training data



1. It could extract the **feature vector** $\mathbf{v}_{s,k}$
    - The model will **generalize** to test examples having the same feature $\mathbf{v}_{s,k}$

**“Benign overfitting”**

# Proof Overview

There are two ways for the model to “fit” training data



1. It could extract the **feature vector** $\mathbf{v}_{s,k}$
   - The model will **generalize** to test examples having the same feature $\mathbf{v}_{s,k}$

     **“Benign overfitting”**
2. It could memorize the **noise patches** $\xi_i^{(p)}$ in the data point

# Proof Overview

There are two ways for the model to “fit” training data



1. It could extract the **feature vector** $\mathbf{v}_{s,k}$
   - The model will **generalize** to test examples having the same feature $\mathbf{v}_{s,k}$

     **“Benign overfitting”**
2. It could memorize the **noise patches** $\xi_i^{(p)}$ in the data point
   - Feature vector will ***not*** be learned because loss gradient already $\approx 0$

# Proof Overview

There are two ways for the model to “fit” training data



1. It could extract the **feature vector** $\mathbf{v}_{s,k}$
   - The model will **generalize** to test examples having the same feature $\mathbf{v}_{s,k}$

     **“Benign overfitting”**
2. It could memorize the **noise patches** $\xi_i^{(p)}$ in the data point
   - Feature vector will ***not*** be learned because loss gradient already $\approx 0$
   - Memorized noises are (near-)orthogonal to patches in test examples

# Proof Overview

There are two ways for the model to “fit” training data



1. It could extract the **feature vector** $\mathbf{v}_{s,k}$
   - The model will **generalize** to test examples having the same feature $\mathbf{v}_{s,k}$

     **“Benign overfitting”**
2. It could memorize the **noise patches** $\xi_i^{(p)}$ in the data point
   - Feature vector will ***not*** be learned because loss gradient already $\approx 0$
   - Memorized noises are (near-)orthogonal to patches in test examples
   - The model will make **random guesses** on test examples having features that it **failed** to learn

# Proof Overview

There are two ways for the model to “fit” training data



1. It could extract the **feature vector** $\mathbf{v}_{s,k}$
   - The model will **generalize** to test examples having the same feature $\mathbf{v}_{s,k}$

     **“Benign overfitting”**
2. It could memorize the **noise patches** $\xi_i^{(p)}$ in the data point
   - Feature vector will ***not*** be learned because loss gradient already $\approx 0$
   - Memorized noises are (near-)orthogonal to patches in test examples
   - The model will make **random guesses** on test examples having features that it **failed** to learn

     **“Harmful overfitting”**

# Proof Overview - ERM and Cutout

Consider ERM with the case $\alpha = 0$.

$$y_i \left\langle \mathbf{w}_s^{(t+1)}, \mathbf{v}_{s,k} \right\rangle = y_i \left\langle \mathbf{w}_s^{(t)}, \mathbf{v}_{s,k} \right\rangle - \frac{\eta}{n} \sum_{j \in \mathscr{V}_{s,k}} \ell'(y_j f_{\mathbf{W}^{(t)}}(\mathbf{X}_j)) \phi' \left( \left\langle \mathbf{w}_s^{(t)}, \mathbf{v}_{s,k} \right\rangle \right)$$

$$y_i \left\langle \mathbf{w}_s^{(t+1)}, \xi_i^{(p)} \right\rangle \approx y_i \left\langle \mathbf{w}_s^{(t)}, \xi_i^{(p)} \right\rangle - \frac{\eta}{n} \ell'(y_i f_{\mathbf{W}^{(t)}}(\mathbf{X}_i)) \phi' \left( \left\langle \mathbf{w}_s^{(t)}, \xi_i^{(p)} \right\rangle \right) \left\| \xi_i^{(p)} \right\|^2$$

**Here, $\mathscr{V}_{s,k}$ denotes the set of data with feature vector $\mathbf{v}_{s,k}$**

# Proof Overview - ERM and Cutout

Consider ERM with the case $\alpha = 0$.

$$y_i \left\langle \mathbf{w}_s^{(t+1)}, \mathbf{v}_{s,k} \right\rangle = y_i \left\langle \mathbf{w}_s^{(t)}, \mathbf{v}_{s,k} \right\rangle - \frac{\eta}{n} \sum_{j \in \mathscr{V}_{s,k}} \ell'(y_j f_{\mathbf{W}^{(t)}}(\mathbf{X}_j)) \phi' \left( \left\langle \mathbf{w}_s^{(t)}, \mathbf{v}_{s,k} \right\rangle \right)$$

The main factor for the speed of learning features and noises are number of feature occurrence and the strength of noises

$$y_i \left\langle \mathbf{w}_s^{(t+1)}, \xi_i^{(p)} \right\rangle \approx y_i \left\langle \mathbf{w}_s^{(t)}, \xi_i^{(p)} \right\rangle - \frac{\eta}{n} \ell'(y_i f_{\mathbf{W}^{(t)}}(\mathbf{X}_i)) \phi' \left( \left\langle \mathbf{w}_s^{(t)}, \xi_i^{(p)} \right\rangle \right) \left\| \xi_i^{(p)} \right\|^2$$

**Here, $\mathscr{V}_{s,k}$ denotes the set of data with feature vector $\mathbf{v}_{s,k}$**

# Proof Overview - ERM and Cutout

Consider ERM with the case $\alpha = 0$.

$$y_i \left\langle \mathbf{w}_s^{(t+1)}, \mathbf{v}_{s,k} \right\rangle = y_i \left\langle \mathbf{w}_s^{(t)}, \mathbf{v}_{s,k} \right\rangle - \frac{\eta}{n} \sum_{j \in \mathscr{V}_{s,k}} \ell'(y_j f_{\mathbf{W}^{(t)}}(\mathbf{X}_j)) \phi' \left( \left\langle \mathbf{w}_s^{(t)}, \mathbf{v}_{s,k} \right\rangle \right)$$

The main factor for the speed of learning features and noises are number of feature occurrence and the strength of noises

$$y_i \left\langle \mathbf{w}_s^{(t+1)}, \xi_i^{(p)} \right\rangle \approx y_i \left\langle \mathbf{w}_s^{(t)}, \xi_i^{(p)} \right\rangle - \frac{\eta}{n} \ell'(y_i f_{\mathbf{W}^{(t)}}(\mathbf{X}_i)) \phi' \left( \left\langle \mathbf{w}_s^{(t)}, \xi_i^{(p)} \right\rangle \right) \left\| \xi_i^{(p)} \right\|^2$$

Common features ≫ dominant noise ≫ rare features≫ background noise≫ extremely rare feature

**Here, $\mathscr{V}_{s,k}$ denotes the set of data with feature vector $\mathbf{v}_{s,k}$**

# Proof Overview - ERM and Cutout

Consider ERM with the case $\alpha = 0$.

$$y_i \left\langle \mathbf{w}_s^{(t+1)}, \mathbf{v}_{s,k} \right\rangle = y_i \left\langle \mathbf{w}_s^{(t)}, \mathbf{v}_{s,k} \right\rangle - \frac{\eta}{n} \sum_{j \in \mathscr{V}_{s,k}} \ell'(y_j f_{\mathbf{W}^{(t)}}(\mathbf{X}_j)) \phi' \left( \left\langle \mathbf{w}_s^{(t)}, \mathbf{v}_{s,k} \right\rangle \right)$$

The main factor for the speed of learning features and noises are number of feature occurrence and the strength of noises

$$y_i \left\langle \mathbf{w}_s^{(t+1)}, \xi_i^{(p)} \right\rangle \approx y_i \left\langle \mathbf{w}_s^{(t)}, \xi_i^{(p)} \right\rangle - \frac{\eta}{n} \ell'(y_i f_{\mathbf{W}^{(t)}}(\mathbf{X}_i)) \phi' \left( \left\langle \mathbf{w}_s^{(t)}, \xi_i^{(p)} \right\rangle \right) \left\| \xi_i^{(p)} \right\|^2$$

ERM 

Common features $\gg$ dominant noise $\gg$ rare features $\gg$ background noise $\gg$ extremely rare feature

**Here, $\mathscr{V}_{s,k}$ denotes the set of data with feature vector $\mathbf{v}_{s,k}$**

# Proof Overview - ERM and Cutout

Consider ERM with the case $\alpha = 0$.

$$y_i \left\langle \mathbf{w}_s^{(t+1)}, \mathbf{v}_{s,k} \right\rangle = y_i \left\langle \mathbf{w}_s^{(t)}, \mathbf{v}_{s,k} \right\rangle - \frac{\eta}{n} \sum_{j \in \mathscr{V}_{s,k}} \ell'(y_j f_{\mathbf{W}^{(t)}}(\mathbf{X}_j)) \phi' \left( \left\langle \mathbf{w}_s^{(t)}, \mathbf{v}_{s,k} \right\rangle \right)$$

The main factor for the speed of learning features and noises are number of feature occurrence and the strength of noises

$$y_i \left\langle \mathbf{w}_s^{(t+1)}, \xi_i^{(p)} \right\rangle \approx y_i \left\langle \mathbf{w}_s^{(t)}, \xi_i^{(p)} \right\rangle - \frac{\eta}{n} \ell'(y_i f_{\mathbf{W}^{(t)}}(\mathbf{X}_i)) \phi' \left( \left\langle \mathbf{w}_s^{(t)}, \xi_i^{(p)} \right\rangle \right) \left\| \xi_i^{(p)} \right\|^2$$



Cutout

Common features ≫ dominant noise ≫ rare features≫ background noise≫ extremely rare feature

**Here, $\mathscr{V}_{s,k}$ denotes the set of data with feature vector $\mathbf{v}_{s,k}$**

# Proof Overview - CutMix

For CutMix, the evolution of inner products is no longer **monotone**!
Instead, we characterize the global minimum of CutMix loss.

# Proof Overview - CutMix

For CutMix, the evolution of inner products is no longer **monotone**!
Instead, we characterize the global minimum of CutMix loss.

Surprisingly, at the global minimum the model has **"even" outputs** for **all patches** (features and noise) regardless of the frequency and strength.

# Proof Overview - CutMix

For CutMix, the evolution of inner products is no longer **monotone**!
Instead, we characterize the global minimum of CutMix loss.

Surprisingly, at the global minimum the model has **"even" outputs** for **all patches** (features and noise) regardless of the frequency and strength.

$$\phi\left(\left\langle \mathbf{w}_s, \mathbf{x}_i^{(p)} \right\rangle\right) - \phi\left(\left\langle \mathbf{w}_{-s}, \mathbf{x}_i^{(p)} \right\rangle\right) = C_s$$

# Proof Overview - CutMix

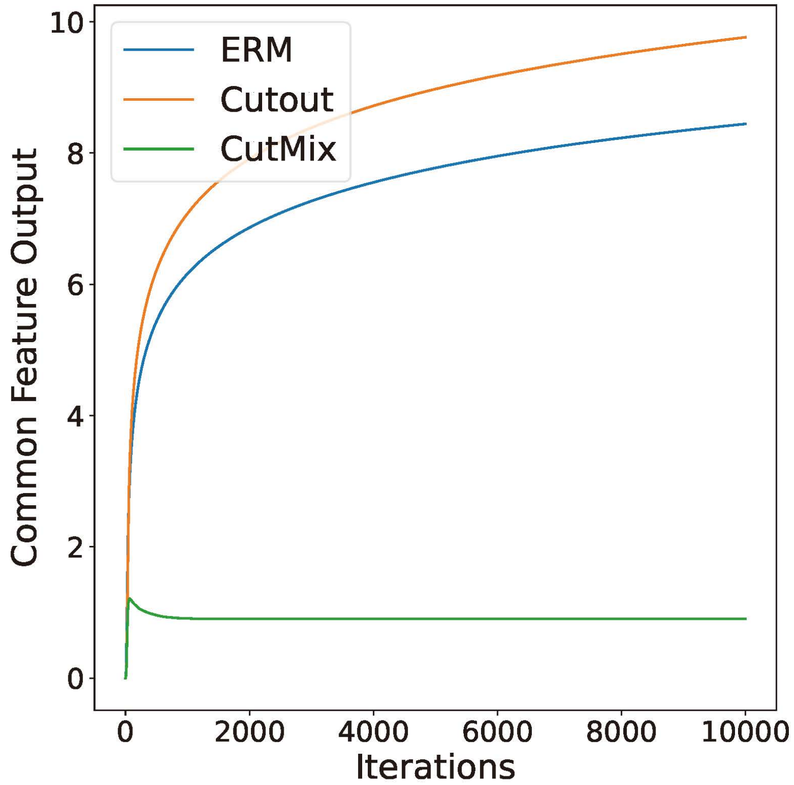
For CutMix, the evolution of inner products is no longer **monotone**!
Instead, we characterize the global minimum of CutMix loss.

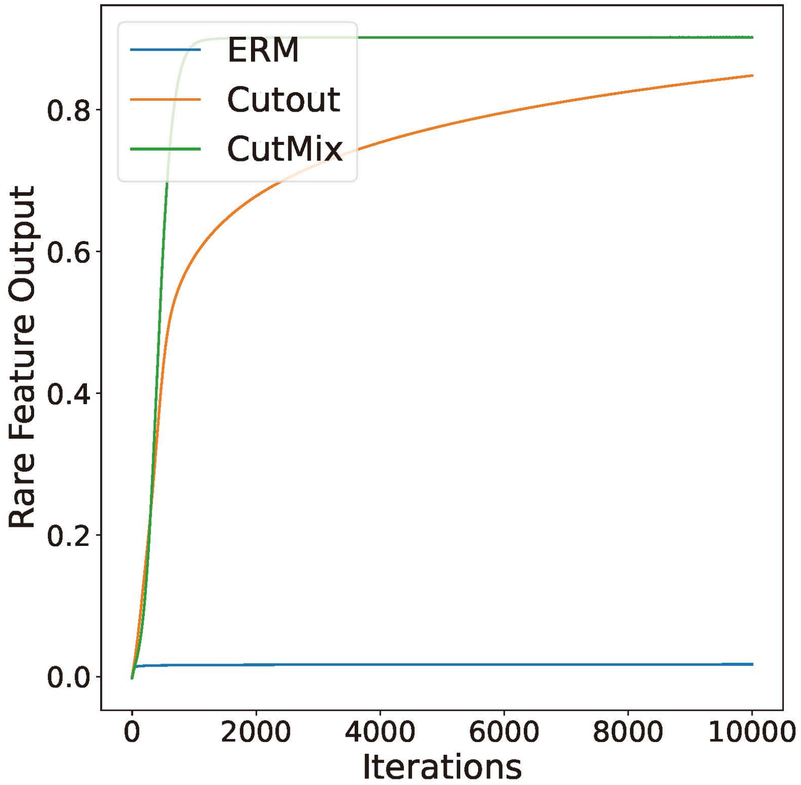
Surprisingly, at the global minimum the model has **"even" outputs** for **all patches** (features and noise) regardless of the frequency and strength.

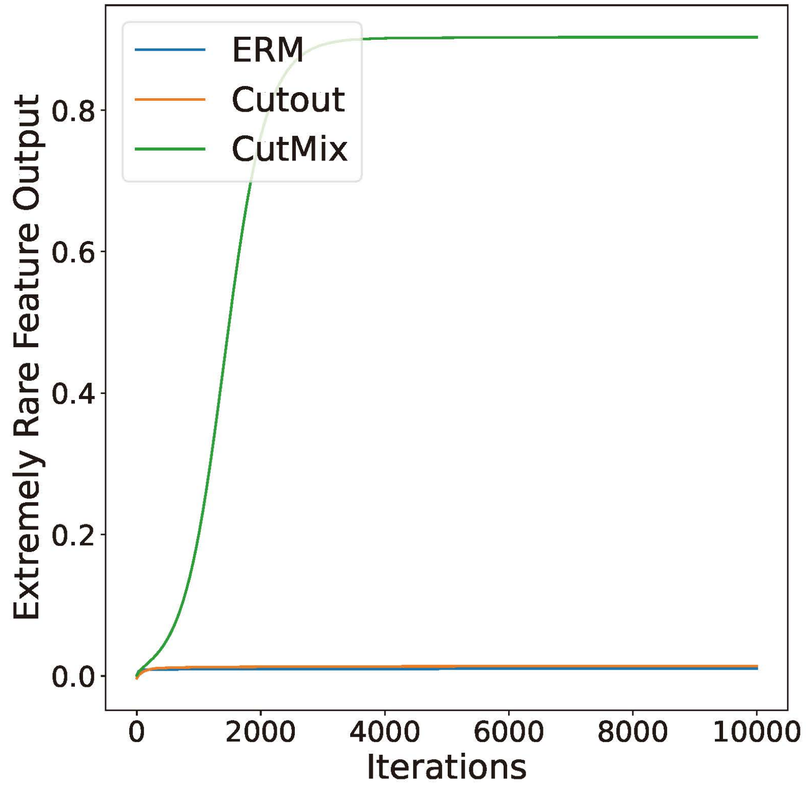
$$\phi\left(\left\langle \mathbf{w}_s, \mathbf{x}_i^{(p)} \right\rangle\right) - \phi\left(\left\langle \mathbf{w}_{-s}, \mathbf{x}_i^{(p)} \right\rangle\right) = C_s$$

We prove the convergence to a global minimum based on descent lemma.

# Experimental Results in Our Setting



# Summary

- Cutout enables learning rare features that cannot be learned through vanilla training, by mitigating noise memorization.

# Summary

- Cutout enables learning rare features that cannot be learned through vanilla training, by mitigating noise memorization.

- CutMix can learn all rare features that Cutout cannot learn,
  by learning all features and noise vectors evenly regardless of the rarity and strength.

# Summary

- Cutout enables learning rare features that cannot be learned through vanilla training, by mitigating noise memorization.
- CutMix can learn all rare features that Cutout cannot learn,
  by learning all features and noise vectors evenly regardless of the rarity and strength.
- Future direction: Remove the high-dimensional requirement $d$? Extension to other techniques?

# Thank you! Questions?

Oh & Yun. Provable Benefit of Cutout and CutMix for Feature Learning. NeurIPS 2024 (Spotlight).

Junsoo Oh (junsoo.oh@kaist.ac.kr)
Chulhee Yun (chulhee.yun@kaist.ac.kr)

OptiML Optimization & Machine Learning Laboratory
KAIST AI
Kim Jaechul Graduate School