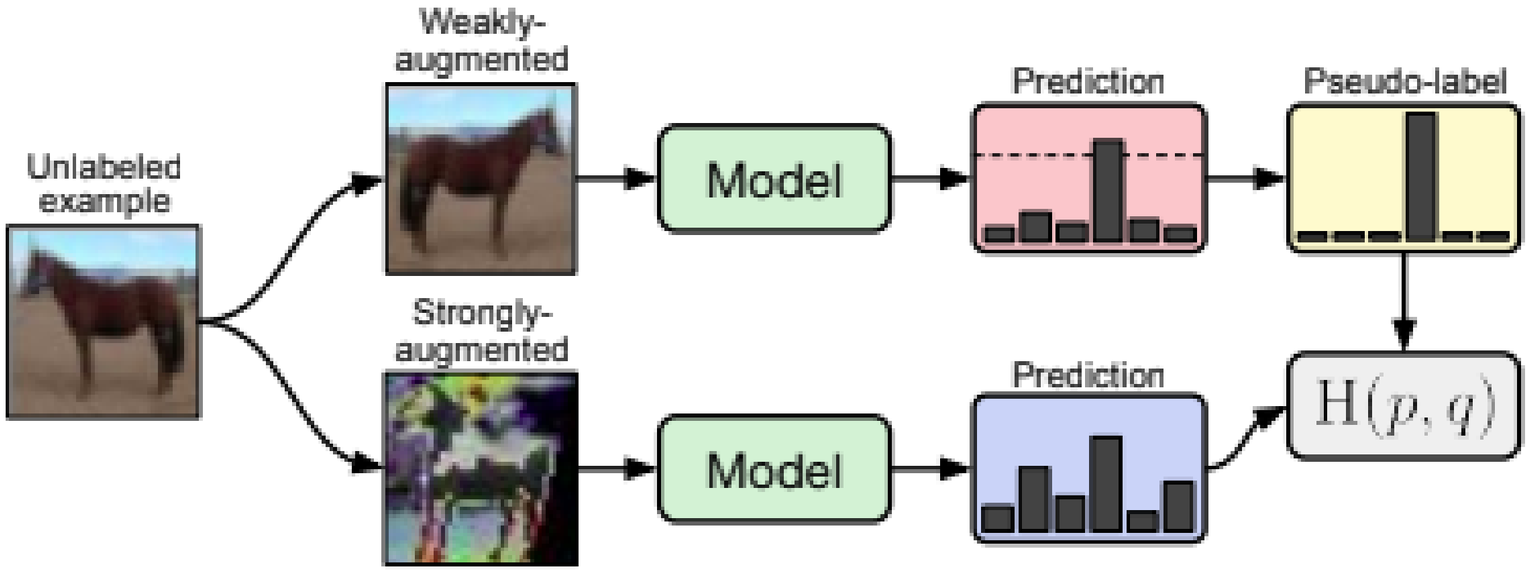
FixMatch - Tarefa 3

Implementação e apresentação

O que é o FixMatch



consistency regularization and pseudo-labeling

Weakly-augmented

usamos o mesmo que o paper

- Flip horizontal (50% de probabilidade)
- Translação (±12,5%) horizontal e vertical

```
weak_transform = transforms.Compose([
    transforms.RandomHorizontalFlip(p=0.5),
    transforms.RandomAffine(degrees=0, translate=(0.125, 0.125)),
    transforms.ToTensor(),
    transforms.Normalize(mean, std),
])
```

Strongly-augmented

usamos o RandAugument + Cutout
(Como no paper)

```python
class Cutout(object):
    def __init__(self, mask_size, p=1.0):
        self.mask_size = mask_size
        self.p = p

    def __call__(self, img):
        if random.random() > self.p:
            return img
        if not isinstance(img, torch.Tensor):
            img = transforms.functional.to_tensor(img)
        c, h, w = img.shape
        mask_size_half = self.mask_size // 2
        cx = random.randint(0, w - 1)
        cy = random.randint(0, h - 1)
        x1 = max(0, cx - mask_size_half)
        y1 = max(0, cy - mask_size_half)
        x2 = min(w, cx + mask_size_half)
        y2 = min(h, cy + mask_size_half)
        img[:, y1:y2, x1:x2] = 0.0
        return img
```

Transformation	Description	Parameter	Range
Autocontrast	Maximizes the image contrast by setting the darkest (lightest) pixel to black (white).		
Brightness	Adjusts the brightness of the image. $B = 0$ returns a black image, $B = 1$ returns the original image.	B	[0.05, 0.95]
Color	Adjusts the color balance of the image like in a TV. $C = 0$ returns a black & white image, $C = 1$ returns the original image.	C	[0.05, 0.95]
Contrast	Controls the contrast of the image. A $C = 0$ returns a gray image, $C = 1$ returns the original image.	C	[0.05, 0.95]
Equalize	Equalizes the image histogram.		
Identity	Returns the original image.		
Posterize	Reduces each pixel to B bits.	B	[4, 8]
Rotate	Rotates the image by θ degrees.	θ	[-30, 30]
Sharpness	Adjusts the sharpness of the image, where $S = 0$ returns a blurred image, and $S = 1$ returns the original image.	S	[0.05, 0.95]
Shear_x	Shears the image along the horizontal axis with rate R.	R	[-0.3, 0.3]
Shear_y	Shears the image along the vertical axis with rate R.	R	[-0.3, 0.3]
Solarize	Inverts all pixels above a threshold value of T.	T	[0, 1]
Translate_x	Translates the image horizontally by ($\lambda \times$image width) pixels.	λ	[-0.3, 0.3]
Translate_y	Translates the image vertically by ($\lambda \times$image height) pixels.	λ	[-0.3, 0.3]

```python
self.transforms = [
    ('autocontrast', lambda img, mag: transforms.functional.autocontrast(img)),
    ('brightness', lambda img, mag: transforms.functional.adjust_brightness(img, 0.05 + (0.95-0.05) * mag/10)),
    ('color', lambda img, mag: transforms.functional.adjust_saturation(img, 0.05 + (0.95-0.05) * mag/10)),
    ('contrast', lambda img, mag: transforms.functional.adjust_contrast(img, 0.05 + (0.95-0.05) * mag/10)),
    ('equalize', lambda img, mag: transforms.functional.equalize(img)),
    ('identity', lambda img, mag: img),
    ('posterize', lambda img, mag: transforms.functional.posterize(img, int(4 + (8-4) * mag/10))),
    ('rotate', lambda img, mag: transforms.functional.rotate(img, -30 + 60 * mag/10)),
    ('sharpness', lambda img, mag: transforms.functional.adjust_sharpness(img, 0.05 + (0.95-0.05) * mag/10)),
    ('shear_x', lambda img, mag: transforms.functional.affine(img, angle=0, translate=[0,0], scale=1, shear=[-30 + 60 * mag/10, 0])),
    ('shear_y', lambda img, mag: transforms.functional.affine(img, angle=0, translate=[0,0], scale=1, shear=[0, -30 + 60 * mag/10])),
    ('solarize', lambda img, mag: transforms.functional.solarize(img, int(255 * mag/10))),
    ('translate_x', lambda img, mag: transforms.functional.affine(img, angle=0, translate=[int(img.size[0] * (-0.3 + 0.6 * mag/10)), 0], scale=1, shear=[0,0])),
    ('translate_y', lambda img, mag: transforms.functional.affine(img, angle=0, translate=[0, int(img.size[1] * (-0.3 + 0.6 * mag/10))], scale=1, shear=[0,0])),
]
```

Modelo usado

Nossa implementação: ResNet-18

Paper: Wide ResNet-28-2

Treinamento

```
# Pseudo-labels: weak augment → confiança
self.model.eval()
with torch.no_grad():
    probs_w = F.softmax(self.model(uw), dim=1)

    # Distribution Alignment
    self.p_model = self.p_model * self.da_momentum + (1 - self.da_momentum) * probs_w.mean(dim=0)
    adjust = (self.p_target / (self.p_model + 1e-6))
    probs_w = probs_w * adjust
    probs_w = probs_w / probs_w.sum(dim=1, keepdim=True)

    max_probs, pseudo_labels = probs_w.max(dim=1)
    mask = (max_probs >= current_tau).float()

# MixMatch: concatena labeled + unlabeled strong
inputs = torch.cat([x_l, us], dim=0)
logits_all = self.model(inputs)
logits_l  = logits_all[:x_l.size(0)]
logits_us = logits_all[x_l.size(0):]

# Loss supervisionada
loss_sup = F.cross_entropy(logits_l, y_l)

# Loss não-supervisionada (apenas amostras confiantes)
loss_unsup_all = F.cross_entropy(logits_us, pseudo_labels, reduction='none')
```

$$\ell_s = \frac{1}{B}\sum_{b=1}^{B} \mathrm{H}(p_b, p_{\mathrm{m}}(y \mid \alpha(x_b))) \quad + \quad \lambda_u\, \ell_u = \frac{1}{\mu B}\sum_{b=1}^{\mu B} \mathbb{1}(\max(q_b) \geq \tau)\, \mathrm{H}(\hat{q}_b, p_{\mathrm{m}}(y \mid \mathcal{A}(u_b)))$$

```
# Loss supervisionada
loss_sup = F.cross_entropy(logits_l, y_l)

# Loss não-supervisionada (apenas amostras confiantes)
loss_unsup_all = F.cross_entropy(logits_us, pseudo_labels, reduction='none')
```

```
loss = loss_sup + self.lambda_u * loss_unsup
```

Otimizador

$(\lambda = 1, \eta = 0.03, \beta = 0.9, \pi = 0.25, \mu = 7, B = 64, K = 2^{20})$[4]

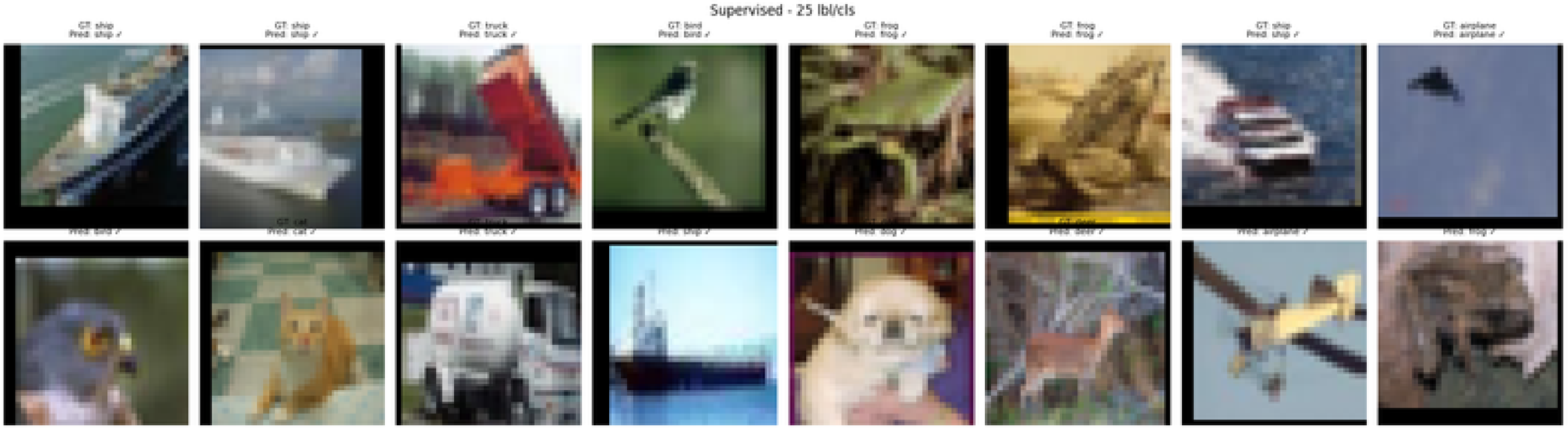
```python
# SGD com momentum=0.9, lr=0.03, weight_decay=0.0005, sem Nesterov
self.optimizer = torch.optim.SGD(
    self.model.parameters(),
    lr=lr,
    momentum=0.9,
    weight_decay=weight_decay,
    nesterov=False,
)

# Cosine decay: η·cos(7πk/16K)
K = num_epochs * len(labeled_loader)
self.scheduler = torch.optim.lr_scheduler.LambdaLR(
    self.optimizer,
    lr_lambda=lambda step: math.cos(7 * math.pi * step / (16 * K))
)
self.history = {"epoch": [], "loss": [], "acc": []}
```

Otimizador: SGD com momentum

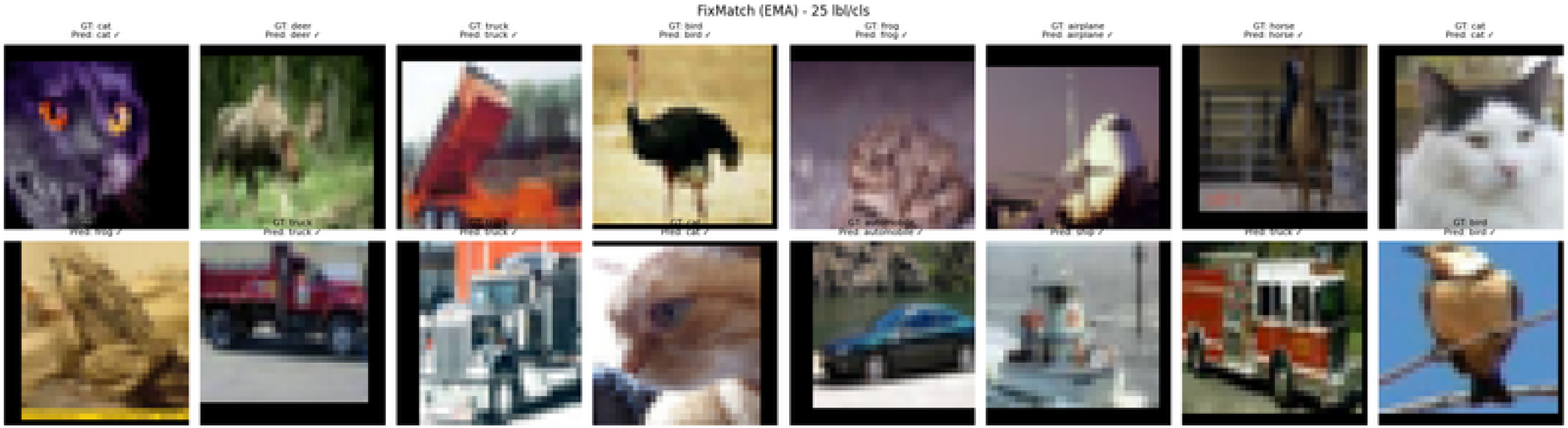
LR scheduler: $\eta \cos\left(\frac{7\pi k}{16K}\right)$

Supervised



- O modelo aprende apenas a partir dos 25 rótulos reais por classe, usando unicamente o sinal supervisionado.
- Cada imagem rotulada define diretamente o gradiente usado para ajustar os pesos.
- A ausência de dados não rotulados impede o modelo de explorar a estrutura natural do espaço de imagens.

FixMatch



- Combina duas ideias: pseudo-rotulagem e consistency regularization, usando versões fracas e fortes da mesma imagem.
- A dimensão não rotulada é utilizada para propagar a decisão do modelo a exemplos sem rótulo, criando um treinamento semi-supervisionado.
- O EMA (Exponential Moving Average) mantém uma versão suavizada dos pesos, servindo como referência mais estável durante o aprendizado.

Pseudo-rotulos

- O modelo gera um rótulo para cada imagem não rotulada usando uma predição de alta confiança.
- Apenas pseudo-rótulos acima de um limiar (ex.: 0.95) são utilizados para evitar erros acumulados.
- Pseudo-rótulos funcionam como dados supervisionados adicionais, permitindo aprender padrões que não estão presentes nos rótulos reais.

Teste adicional

- E se alterássemos tau ao longo das épocas de um mesmo treinamento?

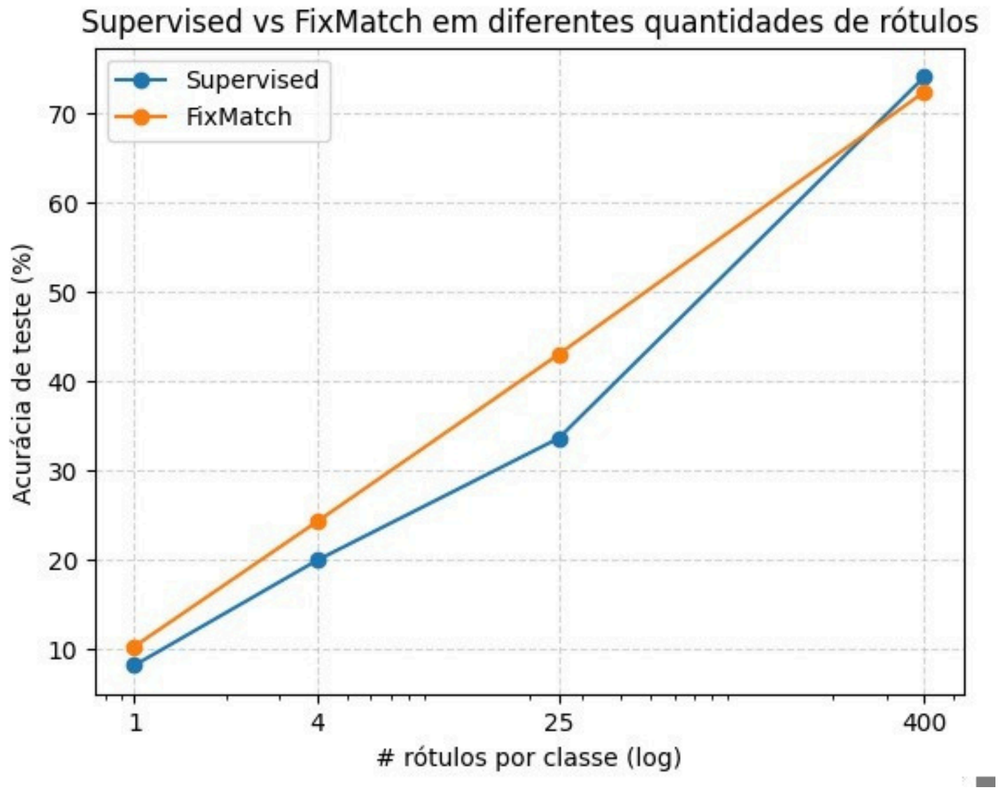
```
for epoch in range(num_epochs):
    # agenda do τ: decai de tau_start -> self.tau ao longo de tau_warmup_epochs
    if tau_start is None or tau_warmup_epochs <= 0:
        current_tau = self.tau
    else:
        frac = min(1.0, (epoch + 1) / max(1, tau_warmup_epochs))
        current_tau = tau_start - (tau_start - self.tau) * frac
```

```
[Extra] FixMatch tau=0.8 - 400 lbl/cls: 72.67%
```

```
FixMatch (modelo bruto) acc: 72.43
```

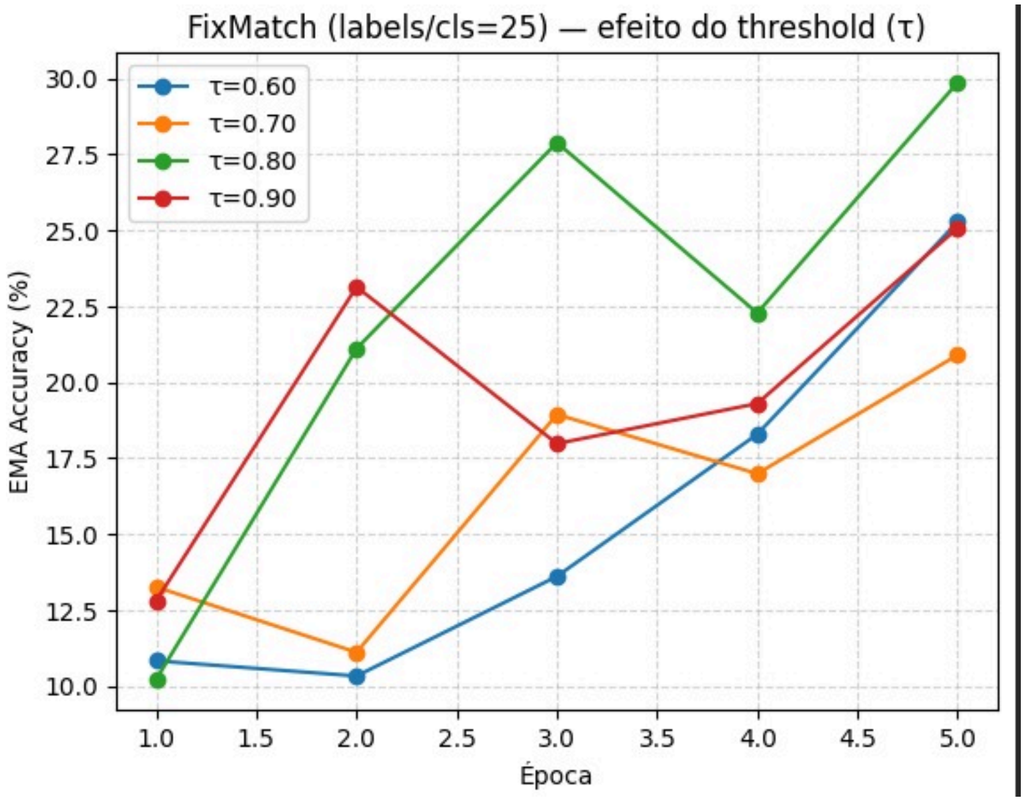
FixMatch vs Supervisionado

Com modelo pré-treinado



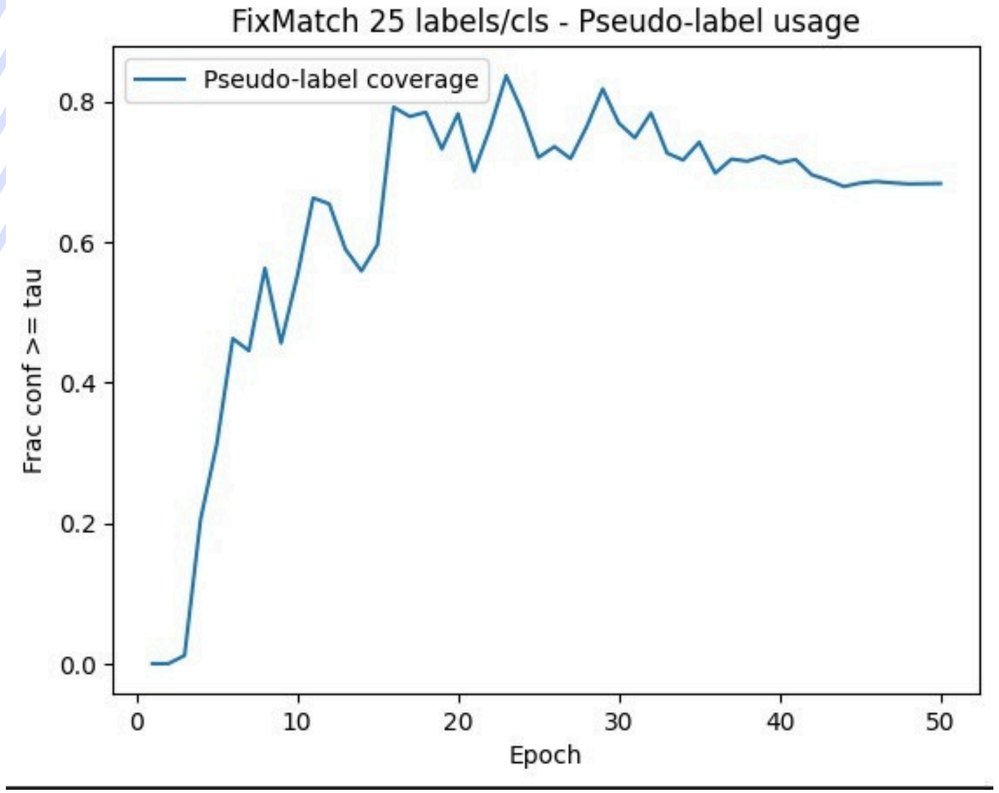
Acurácia x Tau

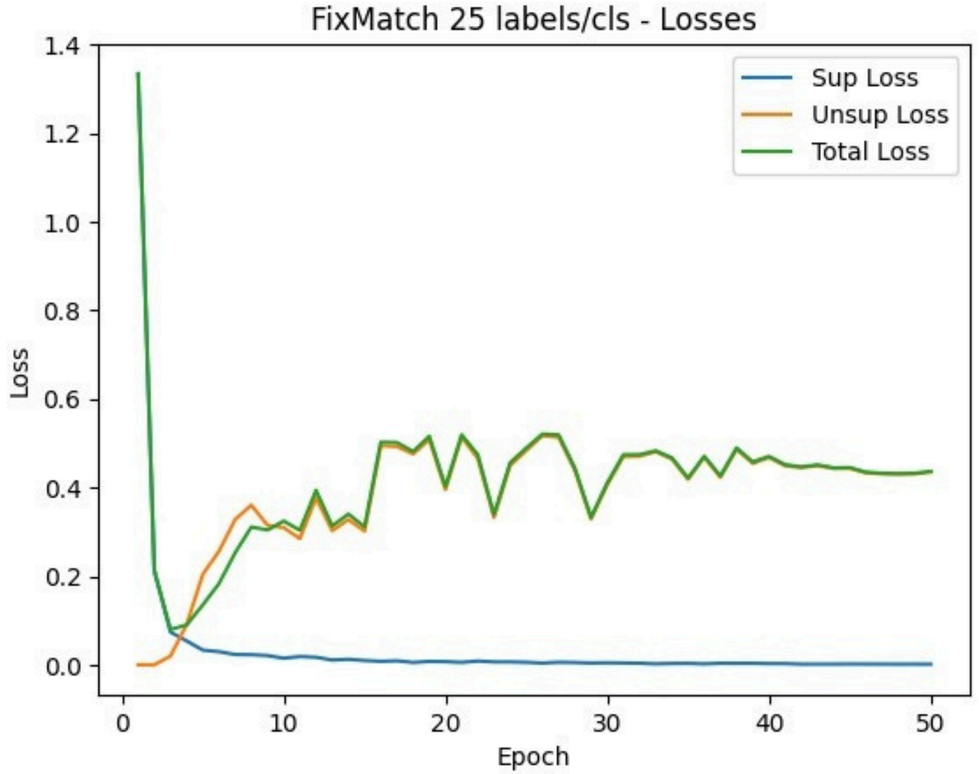
Com modelo pré-treinado



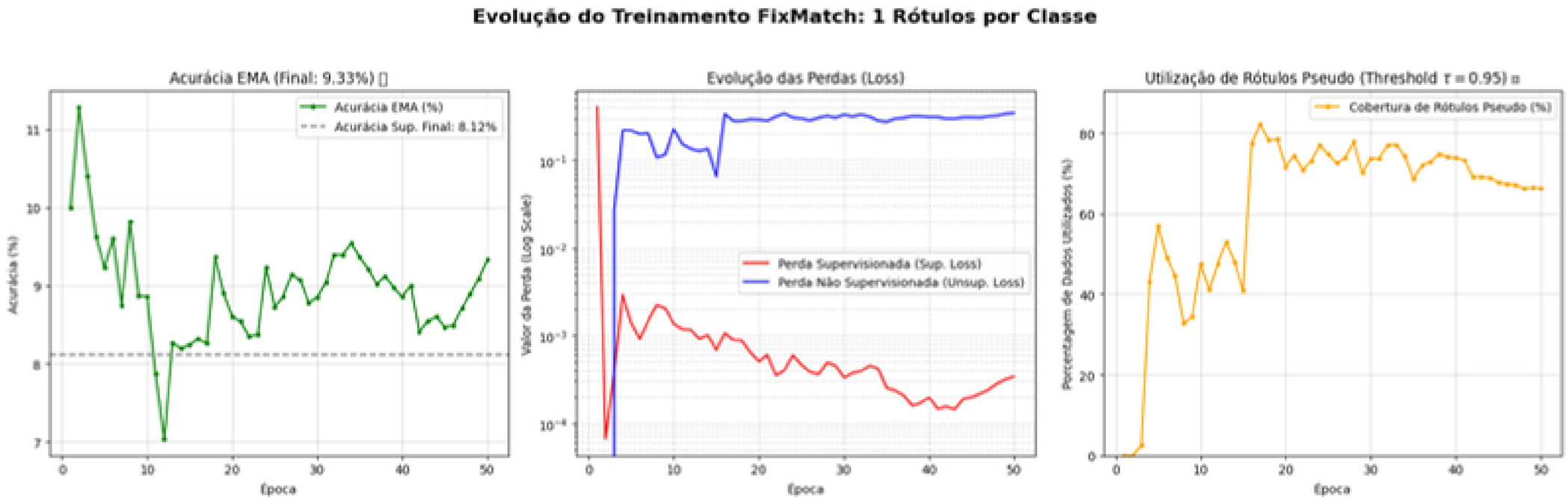
Limitações

- **Sensibilidade a τ e aug fortes: resultados degradam com τ mal ajustado/aug fracas (paper mostra piora sem Cutout/variações).**
- **Confirmation bias quando τ é baixo: pseudo-rótulos ruidosos prejudicam treino.**
- **Mesmo com Distribution Alignment simples, a cobertura pode ficar enviesada; impactos maiores nas configurações com pouquíssimos rótulos.**
- **Diferenças de backbone/augment/seed/tempo de treino (nós usamos ResNet-18 e 5–50 épocas em alguns testes; o paper usa WideResNet/Shake-Drop em partes e regimes mais longos).**

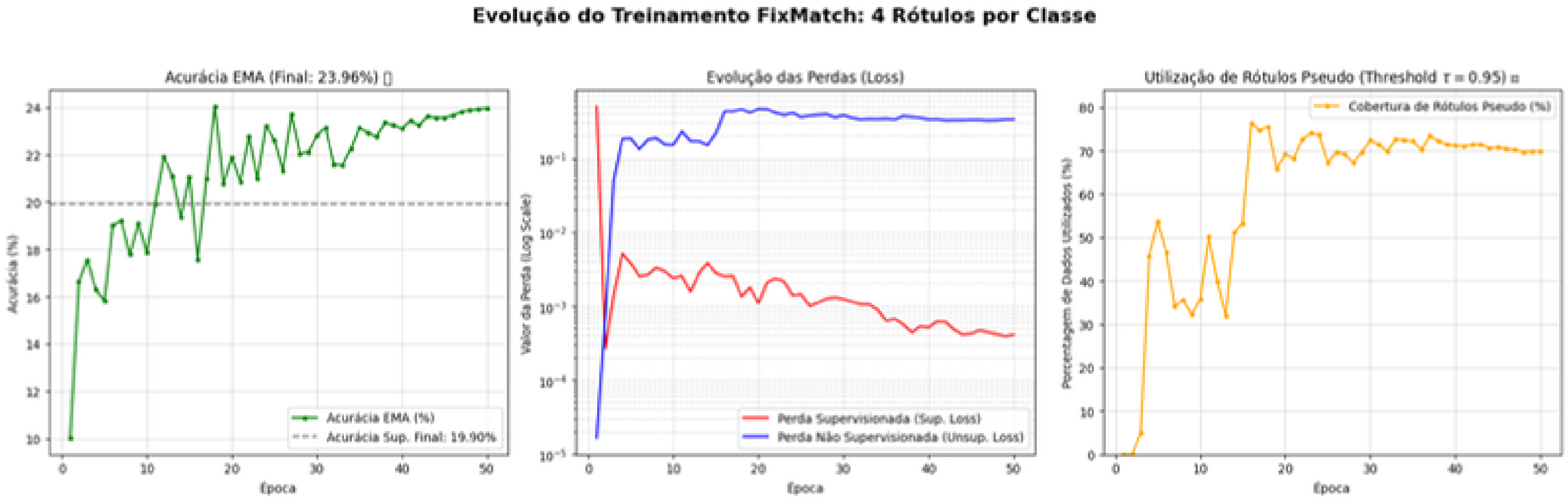

FixMatch 25 labels/cls - Pseudo-label usage
Pseudo-label coverage
Frac conf >= tau
Epoch
0.0
0.2
0.4
0.6
0.8
0
10
20
30
40
50


FixMatch 25 labels/cls - Losses
Sup Loss
Unsup Loss
Total Loss
Loss
Epoch
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
0
10
20
30
40
50

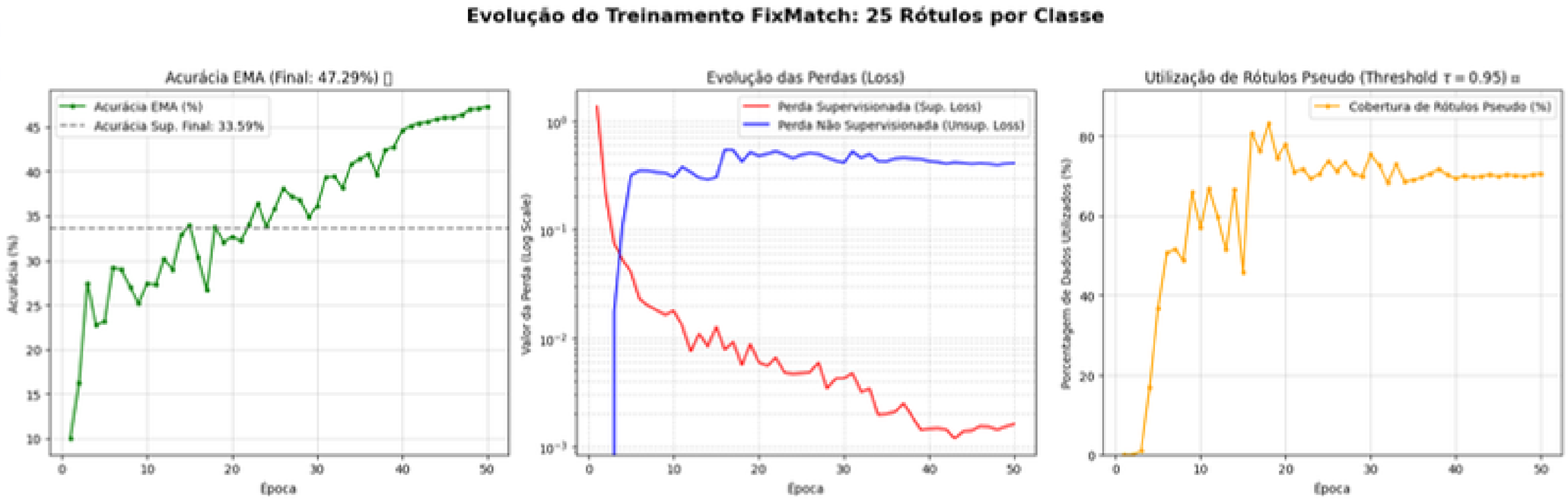
Classe (#)	Acurácia Sup. (%)	Acurácia FixMatch (%)	Acurácia FixMatch τ=0.8 (Dados Brutos) (%)	Tendência (EMA Acc ↑)	Cobertura dos Pseudo-rótulos (%)	Observações Principais
1 Label	8,12	10,23	**9.33** (um pouco menor)	**Moderada (8.54 → 9.33)**	**Baixa (66.29)**	FixMatch oferece um ganho modesto (2.11 p.p.) sobre o supervisionado, mas a acurácia é geralmente baixa.
4 Labels	19,9	24,24	**23.96** (muito próxima)	**Forte (23.92 → 23.96)**	**Média (69.84)**	Ganho significativo de FixMatch (4.34 p.p.).
25 Labels	33,59	43,00	**47.29** (substancialmente maior)	**Muito Forte (47.03 → 47.29)**	**Média-Alta (70.57)**	Maior ganho absoluto (13.70 p.p.) com FixMatch, indicando que o método semi-supervisionado é **muito eficaz** com esta quantidade de rótulos.
400 Labels	74,1	72,43	**72.67** (muito próxima)	**Forte (70.58 → 72.67)**	**Alta (78.91)**	O desempenho supervisionado é o mais alto. FixMatch não supera o modelo puramente supervisionado, mas mantém uma acurácia próxima e a **maior cobertura** de pseudo-rótulos.



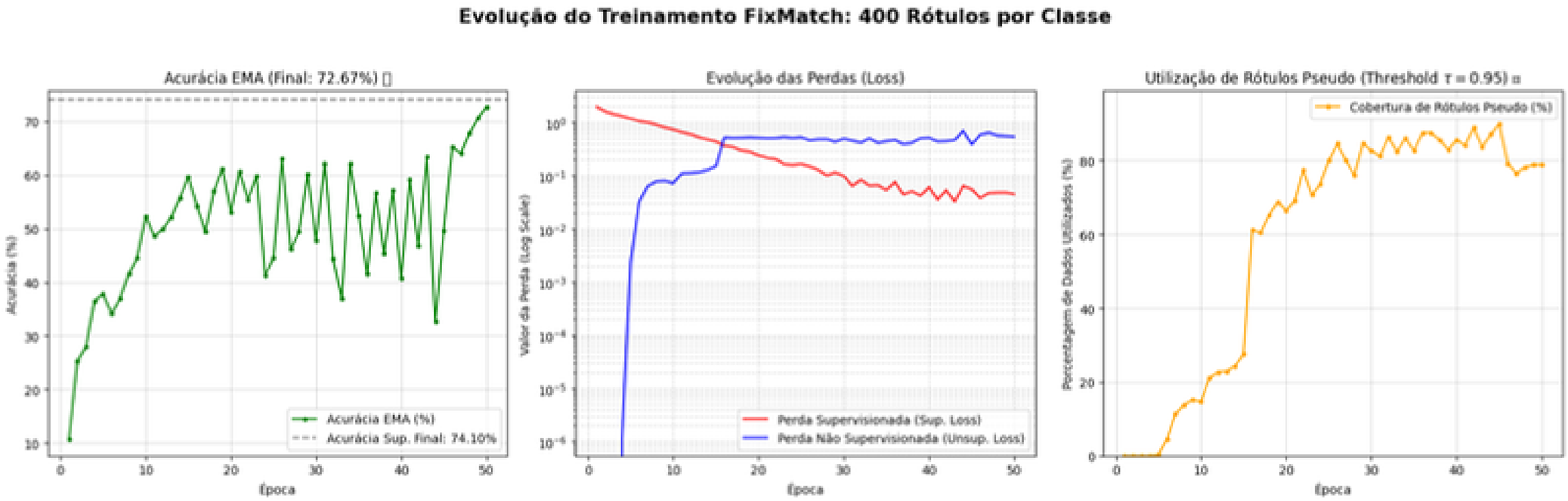
- Acurácia (9.33%): Altamente volátil (7% a 11.5%) e só ligeiramente acima da acurácia supervisionada (8.12%).
- Perdas: Sup. Loss (vermelho) muito baixa (<10^3), indicando overfitting; Unsup. Loss (azul) alta e dominante.
- Cobertura Pseudo: Cresce rápido e se mantém alta, na faixa de 70-80%.



- Acurácia (23.96%): Crescimento estável e significativo, superando claramente a acurácia supervisionada (19.90%).
- Perdas: Sup. Loss (vermelho) em queda gradual, estabilizando-se em torno de <10^3, refletindo melhor generalização.
- Cobertura Pseudo: Crescimento rápido nas primeiras épocas, estabilizando-se consistentemente na faixa de 70-75%.



- Acurácia (43.00%): Esperado um crescimento suave e consistente, atingindo alto desempenho, com ganho significativo sobre o Sup. (33.59%).
- Perdas: Sup. Loss esperada mais alta que as classes 1 e 4, indicando melhor fit e capacidade de generalização com mais dados rotulados.
- Cobertura Pseudo: Esperada uma alta e estável utilização de rótulos pseudo, devido à base rotulada mais forte.



- Acurácia (72.43%): Esperado alto crescimento inicial e estabilidade, alcançando desempenho próximo ao modelo Sup. (74.10%).
- Perdas: Ambas as perdas devem cair gradualmente, com a Sup. Loss sendo a componente dominante e se estabilizando no valor mais alto entre todas as classes.
- Cobertura Pseudo: Esperada a confiança mais alta e imediata na geração de rótulos pseudo, devido à vasta base de dados rotulados.

Obrigado!