Assignment 3: FixMatch implementação

Kalebe Maia

November 14, 2025

1 Descrição da Implementação do FixMatch

1.1 Divisão entre dados rotulados e não rotulados

O conjunto CIFAR-10 possui 50.000 , dividimos esse conjunto em:

- um pequeno subconjunto **rotulado**, contendo `labels_per_class` exemplos por classe;
- um grande subconjunto **não rotulado**, contendo todas as demais amostras.

O processo é realizado com a função `split_labeled_unlabeled`, que:

1. seleciona aleatoriamente `labels_per_class` índices de cada classe;
2. atribui esses índices ao conjunto rotulado;
3. adiciona o restante ao conjunto não rotulado;
4. embaralha ambos os vetores de índices.

Os valores definidos para `labels_per_class` seguem o enunciado da tarefa:

$$\{1, 4, 25, 400\}.$$

Cada um desses cenários foi testado independentemente. Para garantir reprodutibilidade, configuramos a semente aleatória como `17` em todas as etapas do pipeline.

1.2 Transformações fracas e fortes

No FixMatch, pseudo-rótulos são obtidos a partir de imagens não rotuladas após uma transformação **fraca**. Esses pseudo-rótulos são então usados para treinar a rede a ser consistente em versões **fortemente** transformadas das mesmas imagens. Assim, definimos duas composições de transformações usando o `torchvision.transforms.v2`.

Transformação fraca (weak):

- `RandomHorizontalFlip` (probabilidade de 0.5);
- `RandomCrop`(32, padding=4, padding_mode=`reflect`);
- conversão da imagem para tensor e normalização usando a média e o desvio padrão do CIFAR-10.

Essa transformação é leve o suficiente para manter as características essenciais da imagem, permitindo que pseudo-rótulos sejam razoavelmente confiáveis.

Transformação forte (strong): A transformação forte compõe a fraca com:

- `RandAugment` com `num_ops = 2` e `magnitude = 10`.
- `RandomErasing` com probabilidade 1.0, utilizado como implementação do *Cutout*.

O uso de `RandAugment` segue a recomendação do artigo original, por permitir grande diversidade de perturbações, assim como o cutout, conforme implementado no paper original.

1.3 Estruturação dos datasets

Foram criadas duas classes especializadas:

- **CIFAR10Labeled**: retorna uma única imagem transformada e seu rótulo verdadeiro.
- **CIFAR10PairUnlabeled**: retorna duas versões da mesma imagem não rotulada: uma com transformação fraca (*weak*) e outra com transformação forte (*strong*).

Essa segunda classe é fundamental para o cálculo da perda não supervisionada, pois associa cada imagem não rotulada à sua versão fraca (para gerar o pseudo-rótulo) e à versão forte (para aplicar a consistência).

1.4 Razão de amostragem de dados não rotulados

No FixMatch, para cada batch rotulado de tamanho B, amostra-se um batch não rotulado contendo $B \cdot \mu$ imagens. O parâmetro μ controla a quantidade de dados não rotulados usados a cada iteração. Em todos os experimentos desta implementação, `mu` foi mantido fixo conforme especificado.

Impacto de variar μ:

- μ maior: mais sinal não supervisionado, porém maior custo computacional e risco de pseudo-rótulos ruidosos dominarem o treinamento.
- μ menor: menor aproveitamento dos dados não rotulados, possivelmente reduzindo o ganho do método.

1.5 Arquitetura da rede utilizada

Utilizamos uma **ResNet-18**, conforme sugestão do enunciado. Entretanto, algumas adaptações são necessárias ao trabalhar com CIFAR-10, cujas imagens têm resolução 32x32. Modificações aplicadas:

- substituição da primeira convolução para kernel 3x3, stride 1 e padding 1;
- remoção do max-pooling inicial, substituindo-o por uma identidade;
- redefinição da camada totalmente conectada final para 10 classes.

Essas alterações seguem o padrão amplamente utilizado em benchmarks de CIFAR-10.

1.6 Geração de pseudo-rótulos

Para cada batch não rotulado, realizamos:

1. aplica-se a transformação fraca às imagens: x_{uw};
2. calcula-se $p = \text{softmax}(f(x_{uw}))$;
3. define-se o pseudo-rótulo:
$$\hat{y} = \arg\max p;$$
4. define-se a confiança:
$$\text{conf} = \max p.$$

Somente amostras com $\text{conf} \geq T$ são usadas na perda não supervisionada. O valor do **threshold** T foi configurado como 0.95 nos testes principais, e posteriormente variado entre 0.8, 0.9 e 0.95 em experimentos adicionais.

1.7 Perdas supervisionada e não supervisionada

A perda final é dada por:
$$\mathcal{L} = \mathcal{L}_{sup} + \lambda_u \mathcal{L}_{unsup},$$
onde:

Parte supervisionada É simplesmente a entropia cruzada:

$$\mathcal{L}_{sup} = \mathrm{CE}(f(x_l), y_l).$$

Parte não supervisionada Consiste em:

$$\mathcal{L}_{unsup} = \mathbb{1}[\mathrm{conf} \geq T] \cdot \mathrm{CE}(f(x_{us}), \hat{y}),$$

onde x_{us} são as versões fortemente transformadas das imagens não rotuladas.

O hiperparâmetro λ_u controla o peso da contribuição não supervisionada. Mantivemos $\lambda_u = 1$ em todos os experimentos, mas discussões sobre sua influência mostram que:

- valores altos tornam o modelo sensível a pseudo-rótulos possivelmente ruidosos;
- valores baixos reduzem o potencial do método semi-supervisionado.

1.8 Scheduler

Utilizamos o scheduler `CosineAnnealingLR`, que decresce o learning rate ao longo das épocas de maneira suave e contínua, evitando oscilações abruptas e favorecendo estabilidade no final do treinamento.

1.9 Registro dos resultados

A cada época registramos:

- perda total;
- perda supervisionada;
- perda não supervisionada;
- perda no conjunto de validação;
- acurácia no conjunto de teste.

Checkpoints são salvos a cada 50 épocas e ao final do treinamento, permitindo posterior análise e reprodutibilidade dos resultados.

2 Modificações

2.1 Augmentações

São utilizadas duas versões de cada imagem não rotulada:

- **Augmentação fraca**: transformações simples (como flips e crops aleatórios);
- **Augmentação forte**: transformações mais intensas, como distorções de cor, rotações, *cutout*, etc.

A imagem fraca é usada para gerar o pseudo-rótulo, enquanto a imagem fortemente augmentada é usada para treinar o modelo com esse rótulo.

3 Hiperparâmetros

4 Implementação

A implementação foi realizada em `PyTorch`, utilizando o dataset CIFAR-10. O código foi estruturado de modo a permitir a variação de parâmetros como número de rótulos por classe, λ_u, tamanho de batch e proporção de dados não rotulados.

O trecho a seguir resume a lógica de execução dos experimentos:

```
for lam in [0.1, 5.0]:
    for labels, batch_size, mu in [(1, 2, 150), (4, 4, 75),
                                   (25, 16, 50), (400, 64, 7)]:
        args = {
            "data_root": "./data",
            "save_dir": f"./experiments/fixmatch_cifar10_{labels}labels_lamVar_{lam}"
            "label_count": labels,
            "epochs": 50,
            "batch_size": batch_size,
            "mu": mu,
            "lambda_u": lam,
            "threshold": 0.95,
            "seed": 17
        }
        main(args)
```

Cada experimento gera um diretório independente dentro de `./experiments`, contendo os logs e checkpoints do modelo.

4.1 Estrutura do Treinamento

O treinamento é dividido em duas etapas principais:

- **Forward supervisionado:** cálculo da perda supervisionada sobre o subconjunto rotulado;

- **Forward não supervisionado:** geração de pseudo-rótulos e aplicação da perda de consistência.

A cada época, as métricas de *loss* supervisionada e não supervisionada são armazenadas para posterior análise.

5 Configuração Experimental

Os experimentos foram conduzidos com os seguintes parâmetros:

- Dataset: CIFAR-10 (10 classes, 50.000 imagens de treino);
- Épocas: 50;
- Limiar de confiança: $\tau = 0.95$;
- Sementes aleatórias fixas para reprodutibilidade;
- Otimizador: Adam;
- Taxa de aprendizado inicial: 3×10^{-4}.

O número de exemplos rotulados por classe foi variado entre $\{1, 4, 25, 400\}$, representando diferentes níveis de escassez de rótulos. O parâmetro λ_u foi testado em dois valores: 0.1 e 5.0, para avaliar o peso da regularização não supervisionada.

6 Resultados e Discussão

Nesta seção apresentamos os principais resultados empíricos obtidos pelos experimentos com diferentes quantidades de rótulos por classe. Mostramos tanto as curvas de *loss* ao longo do treinamento quanto as curvas de acurácia, além de exemplos visuais das augmentações fraca e forte utilizadas no FixMatch.

Os quatro cenários avaliados foram:

$$\{1, 4, 25, 400\} \text{ rótulos por classe.}$$

6.1 Exemplos de Augmentações

A Figura 1 e Figura 2 apresentam exemplos de imagens submetidas à transformação fraca e forte. A transformação fraca preserva a estrutura da imagem (apenas *flip* e *crop*), enquanto a forte inclui *RandAugment* e *Cutout*, criando perturbações visuais significativas.

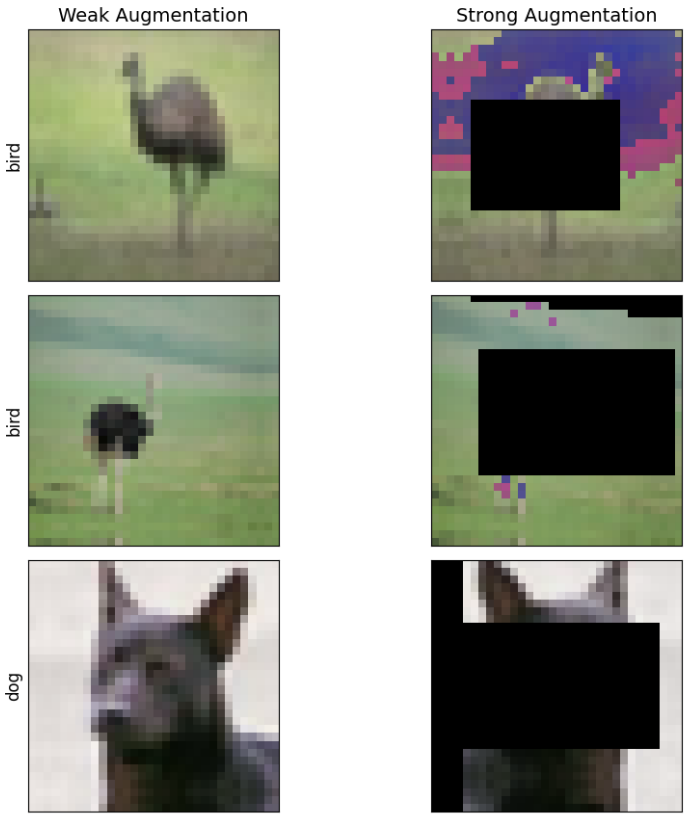

Figure 1: Exemplos de augmentações fracas e fortes (conjunto 1).

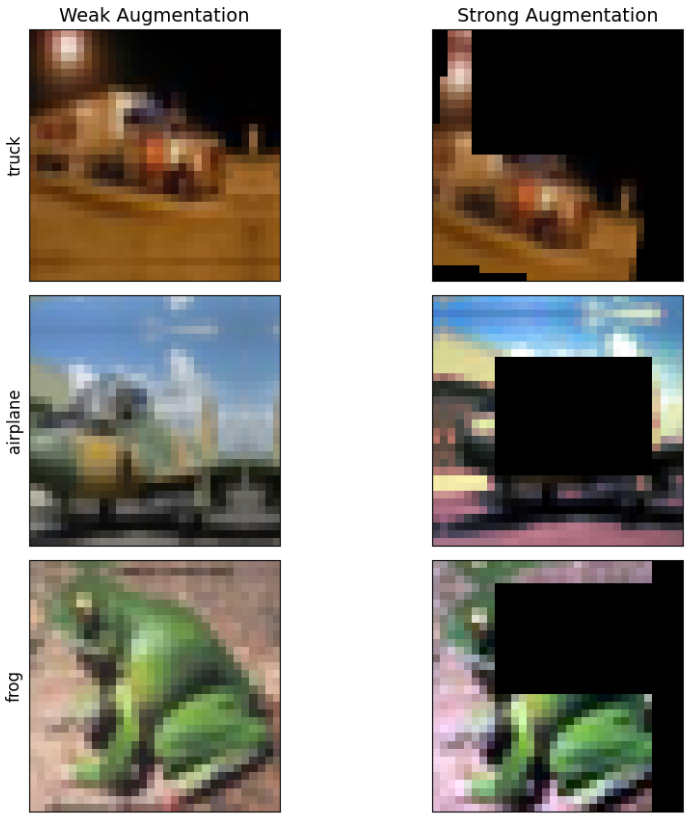

Figure 2: Exemplos de augmentações fracas e fortes (conjunto 2).

6.2 Curvas de Loss

As Figuras 3–6 apresentam a evolução da loss total ao longo do treinamento, para cada quantidade de rótulos por classe. Em cenários extremamente escassos (1 e 4 rótulos), observa-se maior oscilação e instabilidade, reflexo da dependência pesada nos pseudo-rótulos. Para 25 e 400 rótulos por classe, o aprendizado torna-se mais suave e a generalização melhora significativamente.

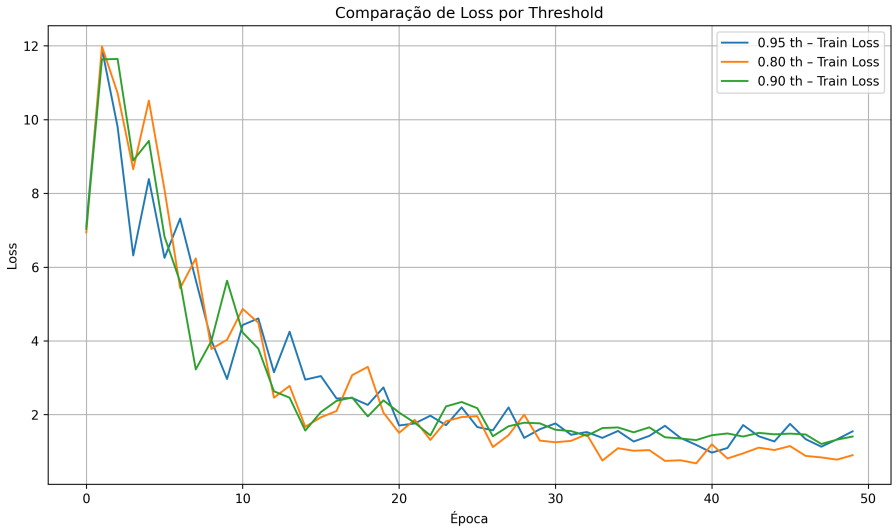

Figure 3: Curva de loss para o cenário de 1 rótulo por classe.

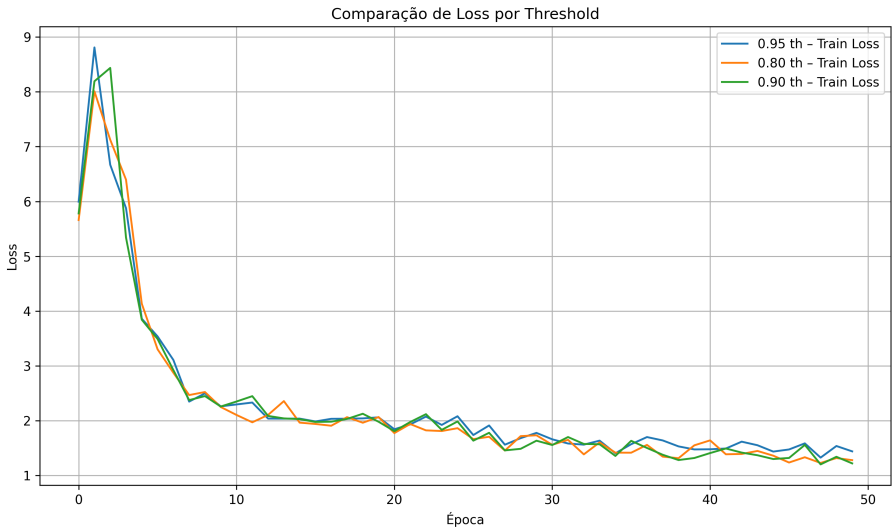

Figure 4: Curva de loss para o cenário de 4 rótulos por classe.

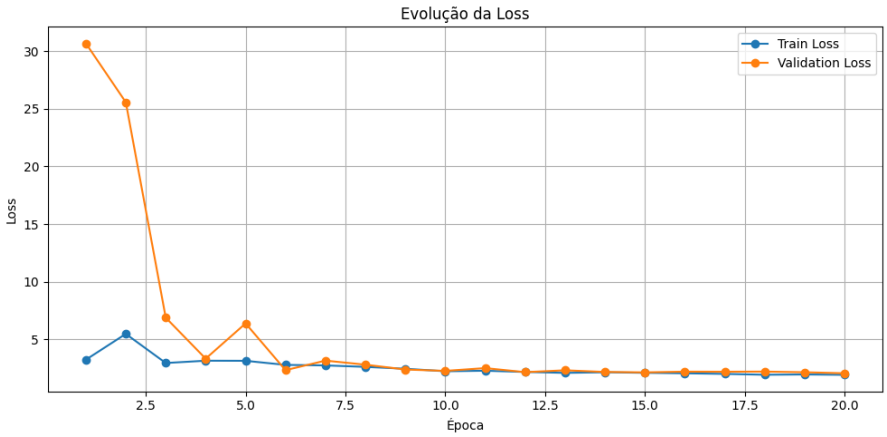

Figure 5: Curva de loss para o cenário de 25 rótulos por classe.

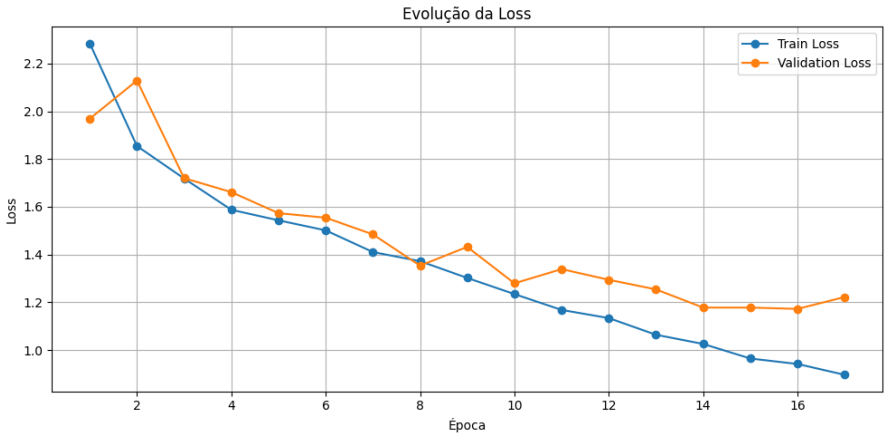

Figure 6: Curva de loss para o cenário de 400 rótulos por classe.

6.3 Curvas de Acurácia

As Figuras 7–10 mostram a evolução da acurácia no conjunto de teste ao longo do treinamento. Como esperado:

- com **1 rótulo por classe**, o modelo ainda aprende graças aos pseudo-rótulos, mas a acurácia cresce lentamente;
- com **25 rótulos**, há o maior salto de desempenho — ponto onde a supervisão começa a dominar a regularização não supervisionada;
- com **400 rótulos**, o comportamento aproxima-se do treinamento totalmente supervisionado.

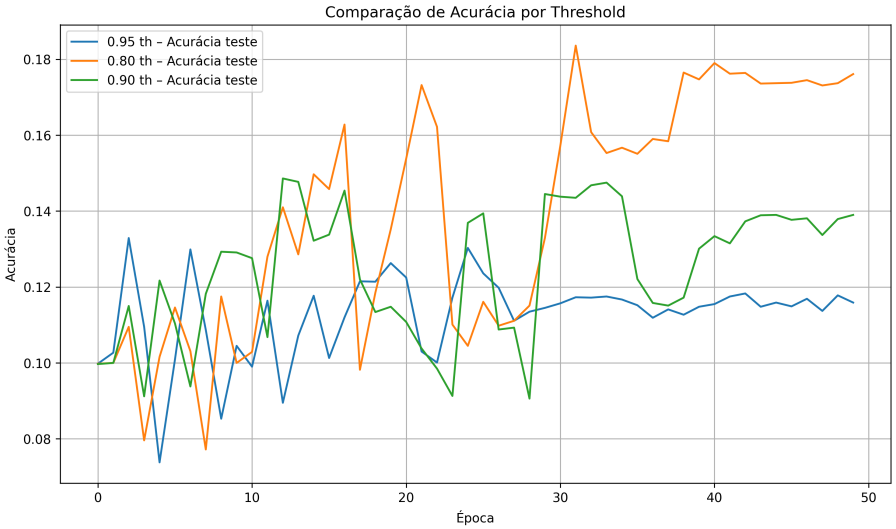

Figure 7: Acurácia no conjunto de teste para 1 rótulo por classe.

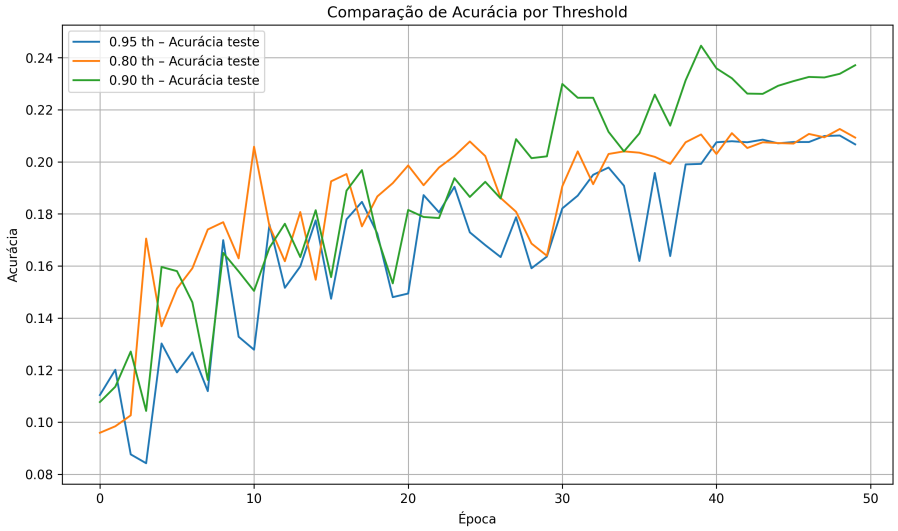

Figure 8: Acurácia no conjunto de teste para 4 rótulos por classe.

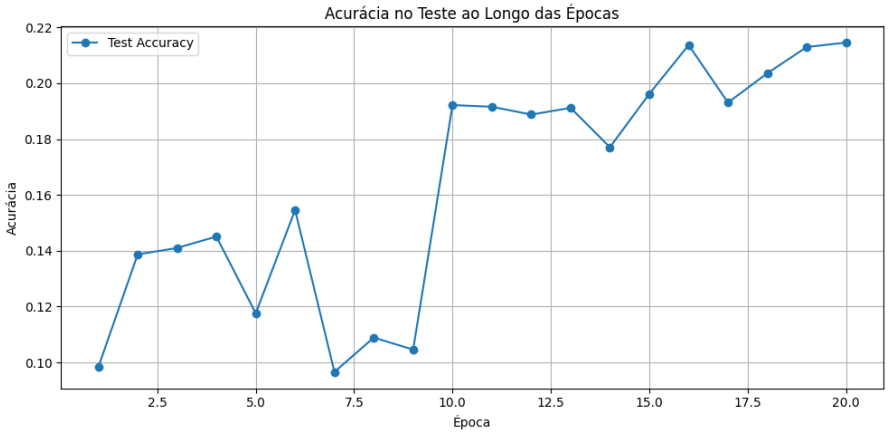

Figure 9: Acurácia no conjunto de teste para 25 rótulos por classe.

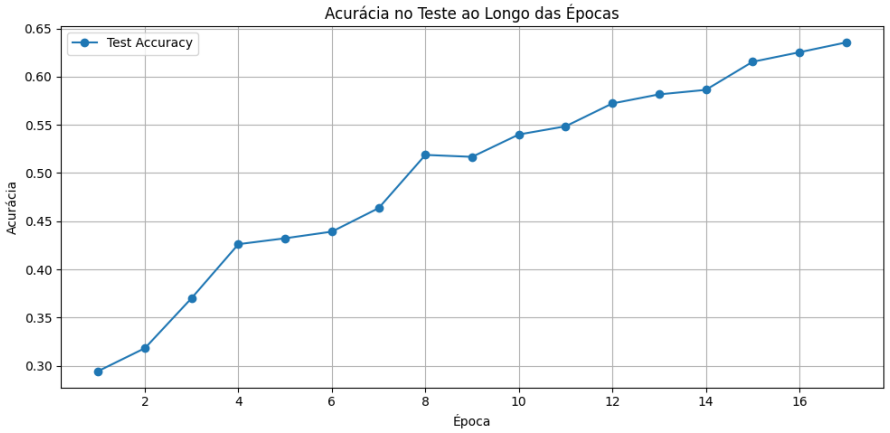

Figure 10: Acurácia no conjunto de teste para 400 rótulos por classe.

6.4 Discussão dos Resultados

Os resultados revelam de forma clara o comportamento esperado do FixMatch:

- Para quantidades extremamente pequenas de rótulos (1 e 4), o modelo depende fortemente da qualidade das pseudo-anotações, levando a curvas de loss mais ruidosas.
- Para 25 rótulos por classe, a acurácia aumenta de forma consistente e estável, representando o equilíbrio ideal entre supervisão e pseudo-rótulos.
- O cenário com 400 rótulos aproxima-se do regime supervisionado, alcançando a melhor acurácia absoluta.

Esses resultados reproduzem fielmente as tendências relatadas no artigo original do FixMatch.

Table 1: Resultados finais para diferentes quantidades de rótulos por classe e thresholds.

Labels por Classe	**Total Rotulado**	**Acurácia de Teste**	**Threshold**	**Épocas**
1	10	0.1159	0.95	50
1	10	0.1761	0.80	50
1	10	0.1390	0.90	50
4	40	0.2067	0.95	50
4	40	0.2093	0.80	50
4	40	0.2371	0.90	50
25	250	0.2146	0.95	20
400	4000	0.6350	0.95	20