Pontifícia Universidade Católica
do Rio de Janeiro

Vitor Bento de Sousa

RDS – Recuperando Amostras Descartadas com Rótulos Ruidosos: Técnicas para Treinamento de Modelos de Deep Learning com Amostras Ruidosas

Tese de Doutorado

Tese apresentada como requisito parcial para obtenção do grau de Doutor pelo Programa de Pós-Graduação em Engenharia Elétrica da PUC-Rio.

Orientador: Marco Aurelio Pacheco
Coorientador: Manoela Kohler

Rio de Janeiro, 05 de Março de 2024

Pontifícia Universidade Católica do Rio de Janeiro

Vitor Bento de Sousa

RDS – Recuperando Amostras Descartadas com Rótulos Ruidosos: Técnicas para Treinamento de Modelos de Deep Learning com Amostras Ruidosas

Tese apresentada como requisito parcial para obtenção do grau de Doutor pelo Programa de Pós-Graduação em Engenharia Elétrica da PUC-Rio. Aprovada pela Comissão Examinadora abaixo:

Prof.: Marco Aurelio Pacheco
Orientador
Departamento de Engenharia Elétrica PUC-RIO

Prof.: Manoela Kohler
Coorientador
Departamento de Engenharia Elétrica PUC-RIO

Prof.: Marley Vellasco
Departamento de Engenharia Elétrica PUC-RIO

Prof.: Wouter Caarls
Departamento de Engenharia Elétrica PUC-RIO

Prof.: José David Bermudez Castro
McMaster University

Prof.: Leonardo Forero Mendoza
Universidade do Estado do Rio de Janeiro

Prof.: Cristina Maria Bentz
Petrobras

Rio de Janeiro, 05 de Março de 2024

Todos os direitos reservados. A reprodução, total ou parcial do trabalho, é proibida sem a autorização da universidade, do autor e do orientador.

Vitor Bento de Sousa

Graduou-se em Engenharia Elétrica na Universidade Federal Fluminense (UFF) em 2017. Mestre em Engenharia Elétrica pela PUC-RIO. Participa de projetos de P&D na área de Petróleo e Gás envolvendo Deep Learning e Visão Computacional.

Ficha Catalográfica

Sousa, Vitor Bento de

RDS recuperando amostras descartadas com rótulos ruidosos : técnicas para treinamento de modelos de deep learning com amostras ruidosas / Vitor Bento de Sousa ; orientador: Marco Aurelio Pacheco ; coorientador: Manoela Kohler. – 2024.

127 f. : il. color. ; 30 cm

Tese (doutorado)–Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Elétrica, 2024.
Inclui bibliografia

1. Engenharia Elétrica – Teses. 2. Amostras ruidosas. 3. Aprendizado profundo. 4. Multiclasse. 5. Multilabel. I. Pacheco, Marco Aurelio. II. Kohler, Manoela. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Elétrica. IV. Título.

CDD: 621.3

Agradecimentos

Agradeço a minha família e a todos os amigos que verdadeiramente me incentivaram e apoiaram na elaboração deste trabalho. Agradeço ao professor Marco Aurelio por todo o suporte fornecido através da equipe ICA. Agradeço a professora Manoela por todo apoio sem os quais este trabalho não poderia ser realizado.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

O autor gostaria de agradecer o Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (Capes) e a Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio) pelo suporte financeiro para o trabalho.

Resumo

Sousa, Vitor Bento; Kohler, Manoela; Pacheco, Marco Aurelio. **RDS – Recuperando Amostras Descartadas com Rótulos Ruidosos: Técnicas para Treinamento de Modelos de Deep Learning com Amostras Ruidosas.** Rio de Janeiro, 2024. 127 p. Tese de Doutorado - Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

Modelos de *Aprendizado Profundo* para classificação de imagens alcançaram o estado da arte em um vasto campo de aplicações. Entretanto, é frequente deparar-se com amostras ruidosas, isto é, amostras contendo rótulos incorretos, nos conjuntos de dados provenientes de aplicações do mundo real. Quando modelos de *Aprendizado Profundo* são treinados nestes conjuntos de dados, a sua performance é prejudicada. Modelos do estado da arte, como *Co-teaching+* e *Jocor*, utilizam a técnica "*Small Loss Approach*" (SLA) para lidar com amostras ruidosas no cenário multiclasse. Nesse trabalho, foi desenvolvido uma nova técnica para lidar com amostras ruidosas, chamada *Recovering Discarded Samples* (RDS), que atua em conjunto com a SLA. Para demostrar a eficácia da técnica, aplicou-se o *RDS* nos modelos *Co-teaching+* e *Jocor* resultando em dois novos modelos *RDS-C* e *RDS-J*. Os resultados indicam ganhos de até 6% nas métricas de teste para ambos os modelos. Um terceiro modelo chamado *RDS-Contrastive* também foi desenvolvido, este modelo superou o estado da arte em até 4% na acurácia de teste. Além disso, nesse trabalho, expandiu-se a técnica SLA para o cenário multilabel, sendo desenvolvido a técnica SLA Multilabel (SLAM). Com essa técnica foi desenvolvido mais dois modelos para cenário multilabel com amostras ruidosas. Os modelos desenvolvidos nesse trabalho para multiclasse foram utilizados em um problema real de cunho ambiental. Os modelos desenvolvidos para o cenário multilabel foram aplicados como solução para um problema real na área de óleo e gás.

Palavras-chave

Amostras ruidosas; Aprendizado Profundo; Multiclasse; Multilabel.

Abstract

Sousa, Vitor Bento; Kohler, Manoela; Pacheco, Marco Aurelio. **RDS - Recovering Discarded Samples with Noisy Labels: Techniques for Training Deep Learning Models with Noisy Samples.** Rio de Janeiro, 2024. 127 p. Tese de Doutorado - Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

Deep Learning models designed for image classification have consistently achieved state-of-the-art performance across a plethora of applications. However, the presence of noisy samples, i.e., instances with incorrect labels, is a prevalent challenge in datasets derived from real-world applications. The training of Deep Learning models on such datasets inevitably compromises their performance. State-of-the-art models, such as Co-teaching+ and Jocor, utilize the "Small Loss Approach" (SLA) technique to handle noisy samples in a multi-class scenario. In this work, a new technique named Recovering Discarded Samples (RDS) was developed to address noisy samples, working with SLA. To demonstrate the effectiveness of the technique, RDS was applied to the Co-teaching+ and Jocor models, resulting in two new models, RDS-C and RDS-J. The results indicate gains of up to 6% in test metrics for both models. A third model, named RDS-Contrastive, was also developed, surpassing the state-of-the-art by up to 4% in test accuracy. Furthermore, this work extended the SLA technique to the multilabel scenario, leading to the development of the SLA Multilabel (SLAM) technique. With this technique, two additional models for the multilabel scenario with noisy samples were developed. The models proposed in this work for the multiclass scenario were applied in a real-world environmental solution, while the models developed for the multilabel scenario were implemented as a solution for a real problem in the oil and gas industry.

Keywords

Noisy samples; Deep Learning; Multiclass; Multilabel.

Sumário

Índice de Figuras

Índice de Tabelas

Algoritmos

1
Introdução

Modelos de *Deep Learning (DL)* para classificação de imagens [1] alcançaram o estado da arte em um vasto campo de aplicações [2], [3], [4]. Atualmente, um dos desafios da área, oriundos de aplicações reais, é o aprendizado destes modelos com amostras ruidosas [5]- o termo "*amostras ruidosas*" se refere a amostras com rótulos incorretos presentes no conjunto de dados -. Modelos de *DL* treinados nesse cenário possuem baixo desempenho, o que é altamente indesejado [6]. Algumas fontes comuns de ruído em conjuntos de dados são consultas na *web* [7], *crowdsourcing* [8], anotações feitas por não especialistas ou até mesmo especialistas em tarefas de anotações muito desafiadoras, subjetivas, repetitivas ou cansativas [9].

Alguns trabalhos anteriormente propostos e desenvolvidos com o foco nessa problemática, com a utilização de *DL,* são focados em *transition matrix* [10], [11]. A deficiência desta abordagem ocorre quando o número de classes no conjunto de dados é muito grande, o que torna a estimativa da matriz de transição muito complicada [5]. Os métodos atuais encontrados na literatura se concentram em outra abordagem baseada na seleção de amostras limpas. Ou seja, essas técnicas selecionam as amostras com os rótulos corretos presentes no conjunto de dados e exclui do treino as amostras ruidosas, e.g. [12] e [13]. Essa abordagem traz a vantagem de não precisar estimar a matriz de transição, além de atingir melhores resultados quando comparados com modelos baseados em *transition matrix*. Uma outra vantagem desses modelos é a praticidade de implementação computacional. Os modelos do estado da arte (SOTA, do inglês *State Of The Art*), *Co-teaching* [5], *Co-teaching+* [14] e *Jocor* [15] são desta última categoria, e para selecionar as amostras limpas, a técnica *Small loss Approach* (SLA) é utilizada [5]. A técnica *SLA* exclui do treinamento as amostras com os maiores resultados na função de custo. Os modelos do SOTA mencionados empregam diferentes estratégias nas amostras selecionadas para otimizar o desempenho do modelo. Esses modelos são especificamente direcionados para tarefas de classificação multiclasse [16].

Nesse trabalho, uma nova técnica chamada *Recovering Discarded Samples* (RDS) foi proposta para lidar com amostras ruidosas. Essa técnica atua sobre as amostras limpas selecionadas pela SLA e sobre as amostras excluídas. A ideia principal

do RDS é recuperar as amostras excluídas atribuindo-as pseudolabels [17] e as retornando ao processo de treinamento.

O RDS atua em conjunto com a SLA e pode ser facilmente adaptado aos modelos do SOTA aumentando significativamente a sua performance. Nesse trabalho também foram propostos dois novos modelos combinando o modelo Jocor com o RDS e o modelo Co-teaching+ com o RDS, resultando nos novos modelos RDS-C e RDS-J. Os resultados apontam uma melhora dos dois modelos em até 6 % em relação ao F1-Score ao utilizar o RDS.

Além disso, também foi proposto um terceiro modelo, chamado RDS-Contrastive, que utiliza a técnica RDS e é inspirado em modelo de DL auto-supervisionado [18]. Os resultados indicam ganhos de até 4% em relação aos modelos do SOTA.

Os modelos desenvolvidos nessa tese para classificação multiclasse foram avaliados em três conjuntos de dados *benchmark:* Mnist [19], Cifar-10 e Cifar-100.[20] Como esses datasets são limpos, foi inserido ruído artificialmente para avaliação dos modelos seguindo o protocolo de [14]. Além desses *datasets*, os modelos foram avaliados em um quarto conjunto de dados *benchmark* Clothing1M [21]. Este conjunto de dados é ruidoso e contém 14 classes referentes a peças de roupas com 10^6 amostras, sendo a fonte do ruído proveniente de consultas na *web*. O ruído presente neste *dataset* é do tipo *open-set noisy* [22], i.e., existe a possibilidade de uma amostra não pertencer a nenhuma classe do *dataset*.

Em contraste com outros modelos do SOTA, como Jo-SRC [22] e *Self Adaptative Training* (SAT) [23], o modelo aqui proposto preserva a simplicidade enquanto lida com rótulos ruidosos de maneira eficaz. A simplicidade da nossa abordagem é importante para evitar hiperparâmetros difíceis de ajustar e demasiadamente dependentes do *dataset*, sendo então uma abordagem adequada para aplicações reais. Além disso, permite o uso de outras funções de custo comumente usadas em aplicações reais, e.g., a entropia cruzada ponderada para conjuntos de dados desbalanceados. Por fim, o treinamento do modelo proposto, em condições ideais, é equivalente ao treinamento de um modelo sem rótulos ruidosos no conjunto de treino, sendo esta uma grande vantagem da abordagem desenvolvida.

A motivação para o desenvolvimento desse trabalho foi uma demanda real de uma empresa de óleo e gás solicitada para o Laboratório de Inteligência Computacional Aplicada (ICA) da PUC-RIO. A empresa, por razões ambientais, necessitava de uma ferramenta para classificação de algas calcárias [24] no assoalho marinho. Ao longo dos

trabalhos realizados pela equipe do projeto foi identificada a presença de amostras ruidosas no conjunto de dados fornecido pela empresa, sendo então utilizados os modelos RDS-C, RDS-J e RDS-Contrastive para a solução do problema. Os resultados dessa aplicação são apresentados no capítulo Estudo de Caso. Por se tratarem de dados sigilosos, as imagens apresentadas são meramente ilustrativas, encontradas por buscas simples na internet, porém, do mesmo tipo presente no conjunto de dados real, não sendo assim apresentada nenhuma imagem real.

Nessa tese, também se expandiu a técnica SLA para o cenário de classificação multilabel [25] com amostras ruidosas. Classificação multilabel é uma tarefa onde uma dada entrada pode ser associada a múltiplos rótulos simultaneamente, em vez de um rótulo exclusivo como em classificação multiclasse. Embora a problemática de amostras ruidosas seja bem explorada na área de classificação multiclasse, trata-se ainda de uma área de estudo em seus estágios iniciais para multilabel.

Em [26] o autor aponta que alguns modelos usados em multiclasse podem ser aplicados para o cenário multilabel com pequenas mudanças na arquitetura. Embora essa abordagem possa ter um efeito positivo, nessa tese foi observado que a técnica SLA precisa de uma série de ajustes para se adequar ao problema multilabel.

No cenário multilabel, o ruído pode estar presente em algumas classes da imagem, enquanto outras classes da mesma imagem continuam com rótulos corretos. Frente a isso, é preciso adaptar a técnica SLA para essa possibilidade. Além disso, a técnica SLA exclui do treinamento as amostras com qualquer nível de ruído, entretanto no cenário multilabel é possível realizar a correção dos rótulos diretamente.

A técnica SLA ajustada para multilabel foi denominada como *Small Loss Approach Multilabel* (SLAM). Com essa adaptação foi possível desenvolver dois novos modelos *Learning by SLAM Multilabel* e *SLAM by Joint Loss* (SLAM-JL). O primeiro modelo desenvolvido foi a aplicação direta da técnica SLAM para treinamento de duas redes simultaneamente, seguindo os princípios do trabalho em [14]. O segundo modelo foi o ajuste da técnica SLAM para funcionar em cima da função de custo Joint Loss, apresentada no modelo multiclasse Jocor [15].

Por se tratar de uma área em seus estágios iniciais de estudos, foi preciso criar dois conjuntos de dados multilabel com amostras ruidosas. Foi então gerada uma versão do *dataset* UcMerced [27] com amostras multilabel ruidosas, e uma versão do *dataset* TreeSatAI [28], também com amostras ruidosas. Esses *datasets* estão disponíveis em https://github.com/ICA-PUC, permitindo que trabalhos futuros realizem comparações nas mesmas condições e se torne um benchmark para novos avanços na

área. Os modelos desenvolvidos para o cenário multilabel também foram validados em cima de um *dataset* multilabel real na área de óleo e gás, proveniente de inspeções submarinas.

Os modelos desenvolvidos aumentaram a performance sobre os conjuntos de dados de forma significativa. Considerando a métrica F1-Score, houve ganhos de até 17% para o *dataset* UcMerced e de até 3% para o *dataset* TreeSatAI, quando comparados aos outros modelos do SOTA. Para o conjunto de dados de inspeções submarinas, os ganhos atingiram 3%.

A primeira contribuição desse trabalho foi o desenvolvimento de uma nova técnica para lidar com amostras ruidosas no âmbito de DL voltada à classificação multiclasse. Essa técnica pode ser utilizada para o desenvolvimento de novos modelos de DL em trabalhos futuros. Outra contribuição importante foi o desenvolvimento de três novos modelos de *DL* para lidar com amostras para o cenário multiclasse. Outra contribuição foi a solução de uma demanda real de importância ambiental utilizando as técnicas desenvolvidas. Além disso, outra contribuição dessa tese foi a adaptação da técnica SLA para o cenário multilabel, que permite o desenvolvimento de novos modelos em trabalhos futuros. Essa adaptação permitiu o desenvolvimento de dois novos modelos para lidar com amostras ruidosas no cenário multilabel.

1.1. Objetivos

O objetivo da Tese:

Desenvolver novas técnicas e modelos de DL para lidar com amostras ruidosas em problemas reais.

Objetivos Específicos da Tese:

- Desenvolver uma nova técnica para lidar com amostras ruidosas em problemas de classificação multiclasse com modelos de DL;
- Desenvolver novos modelos de DL utilizando a nova técnica para lidar com amostras ruidosas em classificação multiclasse;
- Resolver problemas reais com os novos modelos desenvolvidos;
- Propor uma nova técnica, adaptando a técnica SLA para classificação multilabel;
- Desenvolver novos modelos para classificação multilabel com amostras ruidosas.

1.2. Publicações

Ao longo do doutorado foram realizadas 4 publicações. Sendo uma em congresso nacional, uma em congresso internacional e duas publicações em periódico. Ainda está prevista mais uma publicação em um periódico ou congresso internacional.

A primeira publicação foi no *The 23th International Conference on Computational Science and Its Applications* (*ICSSA, Greece 2023*). Sendo apresentado a adaptação da técnica SLA para o cenário multilabel e o modelo Learning by Small Loss Approach Multilabel.

Congresso internacional: (*ICSSA Greece 2023*)
DOI: https://doi.org/10.1007/978-3-031-36805-9_26
Título: Learning by Small Loss Approach Multi-label to Deal with Noisy Labels

A segunda publicação foi no XVI Congresso Brasileiro de Inteligência Computacional (CBIC, Salvador 2023). Sendo apresentado os princípios da técnica RDS.

Congresso nacional: (CBIC, Salvador 2023)
DOI: 10.21528/CBIC2023-135
Título: Aprimorando a Técnica Small Loss Approach para Lidar com Amostras Ruidosas em Modelos Deep Learning

A terceira publicação foi no periódico *Neural Computing and Applications* (Qualis A2). Sendo apresentado os modelos RDS-C e RDS-J, além da aplicação ambiental.

Journal: *Neural Computing and Applications*
DOI: 10.1007/s00521-023-09235-z
Título: Classification of Calcareous Algae Under Noisy Labels.

A quarta publicação foi no periódico *Neural Computing and Applications* (Qualis A2). Sendo apresentado o modelo SLAM-JL e a aplicação do modelo voltada para a área de óleo e gás.

Journal: *Neural Computing and Applications*
DOI: Status Aceito
Título: Multi-Label Noisy Samples in Underwater Inspection from the Oil and Gas Industry.

O modelo RDS-Contrastive está previsto para um congresso internacional ou periódico internacional.

1.3. Contribuições

As principais contribuições dessa tese são:

- Uma nova técnica, chamada de *Recovering Discarded Samples* (RDS), para lidar com amostras ruidosas em modelos DL foi desenvolvida;
- Um modelo de DL utilizando a técnica RDS, chamado de RDS-C, foi desenvolvido para lidar com amostras ruidosas;
- Um modelo de DL, utilizando a técnica RDS, chamado de RDS-J, foi desenvolvido para lidar com amostras ruidosas;
- Um modelo chamado RDS-Contrastive foi desenvolvido para lidar com amostras ruidosas;
- Aplicações dos modelos em um problema real de importância ambiental;
- Adaptação da técnica *Small Loss Approach* para *Multilabel;*
- Desenvolvimento de dois modelos para lidar com amostras ruidosas adequado a problemas de classificação *multilabel;*
- Aplicação dos modelos *multilabel* desenvolvidos em uma base real de inspeções submarinas.

1.4. Organização da Tese

No Capítulo 2 é apresentada a fundamentação teórica para o entendimento do trabalho desenvolvido. No Capítulo 3, é apresentada a nova técnica para lidar com amostras ruidosas RDS, assim como os modelos RDS-C e RDS-J. No Capítulo 4 é apresentado o novo modelo RDS-Contrastive. No Capítulo 5, está apresentada a adaptação da técnica SLA para o cenário multilabel. Também são apresentados os modelos Learning by SLAM e SLAM-JL. No Capítulo 6, os resultados e os estudos de casos abordados nessa tese são apresentados. No Capítulo 7 é apresentado a conclusão da Tese e as possibilidades de trabalhos futuros.

2
Fundamentação Teórica

Esse capítulo apresenta os fundamentos teóricos necessários para a compreensão desta tese. A parte inicial do capítulo apresenta uma introdução ao DL, seguida dos conceitos de modelos supervisionados, semi-supervisionados e não supervisionados. No tópico seguinte é introduzido o conceito de classificação multiclasse e multilabel. Em seguida, o conceito da função de custo e o conceito de amostra ruidosas [6] são apresentados no contexto de DL, assim como suas causas e consequências. Por fim, o capítulo apresenta os modelos do estado da arte para esse cenário.

2.1.
Deep Learning

DL trata-se de uma subcategoria de Machine Learning (ML) [29], desenvolvido a partir do modelo MultiLayer Perceptron (MLP) [30]. Os resultados alcançados por seus modelos atingiram o estado da arte em diversas áreas, como por exemplo, visão computacional [1] e processamento de linguagem natural [31].

Modelos de DL operam com uma abordagem chamada *end-to-end* [32]. Isso significa que o modelo é capaz de aprender representações relevantes e realizar a tarefa atribuída em um passo único, sem auxílio humano ou de algoritmos auxiliares.

Anteriormente ao DL, os modelos clássicos de Inteligência Artificial (IA) [33] [34] realizavam as tarefas em etapas. Tipicamente, utilizavam-se de extração manual [35], que se trata de extrair representações relevantes do problema em questão de forma manual. Na Figura 1 é ilustrado como o fluxo clássico acontecia.

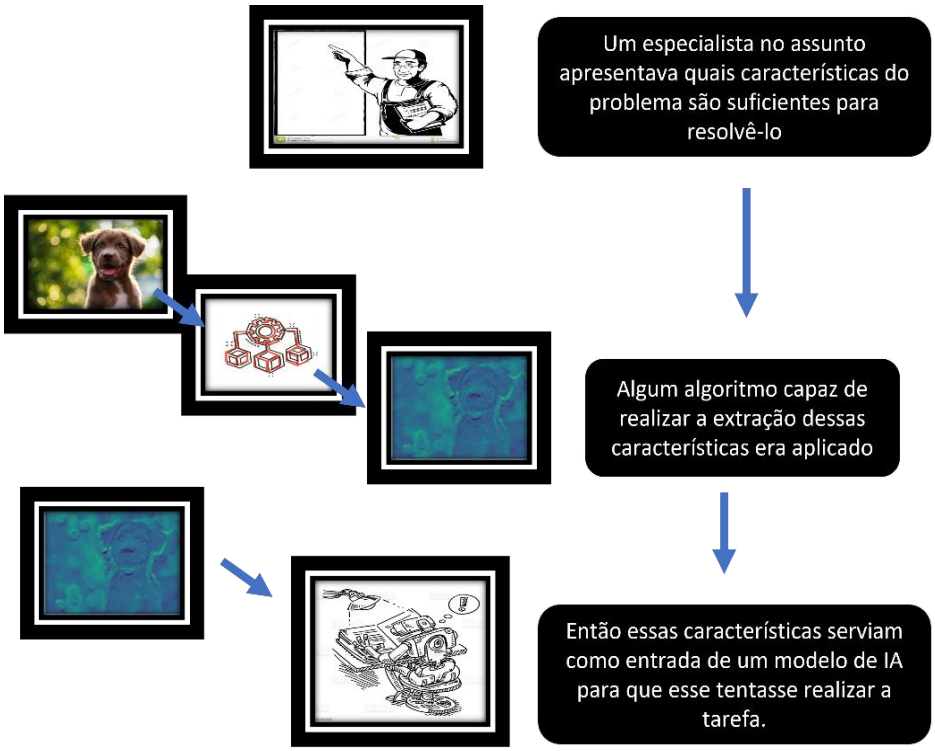


Figura 1 Fluxo clássico de IA

Esse fluxo exigia longos estudos, além de, em muitos casos, exigir o desenvolvimento de técnicas novas para extrair tais características.

Na Figura 2, está representada uma tarefa de classificação de imagens para duas classes - cachorro e gato -. Essa tarefa pode ser realizada em quatro etapas como indicado na figura, classicamente, cada uma dessas etapas exigem um conhecimento prévio do problema e algoritmos específicos. Entretanto, com modelos de DL, todas essas etapas são contempladas no próprio modelo. Ou seja, não é necessário especificar nenhuma regra de como o modelo deve aprender nem extrair características previamente.

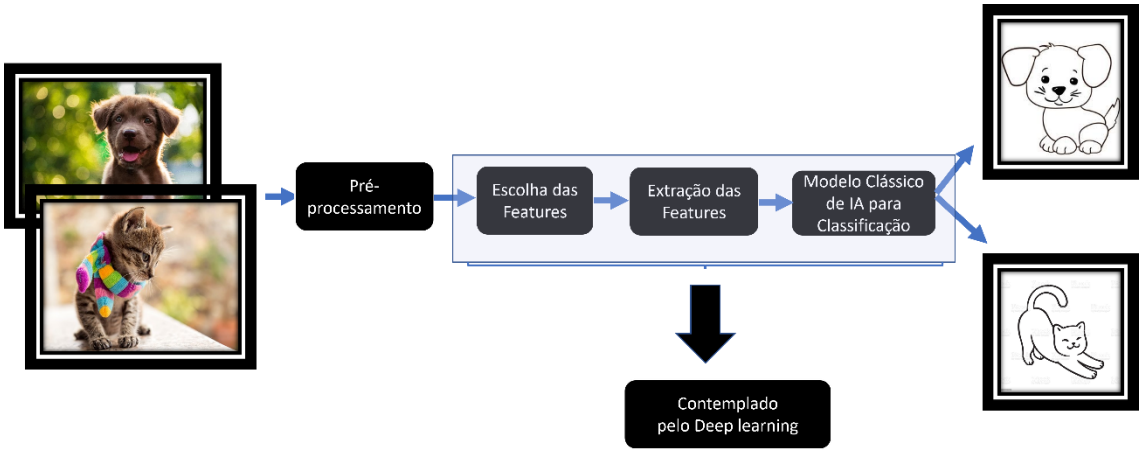


Figura 2 Modelo Clássico

2.2. Modelos Supervisionados, Semi-Supervisionados e Não Supervisionados

Os algoritmos de aprendizado dos modelos de DL podem ser categorizados em quatro principais grupos: supervisionados, semisupervisionados, não supervisionados e auto-supervisionados. Neste trabalho, as soluções propostas para lidar com amostras ruidosas pertencem à categoria semi-supervisionados.

No algoritmo supervisionado, todas as amostras do *dataset* estão rotuladas. Ou seja, a informação da classe presente na amostra é conhecida e utilizada no treinamento. Durante o aprendizado do modelo, essa informação representa a variável objetivo. Dessa forma, dado uma entrada, o modelo deve retornar uma saída equivalente a classe associada da amostra.

No algoritmo não supervisionado [36], as amostras presentes no *dataset* não possuem rótulos associados. Esses modelos extraem informações das amostras e realizam agrupamentos com base nessas informações.

No algoritmo auto-supervisionado [37], o modelo é treinado para aprender representações úteis dos dados através de tarefas auxiliares criadas internamente, sem depender de rótulos externos. Isso é alcançado por meio da formulação de previsões sobre partes ausentes ou relacionadas dos próprios dados, promovendo uma melhor compreensão e representação do espaço de características.

No algoritmo semi-supervisionado, o conjunto de dados utilizado para o treinamento do modelo contém amostras parcialmente rotuladas. Esse algoritmo combina conceitos do aprendizado supervisionado e não supervisionado/auto-supervisionado. Os modelos dessa categoria assumem que os dados rotulados e não rotulados pertencem à mesma distribuição.

2.3. Classificação Multiclasse e Multilabel

Nessa tese, foram desenvolvidos modelos para lidar com amostras ruidosas para classificação multiclasse e multilabel. Na classificação multiclasse, cada amostra do *dataset* está associado exclusivamente a uma classe. Na classificação multilabel, cada amostra do dataset está associado a uma ou mais classe.

Tipicamente, durante o treinamento dos modelos de DL, as classes das amostras são codificadas através do método *one hot encoding*. Esse método mapeia as variáveis representadas por valores inteiros em um vetor de valores binários. Esse vetor possui

dimensão equivalente ao número de classes total do *dataset*, e cada posição do vetor representa uma classe. O valor -*1*- indica presença da classe, enquanto -*0*- representa ausência da classe na amostra. A Figura 3 contém exemplos de representação multiclasse e representação multilabel.

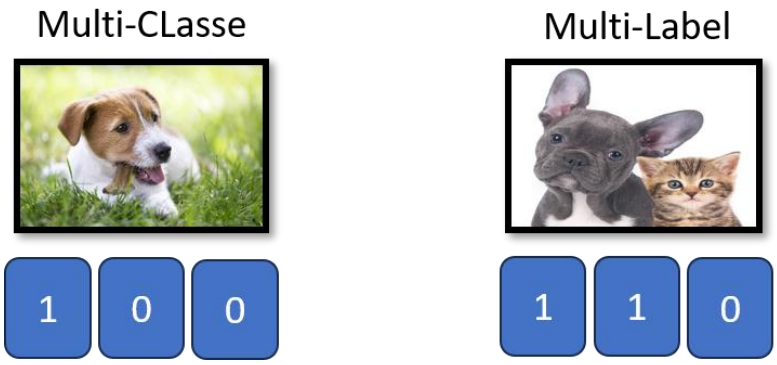


Figura 3 Exemplo de representação one hot encoding para multiclasse e multilabel. Cada posição do vetor representa uma classe. Nesse exemplo, existem 3 classes.

2.4. Função de Custo

As soluções propostas nessa tese para lidar com amostras ruidosas em modelos de DL são baseadas na função de custo. Esta função, mensura a performance de um modelo de DL para um conjunto de dados. Dessa forma, ela quantifica o erro entre a saída do modelo e a saída real esperada, retornando esse erro como um valor real. Existe uma variedade de funções de custo e, dependendo do problema, deve-se minimizar ou maximizar tal função. Em DL, usualmente, se utiliza a CrossEntropy (CE) [38], principalmente em problemas de classificação de imagens, que é o foco dessa proposta.

A função de custo CE é definida pela equação(1):

$$CE(p,q) = -E_{x\sim p}[log(q(x)] \tag{1}$$

Onde *p* e *q* representam duas funções de densidade de probabilidade, sendo *q* a função estimada pelo modelo e *p* a real oriunda dos dados disponíveis para treino. $E_{x\sim p}$ representa o valor esperado, definido por (2):

$$E_{x\sim p}[f(x)] = \int_x p(x)f(x)dx \tag{2}$$

O valor esperado informa o valor médio de um evento representado por uma função *f*(x), quando a sua entrada é uma variável aleatória x de uma distribuição *p* extraída infinitamente. Assim, a CE pode ser escrita como em (3):

$$CE(p,q) = -\int_{x} p(x)log(q(x))dx \tag{3}$$

Para entender o efeito negativo das amostras ruidosas em um modelo de DL, que será apresentado mais a frente, é preciso, antes de tudo, compreender o que de fato ocorre ao utilizar a CE como função de custo. Ao utilizar a CE como função de custo, está sendo utilizado o conceito de "*maximum likelihood estimation*" [39]. O objetivo deste é encontrar a distribuição que melhor representa um conjunto de dados. Dado um conjunto $X=\{x^{(1)},\ldots, x^{(m)}\}$ com uma função densidade de probabilidade dada por $p_{data}(X)$, porém desconhecida, esse conceito busca-se encontrar uma $p'_{data}(X)$ que seja o mais próximo possível de $p_{data}(X)$. Antes da apresentação formal, vamos apresentar a ideia intuitiva do processo na Figura 4:

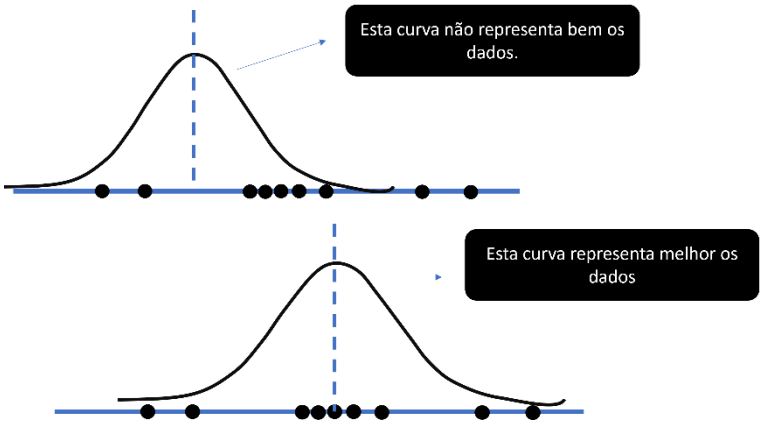


Figura 4 Representação do conceito de maximum likelihood estimation

Na Figura 4, os pontos pretos representam cada instância do conjunto de dados X. Na imagem superior da Figura 4, observa-se que a maioria dos pontos estão com baixa probabilidade. Ao se deslocar a curva para um ponto central mais à direita a maioria dos pontos passa a ter alta probabilidade. Esse processo apresenta a ideia intuitiva do "*maximum likelihood estimation*".

Formalmente, dado um modelo $p_{model}(x;\theta)$ que mapeia uma entrada x para uma estimativa de $p_{data}(x)$. A "*maximum likelihood estimation*" para θ é dado por (4):

$$\begin{aligned}\theta_{ML} &= arg\ max_{\theta}(p_{model}(X,\theta)) \\ &= arg\ max_{\theta}\prod_{i=1}^{m} p_{model}(x^i;\theta)\end{aligned} \tag{4}$$

Onde θ_{ML} pode ser escrito de forma equivalente por (5):

$$\theta_{ML} = arg\ max_{\theta} \sum_{i=1}^{m} log(p_{model}(x^i; \theta)) \quad (5)$$

Pode se escrever então o θ_{ML} em relação a distribuição empírica de *X*, $\hat{p}_{data}(X)$, como em (6) :

$$\theta_{ML} = arg\ max_{\theta} E_{x\sim\hat{p}}[log(p_{model}(x; \theta))\ ] \quad (6)$$

Uma forma de interpretar θ_{ML} é como a dissimilaridade entre a distribuição empírica dos dados $\hat{p}_{data}(X)$ e a distribuição estimada do modelo. Sendo o grau dessa dissimilaridade dado pela Kulback Leibler (KL) *divergence* [40], equação (7). A KL informa o quão similares são duas distribuições distintas:

$$D_{kl}(\hat{p}_{data}||p_{model}) = E_{x\sim\hat{p}}[log(\hat{p}_{data}) - log(p_{model}(x)) \quad (7)$$

O termo $E_{x\sim\hat{p}}[log(\hat{p}_{data})]$ diz respeito somente a entropia dos dados. Dessa forma, quando se treina um modelo para minimizar a KL *divergence*, é preciso apenas minimizar (8):

$$-E_{x\sim\hat{p}}[log(p_{model}(x))] \quad (8)$$

Esse resultado é equivalente a CE apresentado na equação (1). Assim, ao utilizar a CE como função de custo, está se reduzindo a dissimilaridade, ou equivalente, aumentando a similaridade entre a distribuição empírica dos dados e a distribuição aprendida pelo modelo. É claro que o ideal seria aumentar a similaridade com $p_{data}(X)$ real dos dados, porém apenas se tem acesso a distribuição empírica $\hat{p}_{data}(X)$. Destaca-se uma conclusão importante que será utilizada mais a frente ao se analisar o efeito das amostras ruidosas em DL: utilizar a CE como função de custo aumenta a similaridade entre a distribuição empírica dos dados e a distribuição aprendida pelo modelo.

2.5. Redes Neurais Convolucionais

As Redes Neurais Convolucionais (CNNs, do inglês Convolutional Neural Networks) são redes neurais que utilizam a operação matemática de convolução [41]. Ela é particularmente útil em aplicações de DL para visão computacional, pois essa operação conserva uma dependência espacial entre os pixels da imagem de entrada, ao longo da rede, durante o aprendizado [42]. Essa dependência faz sentido de forma

intuitiva, pois um pixel da imagem está mais correlacionado com os seus vizinhos do que com os pixels mais afastados [43].

Cada operação de convolução da CNNs gera um mapa de atributos. A partir deles, é possível ter uma intuição das características extraídas da imagem ao longo da rede. Na Figura 5 esse processo é demostrado.

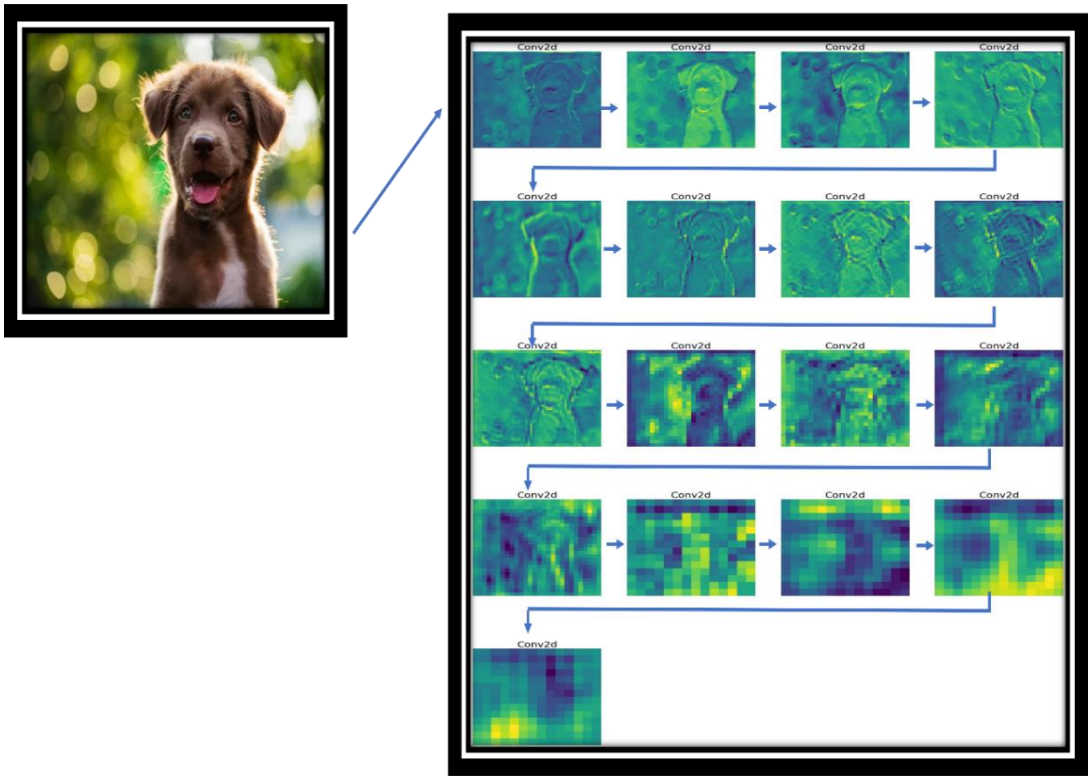


Figura 5 Extração de atributos ao longo de uma rede CNN para a imagem de um cachorro (à esquerda)

Além disso, o número de parâmetros a serem aprendidos pela rede reduz drasticamente com o uso das convoluções [44].

A convolução, como já mencionado, é uma operação matemática com aplicações em áreas como [45][46]. Formalmente, a convolução discreta é dado matematicamente por (9):

$$s(t) = f * k = \sum_{i=-\infty}^{\infty} f(i)k(t-i) \tag{9}$$

Em DL, *f* pode ser a entrada da rede ou um mapa de atributos e *k* é o kernel ou filtro. Durante o processo de aprendizado os parâmetros do kernel são aprendidos pela

rede. A convolução é aplicável também em mais dimensões, por exemplo, para duas, o que pode representar uma imagem de 1 canal, como em (10):

$$s(t,j) = f * k(t,j) = \sum_m \sum_n f(m,n)k(t-m, j-n) \tag{10}$$

Na Figura 6, representamos o processo de convolução. Na imagem, o kernel irá percorrer toda a matriz de entrada gerando uma segunda matriz de saída.

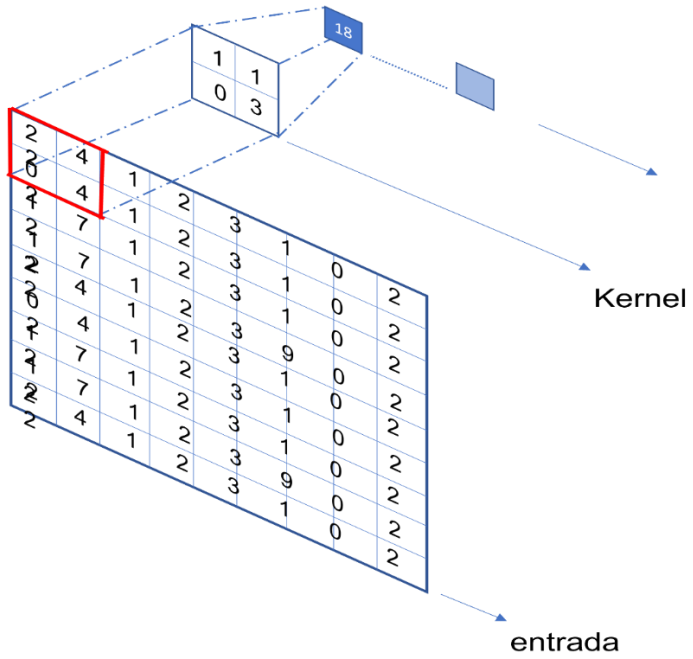


Figura 6 Representação da aplicação de kernel para uma dada entrada. O kernel irá percorrer toda a matriz de entrada gerando uma segunda matriz de saída.

2.6. Deep Learning Com Amostras Ruidosas

Como mencionado anteriormente, para o treinamento de um modelo supervisionado [43] é necessário um conjunto de dados pré-anotados. Ou seja, para cada amostra do conjunto de dados é preciso um rótulo. Por exemplo, para treinar um modelo capaz de classificar fotos de cachorro e de gato precisamos de um *dataset* composto com imagens de cachorros e de gatos, além disso precisamos da informação de qual classe cada imagem pertence. A Figura 7 ilustra a estrutura típica de um *dataset*.

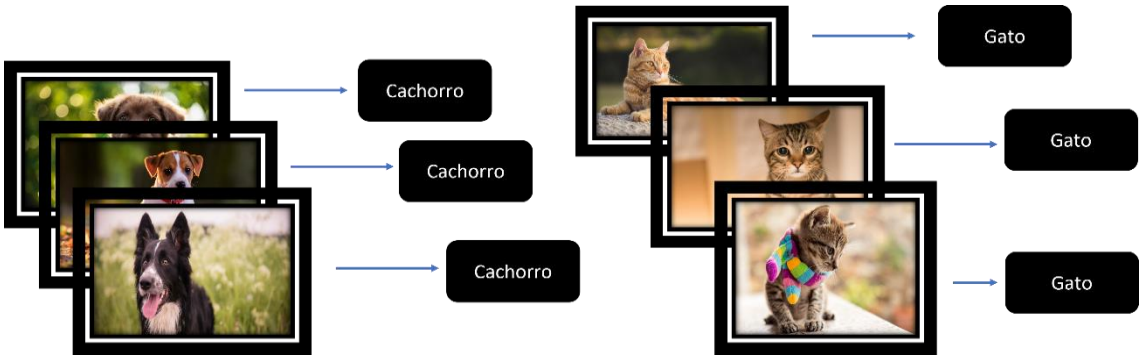


Figura 7 Representação de um dataset simples.

Amostras ruidosas se refere a imagens de um *dataset* que possui amostras com rótulos incorretos, como ilustrado na Figura 8:

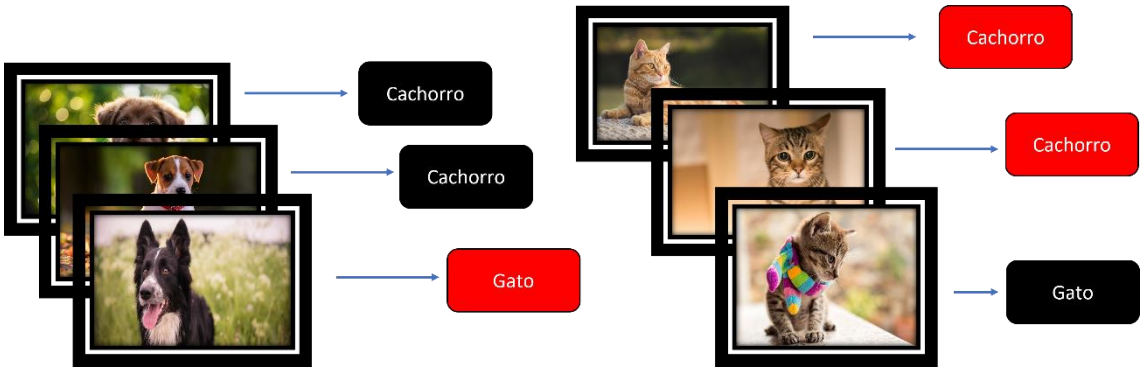


Figura 8 Representação de um dataset com labels errados

Datasets com amostras ruidosas são comuns, até mesmo o *dataset benchmark ImageNet* [47] em suas versões iniciais possuía amostras ruidosas [48], mesmo após amplas revisões. Alguns *datasets* presentes na plataforma Kaggle também são apontados como ruidosos pela comunidade.

Para o treinamento de um modelo de DL ter um bom desempenho é necessário um amplo dataset [42], [49]. Esse dataset precisa ter exemplos diversos de cada classe, que representem bem o cenário real da aplicação do modelo. Isso, em termos práticos, é custoso para ser elaborado e depende da atuação humana. Algumas das causas comuns da presença de amostras ruidosas nos datasets são (i) uso de não especialistas para definição das classes. Em problemas complexos é necessário que pessoas com vasto conhecimento no assunto anotem as imagens, entretanto esses profissionais são caros. Portanto, alguns dataset são elaborados por não especialistas; (ii) processo de anotação exaustivo e repetitivo, o que leva a perda de atenção do anotador; (iii) falhas

no código (*bugs*) para o processo de anotação das amostras; e (iv) anotadores com conceitos diferentes para cada classe, ou classes subjetivas, o que leva à incoerência entre anotadores.

2.6.1. Consequência das Amostras Ruidosas

Modelos de DL treinados com amostras ruidosas apresentam baixo desempenho nos testes e nas aplicações reais, o que é altamente indesejado. Isso é esperado de forma intuitiva, pois a rede está recebendo sinais controversos durante o treinamento. Se fizermos uma analogia com um ser humano recebendo sinais opostos, também é esperado que este não consiga aprender bem determinada tarefa.

O motivo disso decorre do uso da CE como função de custo, abordado de forma aprofundada no tópico 2.4. Destacamos a seguinte conclusão do tópico: utilizar a CE como função de custo aumenta a similaridade entre a distribuição empírica dos dados e a distribuição aprendida pelo modelo. Essa conclusão decorre que utilizar a CE como função de custo é equivalente a reduzir a KL divergence entre duas distribuições. Quando *KL=0* significa que as duas distribuições são iguais.

Dessa forma, dado um dataset $D = \{x_i , y_i\}_{i=1}^{z}$, em que x_i é a *i-ésima* amostra do *dataset* e y_i o rótulo dessa amostra $\in \{1,..M\}$ onde *M* $\in N^+$, e com uma função densidade de probabilidade (FDP) $p_D(Y|X)$, ao treinarmos um modelo com os dados de *D* utilizando a CE, ocorre a redução da similaridade entre $p_{model}(Y|X)$ *e* $p_D(Y|X)$. $p_D(Y|X)$ é uma FDP que descreve um subconjunto da distribuição real descrita pela FDP $p_{real}(Y|X)$, espera-se que $p_D(Y|X)$ seja uma boa representação de $p_{real}(Y|X)$, para que o modelo treinado seja capaz de fazer previsões corretas para novos dados.

Quando D possui amostras ruidosas, nesse caso $D' = \{x_i , y'_i\}_{i=1}^{z}$, sendo y'_i o rótulo contendo amostras ruidosas, com uma função densidade de probabilidade $p_{D'}(Y|X)$, ao realizarmos um treinamento com D' estamos aproximando $p_{model}(Y|X)$ de $p_{D'}(Y|X)$. Como $p_{D'}(Y|X)$ não é uma boa representação de $p_{real}(Y|X)$, $p_{model}(Y|X)$ não será capaz de realizar boas generalizações. Esse processo está representado de forma intuitiva na Figura 9.

Na Figura 9 (A), a linha sólida cinza representa a curva gerada pela função densidade de probabilidade p_{real}. Essa curva é uma representação hipotética na qual se conhecem todos os pares *X* e *Y* de uma dada problemática. O *dataset* não ruidoso,

gera uma curva p_D, representada pela linha pontilhada preta. Essa curva é uma representação correta de um subconjunto de p_{real}. Na figura, p_{real} está representado com dois picos, e as amostras presentes no dataset descrevem apenas o primeiro pico (curva p_D). A curva p_{model} aprendida pelo modelo vai se aproximar nesse caso da curva p_D.

Na Figura 9 (B), a linha sólida cinza representa a curva gerada pela função densidade de probabilidade p_{real}. O *dataset* ruidoso gera uma curva $p_{D'}$, representada pela linha pontilhada preta. Essa curva não é uma representação correta de um subconjunto de p_{real}. A curva p_{model} aprendida pelo modelo vai se aproximar nesse caso da curva $p_{D'}$. Como a curva $p_{D'}$ não é uma boa representação de p_{real}, o modelo não será capaz de realizar previsões corretas.

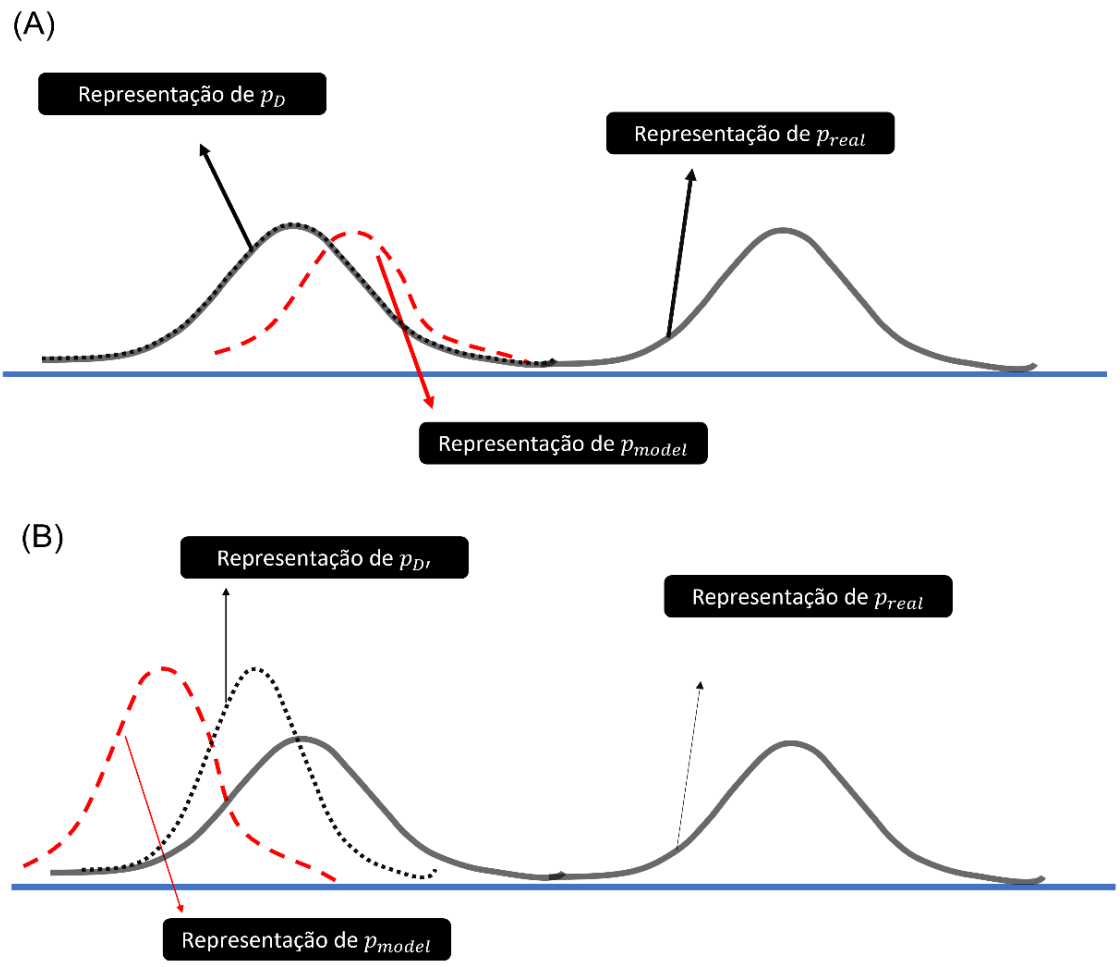


Figura 9 (A) A linha pontilhada em vermelho representa a curva de Pmodel, a curva pontilhada em preto representa a curva dos dados empíricos disponíveis pelo dataset Pd e a curva em cinza Preal representa os dados reais desconhecida. Observa-se que a pd só representa parte da curva real e Pmodel está se aproximando de Pd. (B) Representação de Pmodel se aproximando de Pd' - curva com dados ruidosos -, observa-se que Pd' não é mais uma boa representação de Preal, dessa forma o modelo não terá bom desempenho em suas aplicações. OBS: Essa figura é apenas uma representação intuitiva visual do processo.

2.6.2.
Tipos de Ruídos

Na literatura, usualmente, utiliza-se ruídos inseridos artificialmente em *dataset benchmarks* para realizar comparações entre os diversos modelos do estado da arte. Esses ruídos são inseridos manualmente por *Noise Transition Matrix* [5] Q. $Q_{ij} = P_r[\tilde{y} = j|y = i]$ sendo $\tilde{y}$ o rótulo trocado com o rótulo correto y. A matriz Q tem duas representações *symmetric* e *pairflip* [50], ilustrado na Figura 10. Essa matriz Q possui o parâmetro τ, que indica o ruído inserido, ou seja, a porcentagem de rótulos trocados.

(A)

Symmetric τ = 30			
Classe 1	70 %	10 %	10 %	10 %
Classe 2	10 %	70 %	10 %	10 %
Classe 3	10 %	10 %	70 %	10 %
Classe 4	10 %	10 %	10 %	70 %
	Classe 1	Classe 2	Classe 3	Classe 4

(B)

	Pairflip τ = 45			
Classe 1	55 %	45%		
Classe 2		55 %	45 %	
Classe 3			55 %	45 %
Classe 4	45 %			55 %
	Classe 1	Classe 2	Classe 3	Classe 4

Figura 10 (A) Representação do ruído symmetric para τ=30. (B) Representação do ruído pair flip para τ=45

Em casos de dataset ruidosos reais, a estimativa do ruído presente é tipicamente realizada por amostragem [51].

2.7.
Estado da Arte

Nessa seção, serão explicados os principais modelos do estado da arte para lidar com amostras ruidosas em DL.

2.7.1.
Small Loss Approach

A *Small Loss Approach (SLA)* [5] é uma técnica amplamente utilizada pelos modelos atuais para lidar com amostras ruidosas. Consiste em selecionar as amostras limpas, ou seja, que possuem o label correto durante o treinamento e excluí-las do conjunto treino. Essa situação está ilustrada na Figura 11. A técnica é baseada no fato de que um modelo de DL treinado com menos amostras ruidosas deve ter um desempenho melhor do que um mais ruidoso [52].

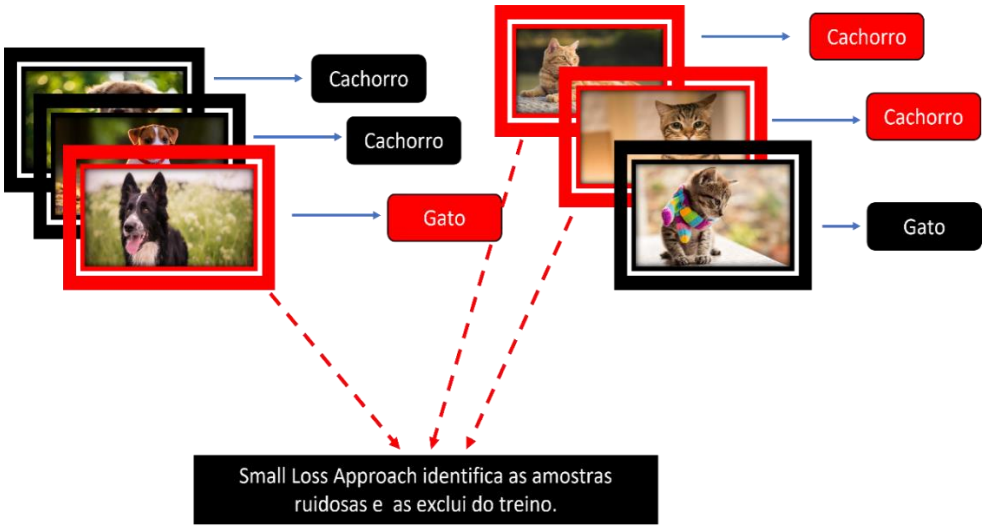


Figura 11 Representação da abordagem da Small Loss Approach

Para selecionar as amostras limpas, a técnica seleciona as amostras que retornarem os menores valores na função de custo CE. Uma vez que essas amostras são selecionas, elas são utilizadas para o treino, ou seja, são utilizadas pelo otimizador SGD (do inglês Stochastic Gradient Descent), enquanto as ruidosas são descartadas, como ilustrado na Figura 12.

A seleção de amostras limpas é realizada dessa forma com base na observação que modelos de DL tendem a aprender as amostras mais fáceis primeiramente e, gradualmente, passam para as amostras mais difíceis [5]. Portanto, é esperado que as amostras ruidosas sejam memorizadas em etapas mais avançadas durante o treino [14]. Em outras palavras, o modelo primeiro aprende as amostras limpas ou fáceis do dataset e depois as amostras ruidosas ou difíceis do dataset. Dessa forma, amostras limpas tendem a gerar custos menores do que as ruidosas nas etapas iniciais do treinamento. O trabalho em [5] aponta uma seleção correta de amostras limpas por esse método na faixa de 75 a 85% para os datasets CIFAR10, CIFAR 100 [53] e MINIST [54] com ruídos de 50%, 45% e 20%.

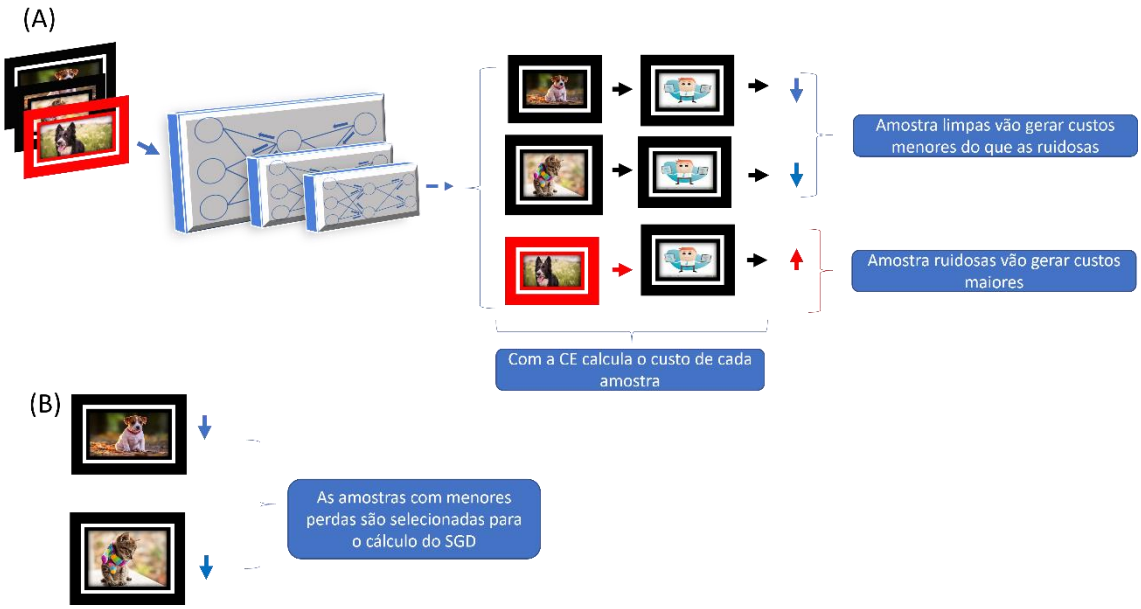


Figura 12 A) Primeira etapa de seleção das amostras limpas utilizando o SLA. (B) Segunda etapa do SLA seleciona as amostras com menores perdas e as utiliza para o cálculo do SGD

Formalmente, dado um mini-batch $B = \{x_i + y'_i\}_{i=1}^{z}$, onde y_i' é o rótulo dessa amostra ∈ $\{1,..M\}$, sendo $M \in N^+$, contendo rótulos ruidosos e limpos, e x_i a ith amostra do mini-batch B; dado um modelo de DL $P(y|x:\theta)$ - um modelo que realiza uma previsão sobre os rótulos *y* para uma entrada *x* com parâmetros θ -; dado R(τ) - uma função que estima quantas amostras serão selecionadas -, e dado uma função de custo *L*, a SLA seleciona as amostras limpas por (11) :

$$B_l = arg\ min_{B:|B|\geq B\ \mathrm{R}(\tau)} \mathrm{L}\ (B\ , P(y|x:\theta)) \tag{11}$$

2.7.2. Co-teaching: Robust Training of Deep Neural Networks with Extremely Noisy Labels

No trabalho [5], emprega-se a técnica SLA utilizando um par de redes neurais, onde uma rede decide quais são as amostras limpas da outra e vice-versa. Durante o treinamento, cada uma das redes realiza previsões no mesmo mini-batch, em seguida, cada rede seleciona as amostras limpas utilizando a SLA. O conjunto de amostras selecionadas por uma rede é utilizada no treinamento da outra rede, conforme ilustrado na Figura 13.

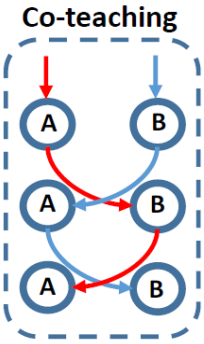


Figura 13 Figura representativa do Co-teaching

O autor destaca que, as duas redes em treinamento possuem habilidades de aprendizado diferentes, i.e., inicializam com pesos diferentes. Com isso, as redes identificam amostras ruidosas distintas, aumentando a identificação de amostras ruidosas totais. A troca de amostras selecionadas reduz gradualmente o fluxo de erros, tornando o processo mais robusto. Ao final do treinamento, é possível utilizar a rede com melhor desempenho.

2.7.3. Decoupling “when to update’’ from “how to update”

O trabalho em [13] apresenta o método *Decoupling* para lidar com amostras ruidosas. Ele desassocia o que o autor chama de quando atualizar (“*when to update*’’) de como atualizar (“*how to update*”). Dessa forma, o autor propõe um método diferente para realizar a atualização dos pesos. Esse método determina quando vale a pena atualizar os pesos ou não (aplicar ou não SGD e Backpropagation).

Isso é realizado treinando dois modelos de DL simultaneamente com os mesmos dados. A etapa de atualização dos pesos ocorre quando as duas redes discordam da previsão de determinada amostra do mini-batch. Assim, de forma similar ao SLA, o modelo seleciona um grupo de amostras para o treinamento. Sendo descartadas as amostras em que os dois modelos realizam previsões iguais. Na Figura 14, o método está apresentado.

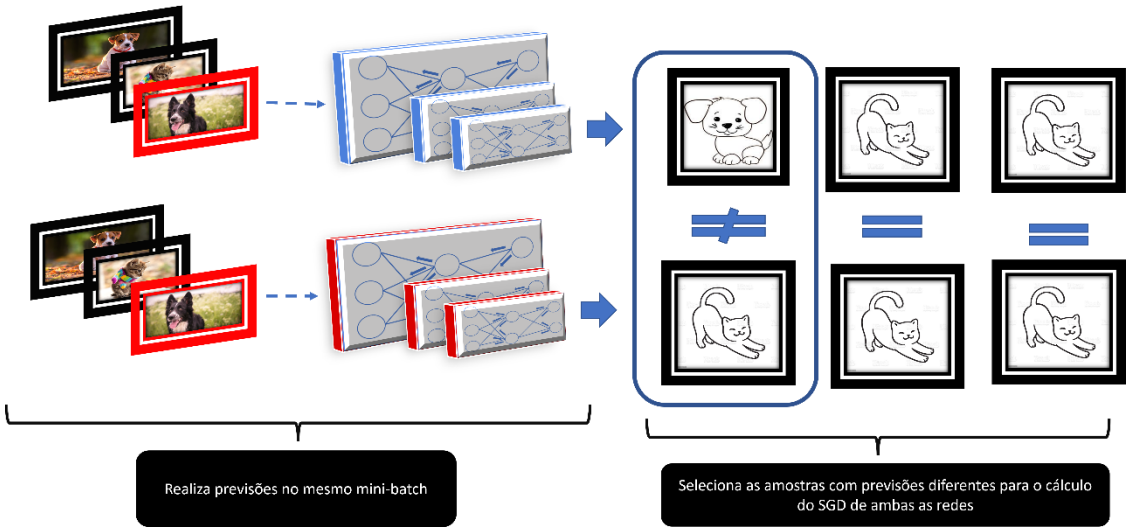


Figura 14 Representação do Decoupling. As amostras em que as redes realizam previsões diferentes são selecionadas para o treino de ambas as redes em cada época.

2.7.4. How does Disagreement Help Generalization against Label Corruption?

O trabalho [14] apresenta o modelo Co-teaching+. O autor incorpora a ideia apresentada pelo *Decoupling* no modelo *Co-teaching*, Figura 15.

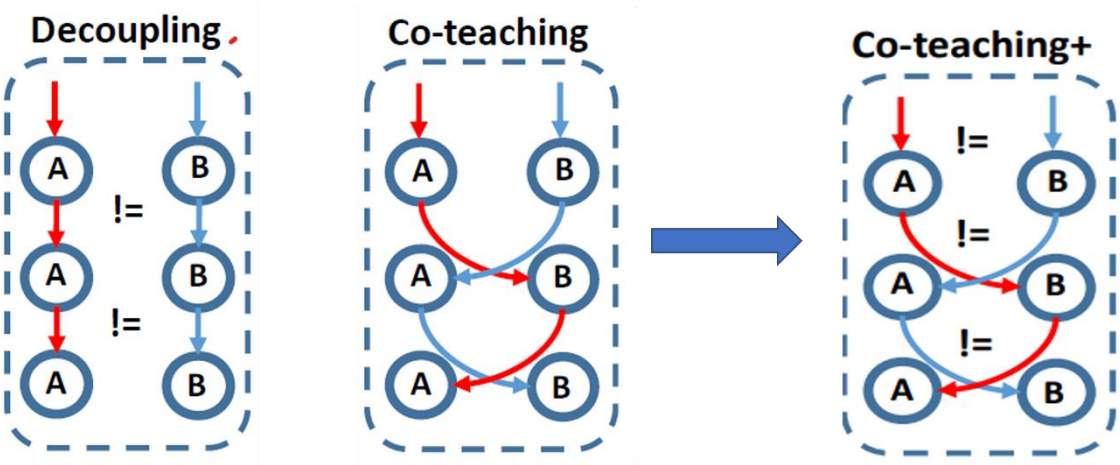


Figura 15 Representação do modelo Co-teaching+

O método *Co-teaching+* emprega um par de redes para realizar previsões no mesmo mini-*batch,* assim como na abordagem do *Co-teaching*. No entanto, apenas as amostras com previsões diferentes, como no *Decoupling*, são mantidas no treino. Após esse primeiro filtro, aplica-se o SLA. Em seguida, como no *Co-teaching*, as amostras selecionadas por uma rede como limpas são utilizadas no treino da outra rede.

O autor alega que manter o par de redes distintas torna o processo mais robusto. Ou seja, os ganhos do modelo *Co-teaching*, apontados como oriundos das diferentes habilidades de aprendizado [5], são aprimorados ao incorporar o *Decoupling* ao processo. A motivação dessa abordagem vem do trabalho [55], que aponta que manter dois classificadores distintos melhora os efeitos conjuntos das amostras em aprendizado semi-supervisionado.

De forma intuitiva, esse processo pode ser visto como uma analogia de dois alunos estudando juntos. Se cada um desses alunos possui conhecimentos diferentes, eles têm a capacidade de se complementar mutuamente.

2.7.5. Combating Noisy Labels by Agreement: A Joint Training Method with Co-Regularization

O trabalho em [15] introduz o modelo *JOCOR*, baseado no princípio do *agreement maximization* [56], onde diferentes modelos devem concordar na maioria das previsões para os rótulos corretos e discordar para a maioria dos rótulos incorretos. Dessa forma, o autor aponta a importância das duas redes utilizadas no *Co-teaching* serem o mais similar possível.

Nesse ponto, é importante ressaltar uma divergência em relação ao que é apontado no trabalho em [14], que destaca a importância de ambas as redes se conservarem distintas. Entretanto, ambos os modelos apresentam ótimos desempenho dependendo do cenário apresentado, como será visto no capítulo de resultados. Assim, destaca-se que este ainda é um ponto de pesquisa em aberto.

Neste modelo, duas redes são treinadas simultaneamente de forma colaborativa. Ambas as redes realizam previsões no mesmo mini-batch, sendo calculado a CE entre os rótulos das amostras e as previsões de cada rede. Em seguida, é calculada a *joint loss*, composta por dois elementos. O primeiro elemento é a CE de cada rede sobre as amostras do *mini-batch*. O segundo elemento é um termo que reduz a divergência entre os dois classificadores, e o autor adotou a *Jensen-Shannon* (JS) *divergence* [43]. O SLA é então aplicado sobre a *joint loss*. As amostras selecionadas como limpas são utilizadas para treinar ambas as redes com o custo calculado pela *joint loss*. Além disso, as redes são siamesas [57], usando o mesmo resultado da *joint loss* para ambas as redes. O processo está ilustrado na Figura 16.

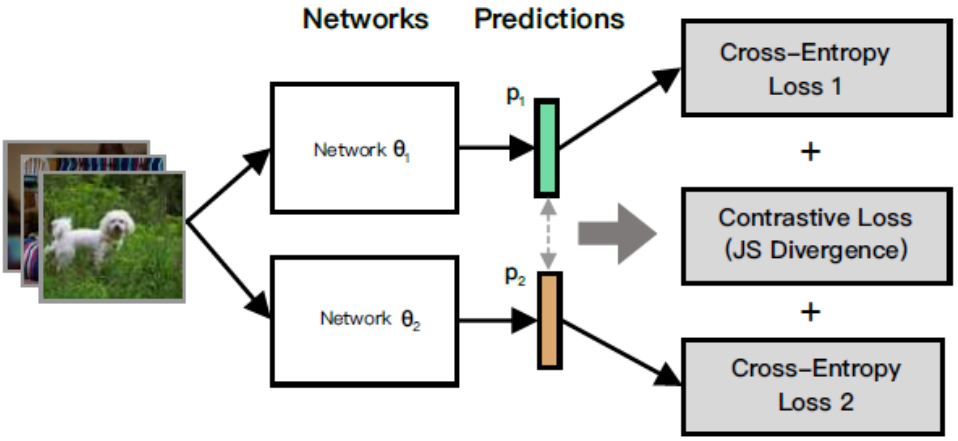


Figura 16 Representação do modelo JOCOR

2.7.6. Self-Adaptive Training: Bridging Supervised and Self-Supervised Learning

O modelo SAT [58] é um modelo semi-supervisionado projetado para lidar com amostras ruidosas. Durante o processo de treinamento, esse modelo faz uma transição de uma abordagem supervisionada para uma abordagem auto-supervisionada [58].

Basicamente, uma rede neural começa com o treinamento de forma supervisionada com o dataset contendo amostras ruidosas. A partir de uma determinada época, o modelo começa a registrar o histórico das previsões realizadas. A partir desse ponto, ele atualiza os rótulos de todas as amostras usando o *Exponencial-Moving-Average (EMA)*. Portanto, a atualização do rótulo de uma amostra x_i com um modelo $P(y|x:\theta)$) - um modelo que realiza uma previsão sobre os rótulos *y* para uma entrada *x* com parâmetros θ -, se dá primeiramente calculando a previsão da rede para a amostras x_i (12):

$$\overrightarrow{\rho\_i} = P(y|\, x_i{:}\,\theta) \tag{12}$$

Então o novo rótulo da amostra será atualizado por (13):

$$\overrightarrow{t\_i} = \alpha\overrightarrow{t\_i} + (1-\alpha)\overrightarrow{\rho\_i} \tag{13}$$

Sendo α o peso atribuído a previsão da rede, e t_i o novo rótulo. Por fim aplica-se a CE levando em conta o rótulot_i para x_i.

A ideia é permitir que o próprio modelo altere os rótulos das amostras gradualmente durante o aprendizado. Espera-se dessa forma que a rede ajuste os rótulos ruidosos. Essa abordagem está baseada na observação que modelos de DL são capazes de extrair informações úteis das amostras mesmo sendo treinados com conjuntos de dados ruidosos [58]. Frente a isso, o autor argumenta que permitir que o modelo ajuste os rótulos antes do processo de memorização das amostras ruidosas pode melhorar o desempenho da rede.

2.7.7. Outros Trabalhos

Além desses trabalhos, existe outra linha de abordagem encontrada na literatura que utiliza *noise transition matrix* [59] [60] [61]. Essa matriz estima a probabilidade de transição do rótulo correto Y da amostra *X* para o rótulo incorreto $\tilde{Y}$.

Em alguns trabalhos a *transition matrix* é dada como previamente conhecida ou por estimativa. Essa estimativa é derivada da observação que uma rede neural treinada com dados ruidosos resulta em um estimador de rótulos ruidosos [62]. Com o conhecimento da matriz transição, é possível realizar correções na função de custo do modelo [63] [62], reduzindo a importância de amostras com alta probabilidade de serem ruidosas.

A limitação desses modelos é dada pela estimação da *transition matrix,* pois a sua estimativa é altamente dependente do conjunto de dados. Além disso, é um processo computacionalmente custoso e o aumento do número de classes torna a estimativa demasiadamente complexa. Dessa forma, na literatura, os modelos baseados na SLA estão com mais relevância em relação aos baseados em *transition* matrix. Assim, os modelos do SOTA são fundamentados no método SLA, pois além de oferecer essas vantagens, apresentam desempenho superior em comparação com os baseados em *transition matrix*.

2.8. Estado da Arte Multilabel

Em contraste com o cenário de amostras ruidosas em um problema multiclasse, onde uma variedade de métodos e *datasets* benchmarks estão estabilizados, o cenário

multilabel encontra-se nos estágios iniciais de estudo na literatura. Essa área ainda carece de *datasets* benchmarks e protocolos bem definidos para comparações dos modelos.

No trabalho em [64], foi proposto utilizar um par de redes professor e aluno para lidar com a problemática. Esse modelo prevê um subconjunto do *dataset* com amostras limpas para inicializar o aprendizado da rede professor, posteriormente, a rede professor seleciona as amostras com rótulos corretos para o treinamento da rede estudante. A necessidade desse subconjunto limita as aplicações desse modelo, pois tal conjunto raramente está disponível em aplicações reais.

Uma outra abordagem foi apresentada recentemente em [65] sem a necessidade de um subconjunto de amostras limpas. Nesse trabalho, foi proposto adaptar diretamente os modelos Jocor e SAT utilizados no cenário multiclasse para o cenário multilabel. As modificações consistem na troca da função de custo CE categórica e da função de ativação *softmax* na última camada da rede por uma CE binária com função de ativação sigmoidal. Embora essa abordagem aumente ligeiramente a performance dos modelos, trata-se de uma típica modificação dos modelos multiclasse para aplicações multilabel em *DL*.

Além disso, no trabalho [66] foi proposto uma expansão de um estimador da *transition matrix* para o cenário multilabel com ruído. Assim como no cenário multiclasse, a eficácia desse algoritmo depende muito da estimativa da matriz de transição.

Os modelos apresentados nessa tese, "*Learning by SLAM*" e "*SLAM by Joint Loss*", foram publicados recentemente e estão apresentados em detalhes no Capítulo 5. Até onde temos conhecimento, nenhum outro trabalho explorou e propôs modificações na técnica SLA para trabalhar no cenário multilabel.

3
Recovering Discarded Samples (RDS)

Nesse trabalho, foi desenvolvida uma nova técnica para lidar com amostras ruidosas, chamada de *Recovering Discarded Samples* (RDS) - ou Recuperando Amostras Descartadas -. Essa técnica é complementar ao SLA e é facilmente adaptado aos modelos atuais que utilizam o SLA. Será apresentado a técnica RDS, bem como dois novos modelos de DL para lidar com amostras ruidosas: (i) RDS-J e (ii) RDS-C, adaptados a partir dos modelos do estado da arte JOCOR e Co-teaching+ respectivamente.

3.1.
Motivação RDS

O SLA exclui do treinamento uma porcentagem de amostras equivalente ao ruído presente no conjunto de dados. Um problema dessa abordagem é que as amostras excluídas do treinamento podem conter atributos importantes da classe da qual ela pertence. Dessa forma, excluindo tais amostras do treinamento, o modelo de DL será impedido de aprender esses atributos importantes, que podem melhorar a generalização do modelo. Além disso, modelos de DL necessitam de uma quantidade representativa de amostras para um bom aprendizado [43].

Inspirado nesse problema foi desenvolvida a abordagem RDS. A ideia principal do RDS é atribuir rótulos novos e corretos para as amostras identificadas como ruidosas pelo SLA. Essas amostras são reintegradas ao conjunto de amostras limpas com os novos rótulos, ou *pseudolabels*, a cada época. A Figura 17 ilustra a ideia intuitiva do RDS. Os resultados mostram que essa abordagem pode melhorar significativamente a performance dos modelos que utilizam SLA.

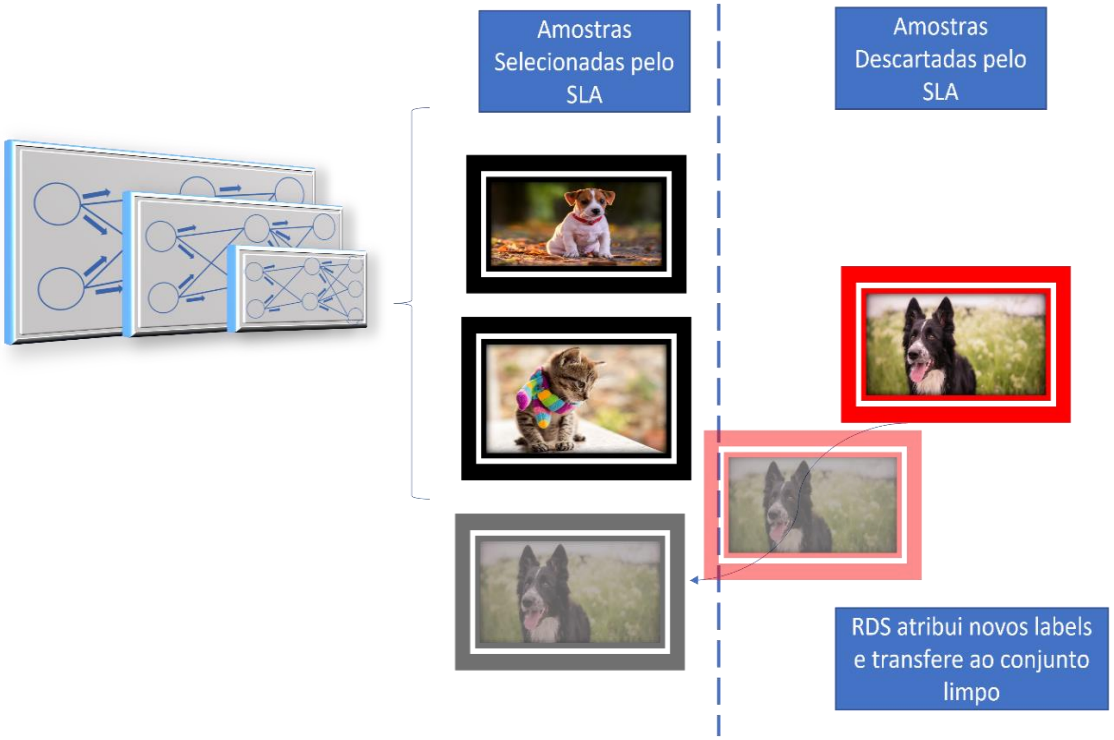


Figura 17 Representação intuitiva do RDS.

3.2. Descrição do RDS

Nesse tópico, será explicado em detalhes o RDS. Antes da apresentação formal, será apresentado de forma intuitiva o processo, detalhando a ideia por trás de cada etapa.

Como explicado anteriormente, o RDS atribui novos rótulos às amostras identificadas como ruidosas pelo SLA, isso é realizado calculando-se *pseudolabels* [17] para essas amostras. Para isso, foi desenvolvido um novo método baseado no *Label Guessing* [67], e este método é chamado de RDS-Label.

O RDS-Label aplica ao conjunto de amostras selecionadas como ruidosas pelo SLA um número *N* de *data augmentation* [68], e.g., *horizontal flips*, *crops*, ajuste de brilho, entre outros, criando-se assim *N* novos conjuntos. Portanto, se *N=2* serão aplicadas duas transformações diferentes às imagens ruidosas, gerando dois novos conjuntos, como ilustra a Figura 18.

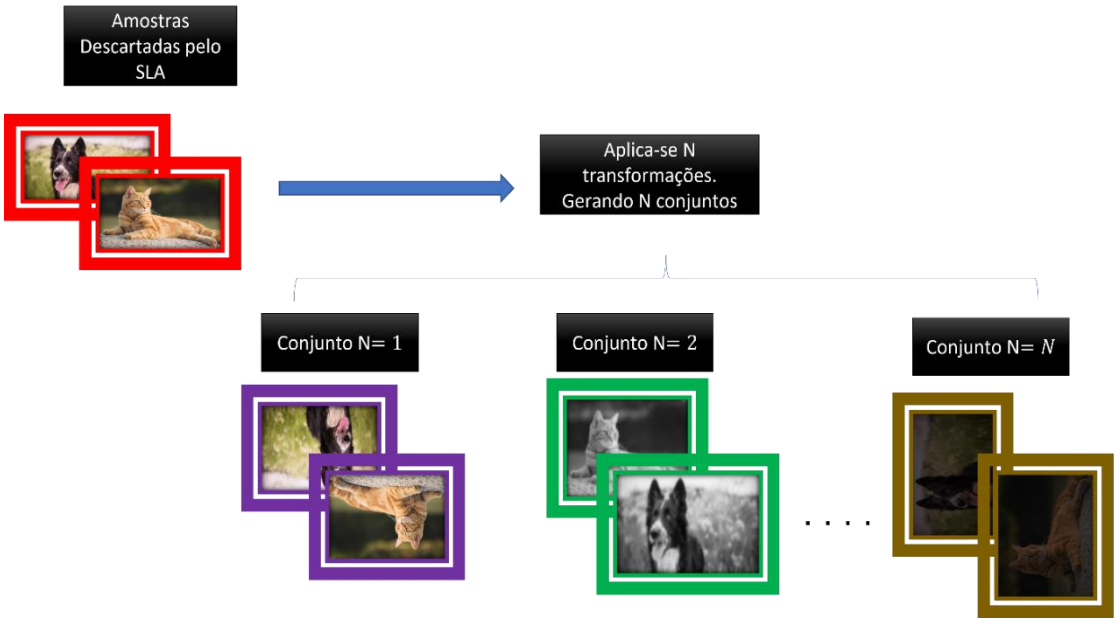


Figura 18 Representação da primeira etapa do RDS-Label

Uma vez gerados esses conjuntos, realiza-se previsões para cada um dos *N* conjuntos com as redes que estão em treinamento. No modelo Jocor e Co-teaching+ utiliza-se duas redes ao longo do treino, dessa forma, cada uma das duas redes realiza previsões em cima de todas as amostras de cada conjunto. Ressalta-se que o número de redes não é restritivo podendo-se utilizar uma ou mais redes para as previsões nessa etapa. Dessa forma, se um modelo *DL* utiliza o SLA, ele pode utilizar o RDS-Label sem restrições. Na Figura 19 esse processo é ilustrado.

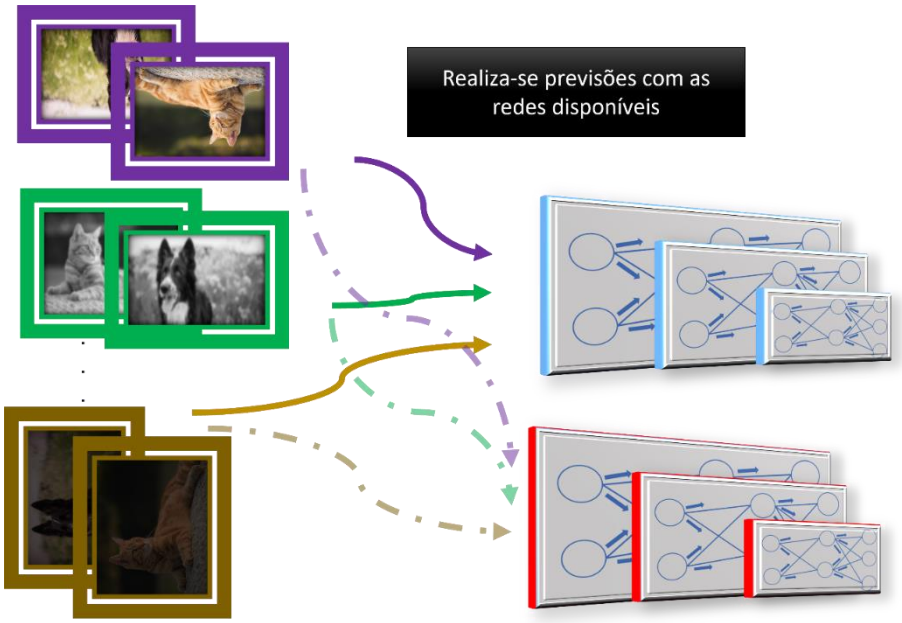


Figura 19 Representação das redes neurais em treinamento realizando previsões para os N conjuntos no RDS-Label

Após realizar as inferências nas amostras dos N conjuntos, salva-se os resultados obtidos após a aplicação da *softmax* para todas as amostras dos N conjuntos. Como ilustrado na Figura 20.

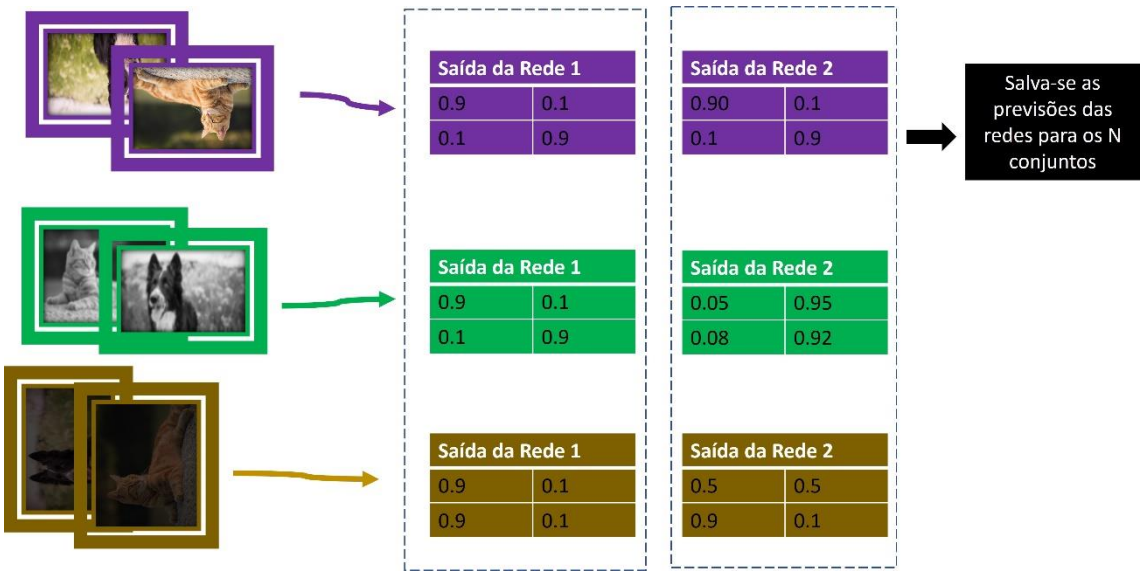


Figura 20 Representação da saída das redes para N=3 conjuntos.

Calcula-se então a previsão média das redes nos N conjuntos por amostra, ou seja, calcula-se a média elemento a elemento. Essa etapa está representada na Figura 21.

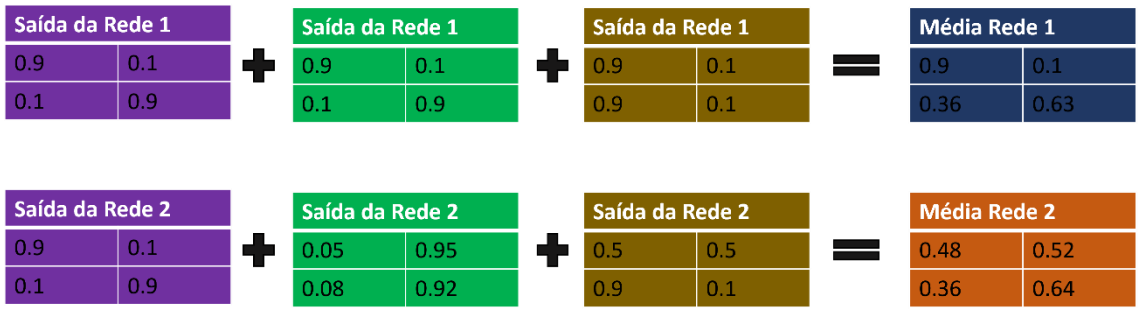


Figura 21 Representação do cálculo da média elemento a elemento da previsão realizada por duas redes, para N=3 conjuntos

De forma intuitiva, nesta etapa, cada rede sugere um novo rótulo para cada amostra do batch. O rótulo é atribuído para a classe com a maior saída média da rede, sendo válido somente se a confiança da rede for superior a um limite pré-estabelecido. Uma vez que a classe é definida, atribui-se a ela confiança máxima -*1*-, enquanto para as outras classes são atribuídas o valor -*0*-.Conforme Figura 22.

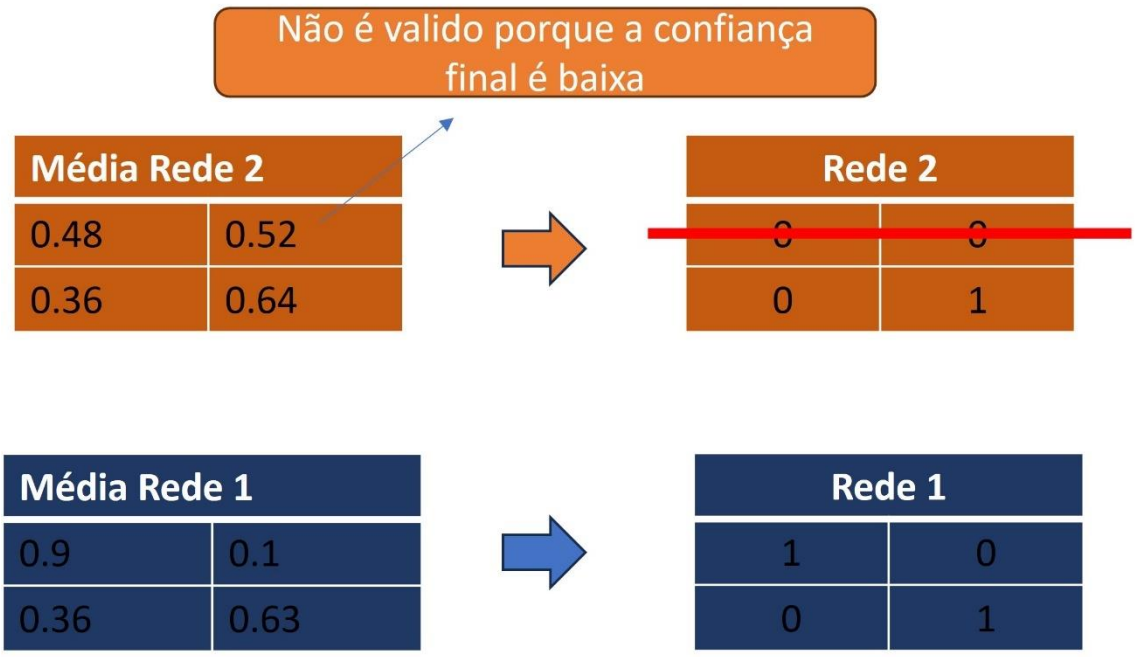


Figura 22 Atribuição como Classe para a maior saída média da rede. Nesse exemplo, considera-se como limiar o valor 0.6 (definido empiricamente). As classes são válidas apenas se a confiança da rede for maior que este valor.

Em seguida, as classes sugeridas por cada rede são comparadas. Sendo considerado um *pseudolabel* da amostra ruidosa quando as classes sugeridas são as mesmas. Esse procedimento está ilustrado na Figura 23.

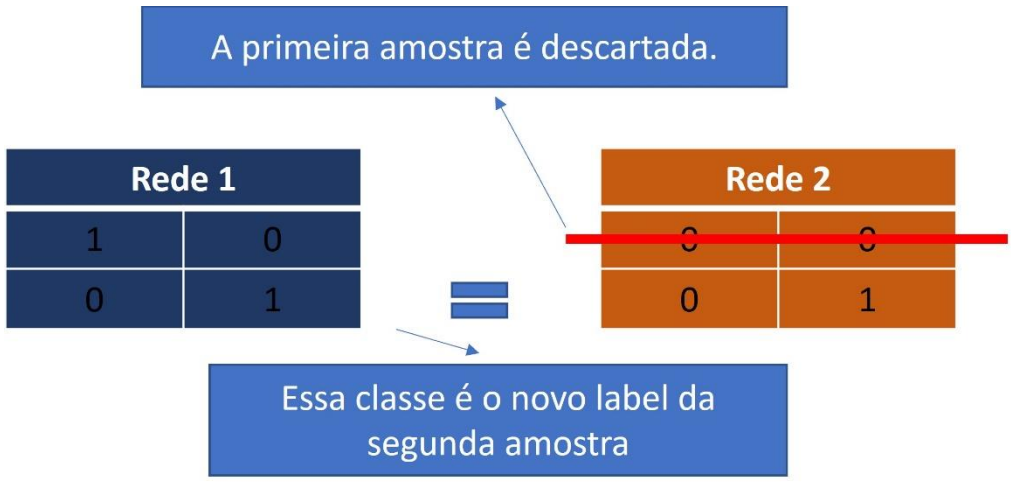


Figura 23 Comparação entre as classes sugeridas por cada rede. Nesse exemplo, como a "rede 2" não teve uma classe válida para a primeira amostra, ela é descartada. Como a classe sugerida da segunda amostra é igual para as duas redes, essa amostra retorna ao treinamento com o novo label sugerido pelas redes.

Assim, as amostras com rótulos incorretos são descartadas e as amostras com rótulos corretos continuam no processo de treino. Quando as redes realizam previsões

diferentes, ou seja, sugerem rótulos diferentes, para uma mesma amostra, essa amostra também é descartada do treinamento.

Por fim, calcula-se a CE para essas amostras levando em conta os *pseudolabels* (essa CE será chamada de RDS-CE explicada em detalhes no tópico 3.2.1). As amostras inicialmente selecionadas como limpas utilizam a CE normalmente. Além disso, o valor da função de custo final do modelo é ponderado, onde a CE tem um peso maior em relação a RDS-CE. Essa etapa está ilustrada na Figura 24. A ideia do RDS-Label parte da seguinte observação: se uma rede realiza a mesma previsão, para a mesma amostra, após aplicar-se variadas transformações, existe uma alta probabilidade de este ser o rótulo correto [52].

Quando se compara os resultados médios da rede, deseja-se aumentar a confiança no processo, pois se as duas redes realizam a mesma previsão, há uma probabilidade maior do rótulo estar correto. Por fim, ao acrescentar um valor limite, deseja-se que a previsão tenha baixa entropia, o que também está associado a aumentar a confiança no rótulo final. No capítulo de resultados será apresentado como o desempenho da rede varia em função deste valor limite.

Todo esse processo visa aumentar a acurácia dos *pseudolabels*. Como será visto no capítulo de resultados, quando o RDS tem uma porcentagem de rótulos válidos maior, a acurácia final do modelo tende a aumentar. Também será visto que a quantidade de amostras com rótulos válidos no processo também interfere no desempenho final do modelo.

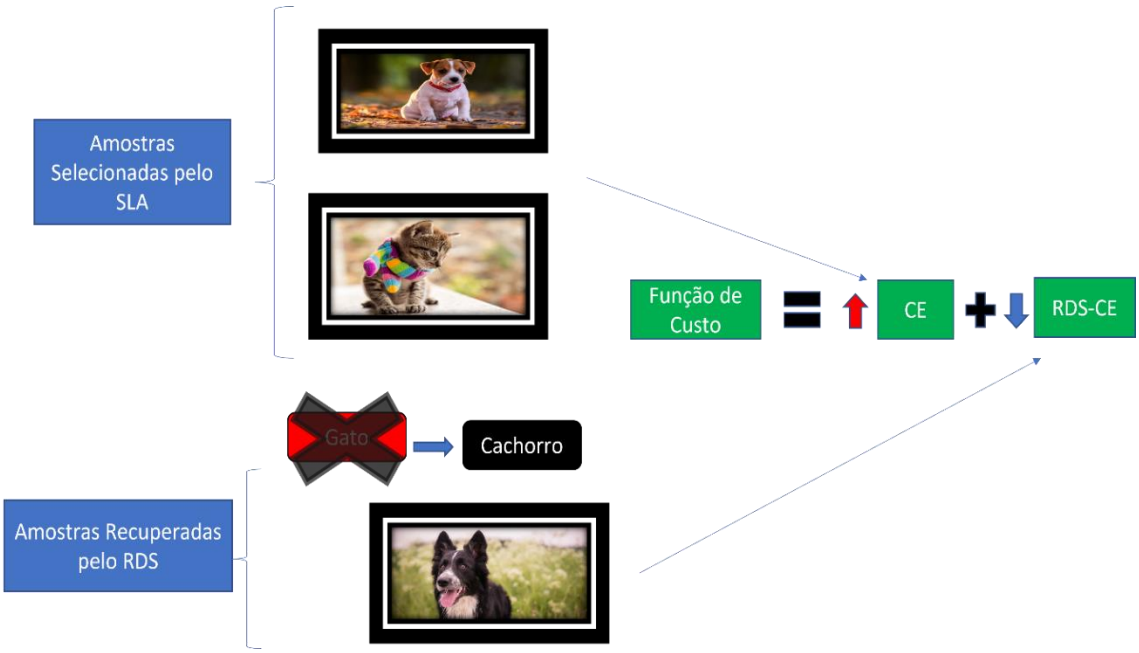


Figura 24 Representação da última etapa do RDS.

3.2.1.
RDS: Descrição Matemática

Formalmente, o RDS pode ser descrito da seguinte forma (14): dado um conjunto de amostras ruidosas selecionadas pelo SLA:

$$B_n = arg\ max_{B:|B|<B\ \mathrm{R}(\tau)} \mathrm{L}\ (B\ , P(y|x:\theta)) \tag{14}$$

Onde L denota uma dada função de custo, R(τ) denota quantas amostras são selecionadas como ruidosas, $P(y|x:\theta)$ denota um modelo com parâmetros θ que para uma dada entrada x retorna a probabilidade desta pertencer a um dos rótulos y, sendo $B = \{x_i, y_i\}_{i=1}^{z}$ o mini-batch inicial contendo amostras limpas e ruidosas, sendo x_i uma amostra de entrada e sendo y_i o rótulo dessa amostra $\in \{1,..M\}$ sendo $M \in N^+$. B_n refere-se a um mini-batch contendo somente as amostras ruidosas $B_n = \{x_i, y'_i\}_{i=1}^{v}$, sendo $y'_i \in \{1,..M\}$ o rótulo identificado como ruidoso pelo SLA. O RDS-Label para a amostra x_i com um número de *data augmentation N* é dado por (15):

$$\bar{y} = \frac{1}{N}\sum_{n=1}^{N} P(y|x_n:\theta) \tag{15}$$

Sendo $\bar{y}$ o pseudolabel a ser assinalado às amostras x_i, x_n é a amostra x_i com uma dada transformação. $\bar{y}$ é considerado um rótulo válido apenas se o resultado tiver baixa entropia, i.e., a saída *softmax* da classe deve ser maior que um valor limite *μ*. Se o rótulo for válido, é atribuído confiança máxima *-1-* para a classe selecionada, e as demais *-0-*. Como a SLA geralmente utiliza dois modelos de *DL*, esse procedimento é realizado com ambas as redes, assim $\bar{y}$ é válido apenas quando ambas as redes retornam a mesma classe $\bar{y}$.

O conjunto de todas as amostras recuperadas pelo RDS será dado por $\bar{A} = (x_k, \bar{y}_k); k \in (1,..k)$, sendo $\bar{y}_k$ o pseudolabel da amostra x_k, sendo esse conjunto denominado aqui de conjunto Relabel total. O RDS-CE será dado por:

$$RDS\_CE = -\frac{\gamma}{K}\sum_{k=1}^{K} \bar{y}_k \log(P(y|x_k:\theta)) \tag{16}$$

Sendo γ um hiperparâmetro utilizado para ponderar a relevância do RDS-CE na função de custo final do modelo.

Por fim, o treinamento de um modelo que utiliza o RDS será dado utilizando dois objetivos, a CE oriunda das amostras previamente selecionadas pelo Small Loss Approach e pela RDS_CE, dessa forma temos (17):

$$CE' = CE + RDS\_CE \tag{17}$$

Sendo equivalente a (18):

$$CE'(P_{D''}, P_{model}) = -\int_{A}^{B} P_{D''}(x) \log P_{model}(x)dx \; - \gamma \int_{B}^{C} P_{D''}(x) \log P_{model}(x)dx \tag{18}$$

Onde $P_{D''}$ é a função densidade de probabilidade dos dados com as alterações de rótulos propostas pelo RDS e as amostras previamente selecionadas pela SLA. P_{model} é a função densidade de probabilidade aprendida pelo modelo ao longo do treino. O intervalo [A, C] é o intervalo do conjunto de dados, e o intervalo [A, B] está contido em [A, C] e representa as amostras selecionadas previamente pela Small Loss Approach, e o intervalo [B, C] está contido em [A, C] e representa as amostras com os rótulos corrigidos pelo processo RDS.

Ressalta-se que o treinamento de um modelo utilizando o objetivo final do RDS, dado por CE' na Equação (18), em condições ideais, equivale a um treinamento livre de amostras ruidosas. Sendo equivalente a realizar um treinamento de um modelo de *DL* dado pela função de custo (19):

$$CE(P_D, P_{model}) = -\int_{A}^{C} P_D(x) \log P_{model}(x)dx \tag{19}$$

Onde que P_D representa a função densidade de probabilidade dos dados livre de ruído. Em condições ideais, estamos nos referindo à possibilidade da técnica Small Loss Approach selecionar todas as amostras com rótulos ruidosos corretamente e o processo RDS-Label atribuir todos os novos rótulos de forma correta. Dessa forma:

$$\begin{gathered} \text{Se } P_{D''}(x) = P_D(x) \;\forall\; x \in [A,B] \\ \text{e} \\ \text{Se } P_{D''}(x) = P_D(x) \;\forall\; x \in [C,B] \end{gathered} \tag{20}$$

Nessas condições a igualdade:

$$\int_{A}^{C} P_D(x) \log P_{model}(x) dx = -\int_{A}^{B} P_{D''}(x) \log P_{model}(x) dx - \int_{B}^{C} P_{D''}(x) \log P_{model}(x) dx \quad (21)$$

É garantida pela propriedade das integrais: Seja f(x) integrável em um intervalo fechado que contenha os pontos [A,B,C] então:

$$\int_{A}^{C} f(x) dx = \int_{A}^{B} f(x) dx + \int_{B}^{C} f(x) dx \quad (22)$$

Sendo assim, em condições ideais:

$$CE(P_D, P_{model}) = CE'(P_{D''}, P_{model}) \quad (23)$$

Ou seja, a técnica RDS se aproxima de um treinamento livre de ruído ao se aproximar das condições ideais com o hiperparâmetro $\gamma = 1$.

3.3. RDS-C

Nessa seção, será apresentado um novo modelo *DL* para lidar com amostras ruidosas, denominado RDS-C. Esse modelo utiliza a técnica RDS sobre o modelo *Co-teaching+.*

No Algoritmo 1, o pseudocódigo do modelo é apresentado, onde D é o conjunto de dados $D = \{x_i, y_i\}_{i=1}^{Z}$, onde x_i é *i-th* instância de D e y_i é o rótulo de $x_i \in \{1, .. M\}$, $w_1(\theta_1)$ é um modelo de *DL* com parâmetros θ_1e $w_2(\theta_2)$ é um modelo de *DL* com parâmetros θ_2.

Do passo 1 ao passo 9 do Algoritmo 1 Modelo RDS-C, foi replicado o modelo *Co-teaching+*. Do passo 10 ao passo 15 foi acrescentado o procedimento RDS, seguido dos passos 16 e 17 da mesma forma do *Co-teaching+,* e finalmente, nos passos 18 e 19 foi acrescentado a função de custo final a RDS-CE ponderada por γ.

No modelo *Co-teaching+,* a função R(τ), que determina quantas amostras são selecionadas pelo SLA, é dada por (24):

$$R(\tau) = min\{\frac{t}{T_k}\tau, \tau\} \quad (24)$$

Onde τ é o valor estimado do ruído presente no conjunto de dados, T_k é um hiperparâmetro, tendo o valor sugerido como 10 em [5], e *t* é a época de treinamento. Essa função retorna o valor mínimo entre as duas entradas.

O Modelo RDS-C é uma demonstração direta de como a técnica RDS pode ser adaptada aos modelos que utilizam o SLA. A ideia básica é continuar com a técnica utilizada pelo modelo *Co-teaching+* no conjunto de amostras limpas e acrescentar ao custo final o valor calculado pela RDS-CE.

Pseudocódigo do Modelo RDS-C

Inputs:	Modelo $w_1(\theta_1)$, Modelo $w_2(\theta_2)$, learning rate η, ruído estimado τ, *threshold μ*, número de épocas $Tmax$, número de transformações *n*, hiperparâmetro γ, *dataset* D; tamanho do *batch* b_z, hiperparâmetro Tk
Passo 1 :	**For** t=1, 2, 3,…$Tmax$:
Passo 2 :	*Shuffle* D
Passo 3 :	Calcule o número de iterações: $\frac{D}{b_z}$
Passo 4 :	**For** i=1,2... $\frac{D}{b_z}$:
Passo 5 :	**Gere** o *mini-batch* D_n de D
Passo 6 :	**Faça** previsões das amostras de D_n com $w_1(\theta_1)$: D_{nw_1}
	Faça previsões das amostras de D_n com $w_2(\theta_2)$: D_{nw_2}
Passo 7 :	**Selecione** as amostras com previsões distintas $D_{nw_1 \neq w_2}$ entre D_{nw_1} e D_{nw_2}
Passo 8 :	**Selecione** R(t)% amostras $D_{n\,small:1}$ do conjunto $D_{nw_1 \neq w_2}$ utilizando SLA com o modelo $w_1(\theta_1)$
Passo 9 :	**Selecione** R(t)% amostras $D_{n\,small:2}$ do conjunto $D_{nw_1 \neq w_2}$ utilizando SLA com o modelo $w_2(\theta_2)$
Passo 10:	**Selecione** as amostras ruidosas com a equação 28 do passo 8. $D_{n+:1}$
Passo 11:	**Selecione** as amostras ruidosas com a equação 28 do passo 9. $D_{n+:2}$
Passo 12:	**Calcule** os pseudolabel s das amostras em $D_{n+:1}$ com a equação 29, usando de augmentations *n* threshold *μ*, gerando o conjunto recuperado $\bar{A}_1$
Passo 13:	**Calcule** os pseudolabel s das amostras em $D_{n+:2}$ com a equação 29, usando de *n* transformações, *threshold μ*, e gerando o conjunto recuperado $\bar{A}_1$
Passo 14:	**Calcule** RDS_CE -equação 30- para as amostras em $\bar{A}_1$:$RDS\_CE_{\bar{A}_1}$
Passo 15:	**Calcule** RDS_CE -equação 30- para as amostras em $\bar{A}_2$:$RDS\_CE_{\bar{A}_2}$
Passo 16:	**Calcule** CE das amostras em $D_{n\,small:1}$: $CE_{D_{n\,small:1}}$
Passo 17:	**Calcule** CE das amostras em $D_{n\,small:2}$: $CE_{D_{n\,small:2}}$
Passo 18:	**Calcule** o custo final para $w_1(\theta_1)$ $L_{final1} = CE_{D_{n\,small:1}} + \gamma RDS\_CE_{\bar{A}_1}$
Passo 19:	**Calcule** o custo final para $w_2(\theta_2)$ $L_{final2} = CE_{D_{n\,small:2}} + \gamma RDS\_CE_{\bar{A}_2}$
Passo 20:	**Atualize** $\theta_1 = \theta_1 + \eta\nabla L_{final1}$
Passo 21:	**Atualize** $\theta_2 = \theta_2 + \eta\nabla L_{final2}$
Passo 22:	**Atualize** $R(\tau)$ pela equação 31
Output :	**Redes** $w_1(\theta_1)$ e $w_2(\theta_2)$

Algoritmo 1 Modelo RDS-C

3.4. RDS-J

Para demostrar a abrangência do RDS, foi proposto um segundo modelo de *DL*, denominado RDS-J, ele utiliza a técnica RDS sobre o modelo Jocor.

Como anteriormente explicado, o modelo JOCOR utiliza a *joint-loss*. Essa função de custo acrescenta a CE, das duas redes em treinamento, a *JS Divergence* para reduzir a divergência entre os dois classificadores. Dado um conjunto de dados $D = \{x_i, y_i\}_{i=1}^{Z}$ onde x_i é a i-th instância de D e y_i é o rótulo ∈ $\{1,..M\}$, para dois modelos de *DL* $w_1(\theta_1)$ e $w_2(\theta_2)$, $p_1 = [p_1^1, p_1^2, \ldots, p_1^M]$ e $p_2 = [p_2^1, p_2^2, \ldots, p_2^M]$ denota a probabilidade prevista pelo modelo para cada classe para a amostra x_i. Então, p_1 e p_2 são a saída da camada softmax das redes $w_1(\theta_1)$ e $w_2(\theta_2)$, respectivamente. A *JS Divergence* foi implementada usando a *KL Divergence* para $w_1(\theta_1)$ e $w_2(\theta_2)$ sobre as amostras em D:

$$J_s(w_{1D}||w_{2D}) = KL(w_{1D}||w_{2D}) + KL(w_{2D}||w_{1D}) \tag{25}$$

Onde:

$$KL(w_{1D}||w_{2D}) = \sum_{i=1}^{Z}\sum_{m=1}^{M} p_1^m(x_i) log \frac{p_1^m(x_i)}{p_2^m(x_i)} \tag{26}$$

E:

$$KL(w_{2D}||w_{1D}) = \sum_{i=1}^{Z}\sum_{m=1}^{M} p_2^m(x_i) log \frac{p_2^m(x_i)}{p_1^m(x_i)} \tag{27}$$

Pseudocódigo do Modelo RDS-J

Inputs:	Modelo $w_1(\theta_1)$, Modelo $w_2(\theta_2)$, *learning rate* η, ruído estimado τ, *threshold* μ, número de épocas $Tmax$, número de transformações *n*, hiperparâmetro γ, *dataset* D; tamanho do batch b_z, hiperparâmetro Tk
Passo 1:	**For** t=1, 2, 3,…$Tmax$:
Passo 2:	*Shuffle* D
Passo 3:	Calcule o número de iterações: $\frac{D}{b_z}$
Passo 4:	**For** i=1,2... $\frac{D}{b_z}$:
Passo 5:	**Gere** o *mini-batch* D_n de D
Passo 6:	**Calcule** a CE sobre $D_n$com $w_1(\theta_1)$:$l_{ce:1}$
Passo 7:	**Calcule** a CE sobre $D_n$com $w_2(\theta_2)$:$l_{ce:2}$
Passo 8:	**Calcule** a $J_s(w_{1D}\|\|w_{2D}) - equação\ 32 - \ entre\ w_1(\theta_1)$ e $w_2(\theta_2)$ sobre D_n l_{js}
Passo 9:	**Calcule** a **Joint Los** $l_{joint} = l_{ce:1} + l_{ce:2} + l_{js}$
Passo10:	**Selecione** R(t)% Amostras D_{nsmall} utilizando SLA sobre ***Joint Loss***

Passo11: **Selecione** as amostras ruidosas com a equação 28 do passo 10. D_{n+}

Passo12: **Calcule** os pseudolabels das amostras D_{n+} com a equação 29 usando *n* transformações, *threshold* μ, gerando o conjunto recuperado $\bar{A}_1$

Passo 13: **Calcule** RDS-CE sobre $\bar{A}_1$ com $w_1(\theta_1)$ $RDS\_CE_{w_1}$

Passo 14: **Calcule** RDS-CE sobre $\bar{A}_1$ com $w_2(\theta_2)$ $RDS\_CE_{w_2}$

Passo 15: **Calcule** a RDS-CE média-$RDS\_CE_{w_{media}}$- entre $RDS\_CE_{w_1}$ e $RDS\_CE_{w_2}$

Passo 16: **Calcule** a **Joint Los** $l_{joint:small}$ para as amostras D_{nsmall}

Passo 17: **Calcule** a *loss* final $l_{final} = (1-\gamma)l_{joint:small} + \gamma RDS\_CE_{w_{media}}$

Passo 18: **Atualize** $\theta_1 = \theta_1 + \eta\nabla L_{final}$

Passo 19: **Atualize** $\theta_2 = \theta_2 + \eta\nabla L_{final}$

Passo 20: **Atualize** $R(\tau)$ pela equação 31

Output : Modelo $w_1(\theta_1)$, Modelo $w_2(\theta_2)$,

Algoritmo 2 Modelo RDS-J

No Algoritmo 2, está ilustrado o pseudocódigo do RDS-J. Os passos 1 ao 10 são os passos originais do modelo JOCOR, do passo 11 ao 15 foi introduzido o procedimento referente ao RDS. No passo 16 calcula-se a *Joint Loss* como no JOCOR, e no passo 17 soma-se a RDS-CE à *Joint Loss*.

Assim como o RDS-C, o RDS-J é uma demonstração de como a técnica RDS pode ser adaptada aos modelos que utilizam o SLA. Basicamente, no RDS-J aplica-se o modelo JOCOR e acrescenta-se o custo calculado pela RDS-CE. Nesse modelo calcula-se a RDS-CE média entre as duas redes em treinamento, pois o modelo é treinado com um sistema de redes siamesas.

4
Modelo RDS-Contrastive

Nesse trabalho foi desenvolvido um terceiro modelo para lidar com amostras ruidosas utilizando o SLA e RDS, chamado de RDS-Contrastive. Esse modelo utiliza uma abordagem de treinamento auto-supervisionado [69].

4.1.
Motivação

A partir dos resultados dos modelos RDS-C e RDS-J foi observado que uma maior acurácia nos *pseudolabels* no processo do RDS-Label resulta em um modelo com um desempenho melhor. Esse comportamento é esperado, pois um modelo de DL treinado com menos amostras ruidosas tem uma performance melhor do que um treinado com mais amostras ruidosas [52]. Frente a isso, buscaram-se técnicas para melhorar o desempenho do RDS-Label. Nesse ponto, destaca-se uma possibilidade de realização de futuros trabalhos de pesquisa como a busca por técnicas que levem ao melhor desempenho do RDS-Label.

Uma segunda motivação para o desenvolvimento desse modelo foi a observação do trabalho de L. Huang [58] que afirma que para treinar um modelo de DL com um conjunto de dados 100% ruidoso é o mesmo que não ter nenhum rótulo. Nesse caso, é necessário realizar um treinamento auto-supervisionado. Dessa forma, espera-se que essas técnicas também possam proporcionar ganhos em cenários com um menor grau de ruído.

A partir dessas duas observações, o modelo proposto, denominado *RDS-Contrastive*, utiliza uma técnica adaptada do trabalho em [18] para lidar com amostras ruidosas. Essa técnica vem sendo amplamente utilizada na área de aprendizado auto-supervisionado [70], [71], [72] com ganhos significativos para a área. A abordagem deste modelo combina as técnicas SLA e RDS com a técnica para aprendizado auto-supervisionados apresentado em [18].

4.2.
RDS-Contrastive em Detalhes

Assim como no tópico 3.2, primeiro apresentaremos o passo a passo intuitivo do modelo RDS-Contrastive. Inicialmente, explicaremos a ideia por trás de cada etapa, em seguida descreveremos o modelo matematicamente.

O modelo *RDS-Contrastive* conserva a estrutura de treinamento apresentado em [5], utilizando um par de redes neurais treinadas com o mesmo conjunto de dados simultaneamente. Conforme ilustrado na Figura 25.

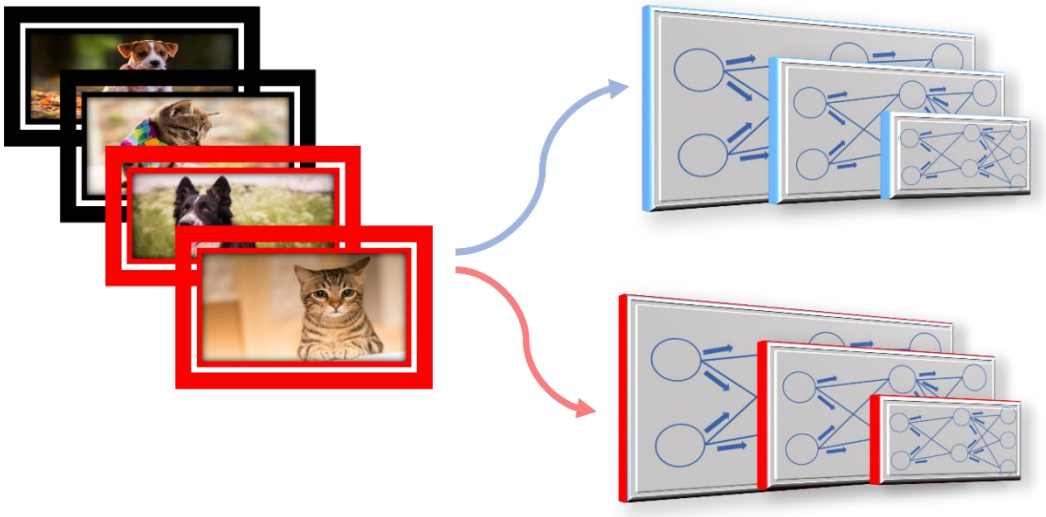


Figura 25 Representação de duas redes treinando sobre os mesmos dados no modelo RDS-Contrastive

Assim como em [5], cada uma das redes decide quais são as amostras limpas da outra através do SLA, conforme ilustrado na Figura 26.

No conjunto de amostras selecionadas como ruidosas pelo SLA aplica-se o RDS-Label, conforme apresentado na Figura 27. Destaca-se que as amostras selecionadas como ruidosas pela primeira rede são utilizadas no procedimento do RDS na própria rede, diferente da inversão utilizada nas amostras limpas da primeira etapa. Até essa etapa o modelo é simplesmente uma aplicação do RDS no modelo [5], sendo a função de custo composta até então por dois termos: a CE e a RDS-CE.

Figura 26 Representação das amostras limpas selecionada por SLA por cada rede sendo utilizada na outra rede.

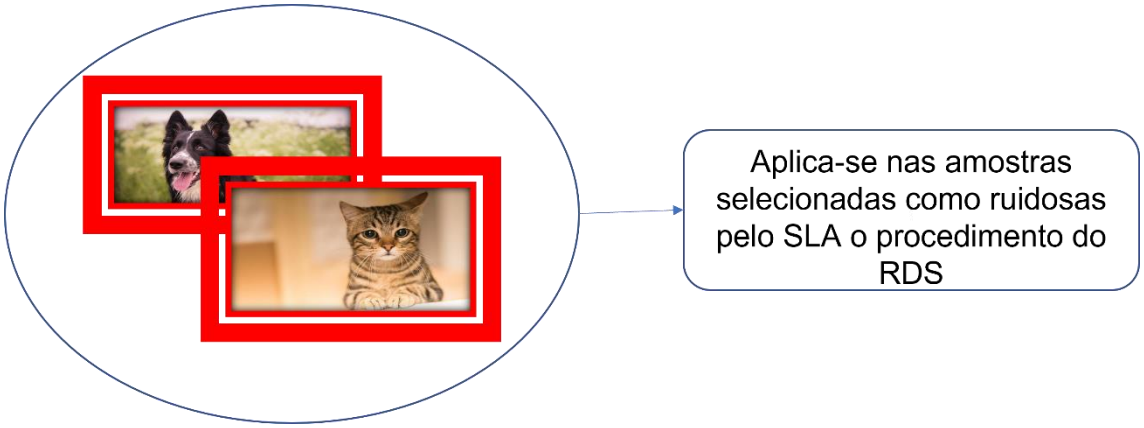


Figura 27 Representação da aplicação do procedimento RDS nas amostras descartadas por ambas as redes

Um terceiro objetivo é acrescentado ao modelo, sendo este chamado de custo *contrastive*. Como adiantando, esse custo é baseado na SimCLR do trabalho em [18]. A SimCLR aprende representações no espaço latente das amostras colocando cada uma das redes em treinamento para gerar a mesma saída - *maximizing agreement* - após aplicação de diferentes transformações para uma mesma amostra. Como ilustra a Figura 28.

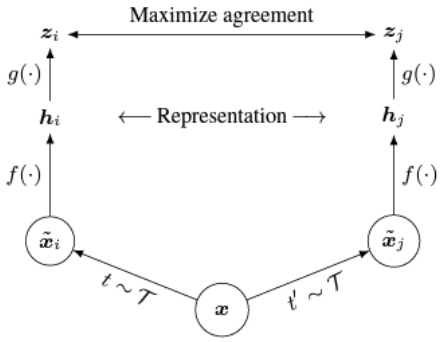


Figura 28 Representação do SimCLR – figura adaptada de [16]

Na Figura 28, aplica-se a uma amostra *x* uma transformação *t* da família de transformações $t \sim \tau$, gerando a amostra transformada $\tilde{x}_i$, uma segunda transformação *t'* é aplicada à mesma amostra *x,* gerando a amostra transformada $\tilde{x}_j$. Uma rede extratora de atributos *f(.)* é aplicada a ambas as amostras transformadas gerando uma representação latente h_i e h_j , aplica-se então uma função $g(.)$ em h_i e h_j gerando a saída z_i e z_j . O objetivo do treino é fazer com que as saídas z_i e z_j sejam iguais, e para isso pode-se utilizar a CE como função de custo. O procedimento do SimCLR ilustrado na Figura 28 permite a rede *f(.)* aprender representações das amostras no espaço latente, permitindo separar amostras distintas aplicando-se posteriormente um algoritmo como o *K-means* [73].

No modelo *RDS-Constrative,* utiliza-se um procedimento similar ao SimCLR visando aumentar o desempenho da etapa auto-supervisionada do modelo, ou seja, o RDS-Label.

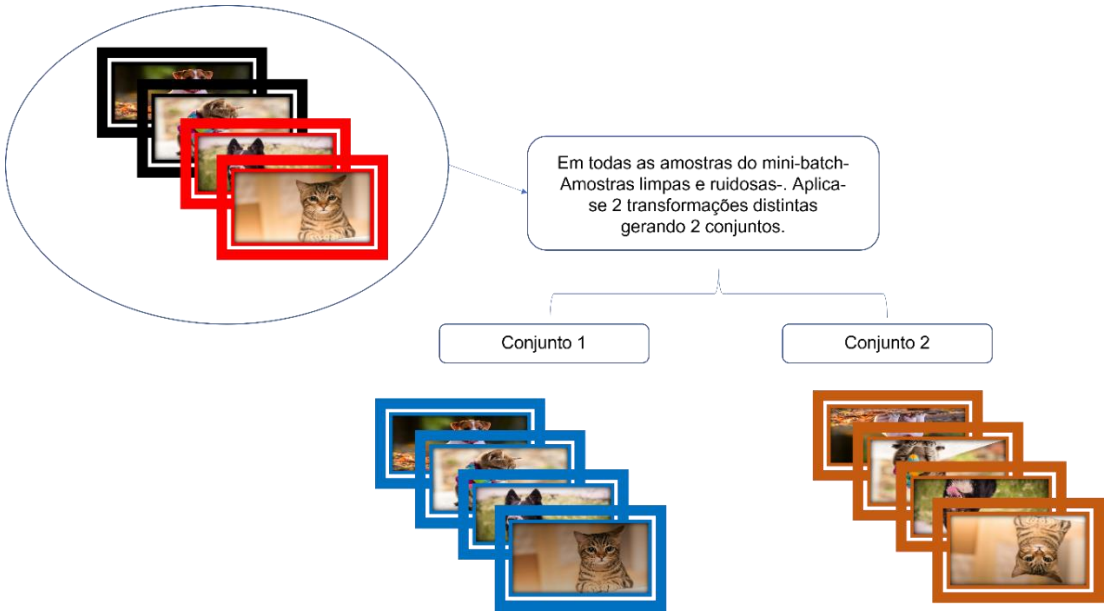


Figura 29 Aplicação de duas transformações distintas no mesmo conjunto de dados

Aplica-se duas transformações distintas em todo o conjunto de amostras do *mini-batch*, isso inclui as amostras ruidosas e limpas. Esse procedimento gera dois conjuntos distintos das mesmas amostras, como ilustrado na Figura 29.

Com ambas as redes, é realizado previsões com os dois conjuntos, conforme ilustrado na Figura 30. Em seguida, é aplicado a função *softmax* com temperatura [74]. Aplica-se a CE entre as previsões das redes. Sendo calculado a CE entre a primeira transformação da rede 1 com a segunda transformação da rede 2, e a CE entre a primeira transformação da rede 2 com a segunda transformação da rede 1. Por fim, calcula-se a média dessas duas CE, conforme ilustrado na Figura 31. Essa CE média é o termo *contrastive*.

A função de custo do modelo é composta por três termos a CE, RDS-CE e o termo *contrastive*. No modelo RDS-Contrastive, visando aumentar o desempenho do RDS-Label, realiza-se uma pequena alteração no processo RDS-Label: no cálculo da RDS-CE acrescenta-se pesos baseados na confiança da rede para o *pseudolabel* de cada amostra. Ou seja, após a definição do *pseudolabel* pela Equação (15), guarda-se o valor encontrado para a classe majoritária, esse valor será o grau de confiança deste *pseudolabel*.

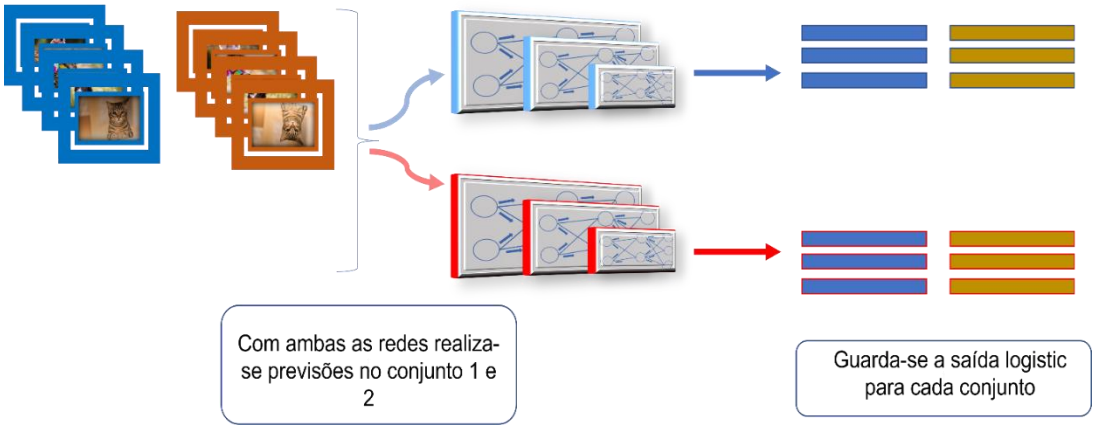


Figura 30 Previsões dos dois conjuntos com ambas as redes em treinamento

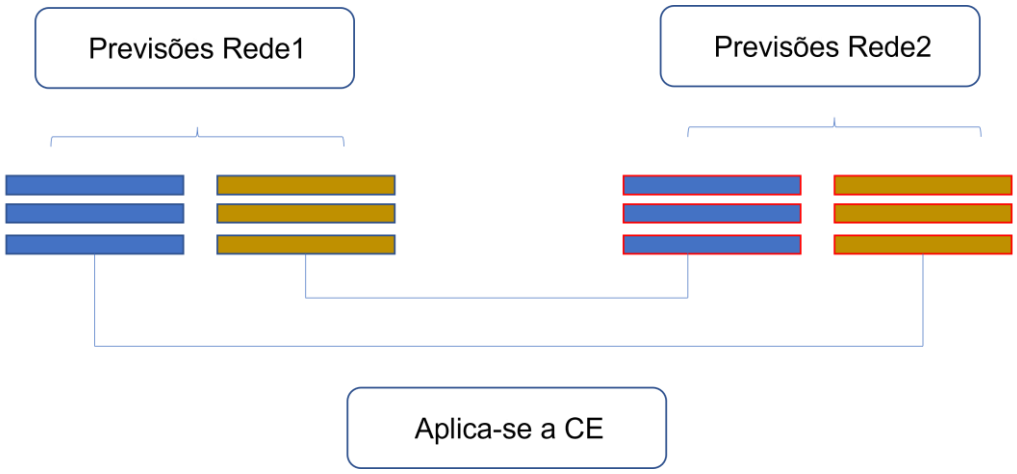


Figura 31 Aplicação da CE entre as previsões das redes

Após calcular o grau de confiança de todos os *pseudolabels*, realiza-se uma normalização entre todos os graus de confiança. Assim, cada termo da RDS-CE será ponderado pelo grau de confiança normalizado correspondente.

4.3. RDS-Contrastive Descrição Matemática

Dado um mini-batch $B = \{x_i, y_i\}_{i=1}^{z}$ contendo amostras limpas e ruidosas, sendo x_i uma amostra de entrada e sendo y_i o rotulo da amostra x_i, onde $y_i \in \{1, .. M\}$ sendo $M \in N^+$. Dado duas funções *T(x)* e *T'(x)* que aplica transformações distintas na entrada *x*, a amostra x_i transformada será dada por (28):

$$x_i{}^T = T(x_i) \tag{28}$$

E a segunda transformação (29):

$$x_i{}^{T'} = T'(x_i) \tag{29}$$

O conjunto contendo todas as amostras com transformações *T* será dado por $B^T = \{x_i{}^T\}_{i=1}^{z}$ e com as transformações T', $B^{T'} = \{x_i{}^{T'}\}_{i=1}^{z}$.

Dado dois modelos $P(y|x:\theta)$ e $P'(y|x:\theta')$, sendo estes dois modelos com parâmetros θ e θ', respectivamente, que para uma dada entrada x retorna a saída logistic. A *softmax* com temperatura para amostra x_i e o modelo $P(y|x:\theta)$ é dado por (30):

$$s_{oftemP}(x^i) = \frac{e^{P(y|x^i:\theta)/t_{emp}}}{\sum_{k=1}^{M} e^{P(y|x^k:\theta)/t_{emp}}} \quad 30$$

Onde t_{emp} é o termo que controla a suavidade da s_{oftemP}. Para o modelo $P'(y|x:\theta')$ temos de forma análoga (31):

$$s_{oftemP'}(x^i) = \frac{e^{P'(y|x^i:\theta')/t_{emp}}}{\sum_{k=1}^{M} e^{P'(y|x^i:\theta')/t_{emp}}} \quad 31$$

A primeira parcela do termo *contrastive* será dado por (32):

$$CE_{contrastive\_1} = -\sum_{i=1}^{Z} s_{oftemP}({x_i}^T)\, log(s_{oftemP'}({x_i}^{T'})) \quad 32$$

A segunda parcela será (33):

$$CE_{contrastive\_2} = -\sum_{i=1}^{Z} s_{oftemP'}({x_i}^{T'}) log(s_{oftemP}({x_i}^T)) \quad 33$$

O termo *contrastive* final será (34):

$$CE_{contrastive\_f} = (\frac{CE_{contrastive\_1} + CE_{contrastive\_2}}{2}) \quad 34$$

O grau de confiança do pseudolabel da amostra x_i será estimado por (35):

$$w = arg\ max(\bar{y}) \quad 35$$

Sendo $\bar{y}$ a saída da equação (15).

Dessa forma, sendo o conjunto de todas as amostras recuperadas pelo RDS dado por $\bar{A} = (x_k, \bar{y}_k, w_k); k \in (1,..k)$, sendo $\bar{y}_k$ o *pseudolabel* da amostra x_k e w_k o grau de confiança do *pseudolabel*. O RDS_{CE_Pesos} no modelo RDS-Contrastive será dado por 36:

$$RDS_{CE\_Pesos} = -\frac{\gamma}{K}\sum_{k=1}^{K} \bar{y}_k\, logP(y|x_k:\theta)\, \frac{w_k}{\sum_i^K w_i} \quad 36$$

Sendo γ um hiperparâmetro utilizado para ponderar a relevância do RDS_{CE_Pesos} na função de custo final do modelo. A função de custo final do modelo RDS-Constrative

é composta por três termos: a CE das amostras previamente selecionadas pela SLA, a RDS_{CE_Pesos} e $CE_{contrastive\_f}$. Assim a função de custo final do modelo é dada por:

$$RDS_{Contrastive\_loss} = CE + RDS_{CE\_Pesos} + CE_{contrastive\_f} \quad (37)$$

4.4. Pseudocódigo RDS-Contrastive

No Algoritmo 3 está apresentado o pseudocódigo do RDS-Contrastive. Até o passo 17 do algoritmo é realizada a simples aplicação do procedimento RDS no modelo *Co-teaching+,* sendo então similar ao RDS-C. A diferença se encontra no uso da RDS_{CE_Pesos} em vez da RDS-CE anteriormente apresentada.

Do passo 18 ao 30 é realizado o procedimento para se calcular o termo *contrastive*. Por fim, calcula-se a função de custo final utilizando a CE, RDS_{CE_Pesos} e o termo *contrastive* ($CE_{contrastive\_f}$).

Pseudocódigo do Modelo RDS-Contrastive

Inputs: Modelo $w_1(\theta_1)$, Modelo $w_2(\theta_2)$, learning rate η, ruído estimado τ, threshold μ, número de épocas $Tmax$, número de transformações n, hiperparâmetro γ, *dataset* D; tamanho do *batch* b_z, hiperparâmetro Tk,hiperparâmetro *temp*, funções transformação *T(.)* e *T'(.)*

Passo 1 : **For** t=1, 2, 3,…$Tmax$:

Passo 2 : *Shuffle* D

Passo 3 : Calcule o número de iterações: $\frac{D}{b_z}$

Passo 4 : **For** i=1,2... $\frac{D}{b_z}$:

Passo 5 : **Gere** o *mini-batch* D_n de D

Passo 6 : **Faça** previsões das amostras de D_n com $w_1(\theta_1)$: D_{nw_1}

Faça previsões das amostras de D_n com $w_2(\theta_2)$: D_{nw_2}

Passo 7 : **Selecione** as amostras com previsões distintas $D_{nw_1 \neq w_2}$ entre D_{nw_1}e D_{nw_2}

Passo 8 : **Selecione** R(t)% amostras $D_{n\ small:1}$ do conjunto $D_{nw_1 \neq w_2}$utilizando SLA com o modelo $w_1(\theta_1)$

Passo 9 : **Selecione** R(t)% amostras $D_{n\ small:2}$ do conjunto $D_{nw_1 \neq w_2}$utilizando SLA com o modelo $w_2(\theta_2)$

Passo 10: **Selecione** as amostras ruidosas com a equação (14) do passo 8. $D_{n+:1}$

Passo 11: **Selecione** as amostras ruidosas com a equação (14) do passo 9. $D_{n+:2}$

Passo 12: **Calcule** os pseudolabels das amostras em $D_{n+:1}$ com a equação (15), usando de *n* transformações, threshold μ, gerando o conjunto recuperado $\bar{A}_1$

Passo 13: **Calcule** os pseudolabels das amostras em $D_{n+:2}$ com a equação (15), usando de *n* transformações, threshold μ, e gerando o conjunto recuperado $\bar{A}_2$

Passo 14: **Calcule** RDS_{CE_Pesos} -equação (36)- para as amostras em $\bar{A}_1$:$RDS_{CE\_Pesos_{\bar{A}_1}}$

Passo 15: **Calcule** RDS_{CE_Pesos} -equação (36)- para as amostras em $\bar{A}_2$:$RDS_{CE\_Pesos_{\bar{A}_2}}$

Passo 16: **Calcule** CE das amostras em $D_{n\ small:1}$: $CE_{D_{n\ small:1}}$

Passo 17: **Calcule** CE das amostras em $D_{n\ small:2}$: $CE_{D_{n\ small:2}}$

Passo 18: **Aplique** transformação com a função T(.) em D_n: ${D_n}^T$

Passo 19: **Aplique** transformação com a função T'(.) em D_n: ${D_n}^{T'}$

Passo 20: **Faça** previsões nas amostras de ${D_n}^T$ com $w_1(\theta_1)$: ${D_{nw1}}^T$

Passo 21: **Faça** previsões nas amostras de ${D_n}^{T'}$ com $w_1(\theta_1)$: ${D_{nw1}}^{T'}$

Passo 22: **Faça** previsões nas amostras de ${D_n}^T$ com $w_2(\theta_2)$: ${D_{nw2}}^T$

Passo 23: **Faça** previsões nas amostras de ${D_n}^{T'}$ com $w_2(\theta_2)$: ${D_{nw2}}^{T'}$

Passo 24: **Calcule** a *softmax* com temperatura - equação (30) - em ${D_{nw1}}^T$: $softemW1T$

Passo 25: **Calcule** a *softmax* com temperatura- equação (30) - com *temp*- em ${D_{nw1}}^{T'}$: $softemW1T'$

Passo 26: **Calcule** a *softmax* com temperatura-equação (31)- com *temp*- em ${D_{nw2}}^T$ - $softemW2T$

Passo 27: **Calcule** a *softmax* com temperatura-equação (31)- - com *temp*- em ${D_{nw2}}^{T'}$ - $softemW2T'$

Passo 28: **Calcule** o primeiro termo *contrastive*-equação (32)- com $softemW1T$ E $softemW2T'$: $CE_{contrastive\_1}$

Passo 29: **Calcule** o segundo termo *contrastive*-equação (33)- com $softemW2T$E $softemW1T'$: $CE_{contrastive\_1}$

Passo 30: **Calcule** o termo *contrastive* final com $CE\_contrastive\_1$ e $CE\_contrastive\_2$: $CE_{contrastive\_f}$

Passo 31: **Calcule** o custo final para $w_1(\theta_1)$ $L_{final1} = CE_{D_{n\ small:1}} + \gamma RDS_{CE\_Pesos_{\bar{A}_1}} + CE_{contrastive\_f}$

Passo 32: **Calcule** o custo final para $w_2(\theta_2)$ $L_{final2} = CE_{D_{n\ small:2}} + \gamma RDS_{CE\_Pesos_{\bar{A}_2}} + CE_{contrastive\_f}$

Passo 33: **Atualize** $\theta_1 = \theta_1 + \eta \nabla L_{final1}$

Passo 34: **Atualize** $\theta_2 = \theta_2 + \eta \nabla L_{final2}$

Passo 35: **Atualize** $R(\tau)$ pela equação 31

Algoritmo 3 Modelo RDS-Contrastive

5
Expansão SLA para Multilabel

Nessa tese, foi feita a expansão da técnica SLA para o cenário de classificação multilabel com amostras ruidosas, e esta adaptação foi chamada de Small Loss Approach Multilabel (SLAM). Embora a técnica SLA possa ser diretamente adaptada para o cenário multilabel, foi observado dois fatores que limitam a performance dessa técnica no cenário multilabel. Assim, algumas modificações foram propostas para mitigar esses pontos.

O primeiro problema é que uma amostra com anotações multilabel pode conter, simultaneamente, classes com anotações corretas e classes com anotações incorretas, conforme ilustrado na Figura 32. Na Figura 32 (A), apresentamos uma amostra com anotações corretas. Na Figura 32 (B), vemos que uma amostra pode conter ausência de alguma classe e simultaneamente uma anotação correta. Na Figura 32 (C), vemos que uma amostra pode ter classes extras anotadas de forma equivocada.

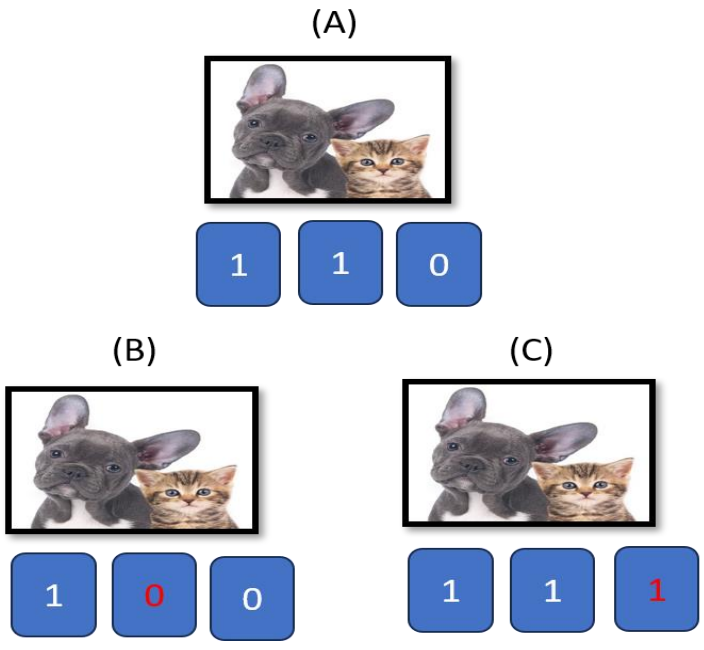


Figura 32 Exemplo dos diferentes ruídos presentes em amostras ruidosas em problemas multilabel, os dígitos em vermelho representam anotações equivocadas. (A) Exemplo da amostra sem ruído. (B) Exemplo da amostra com ausência de anotação. (C) Exemplo de anotação extra equivocada.

Com base nas diferentes formas que o ruído pode estar presente em uma amostra com anotação multilabel, foi proposto realizar a análise SLA por classe em vez de por amostra. Dessa forma, o custo de cada classe, de uma dada amostra, passa a

ser calculado de forma independente, conforme ilustrado na Figura 33. Os custos, por classe, são ranqueados para todas as amostras do batch em treinamento. As classes com os menores custos são classificadas como anotações corretas e as com maiores custos como incorretas, conforme ilustrado na Figura 34. Dessa forma, é possível identificar as classes ruidosas em cada amostra e, simultaneamente, as classes limpas da mesma amostra. Isso permite que as classes com anotações corretas continuem no treinamento e as classes incorretas descartadas.

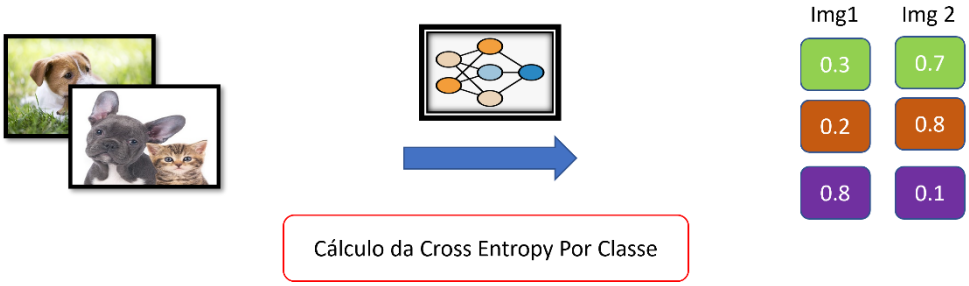


Figura 33 Exemplo do cálculo da CE separado por classe. Nesse exemplo, o dataset contém 3 classes possíveis. Os números dentro dos boxes coloridos (verde, laranja, roxo) representam o custo de cada classe.

Figura 34 Exemplo do ranqueamento do custo separado por classe. As classes com menos custos são consideradas limpas.

O segundo problema observado é o fato da SLA para multiclasse excluir do treinamento as amostras ruidosas, privando o modelo de aprender características importantes do conjunto de dados. Para resolver este problema, primeiro observamos que para uma representação *one-hot encoding* existem apenas duas opções: a classe está presente na imagem -1- ou a classe não está presente na imagem -0-. Assim, uma

vez que a classe é identificada como ruidosa, simplesmente invertemos o sinal do rótulo de -1- para -0- ou de -0- para -1- para tornar o rótulo ruidoso em um rótulo limpo. Conforme ilustrado na Figura 35.

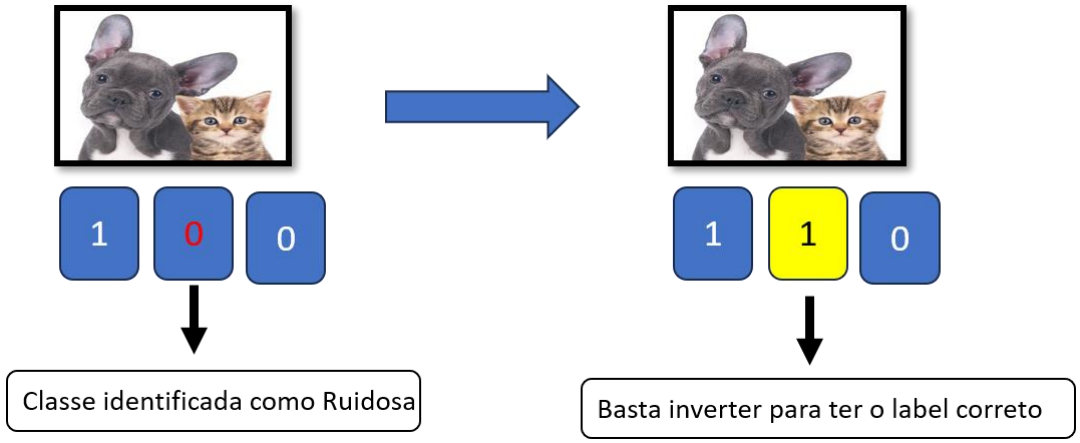


Figura 35 Exemplo para ajuste das classes incorretas

Utilizando a técnica SLAM foram desenvolvidos dois modelos: Learning by Small Loss Approach Multilabel (Learning by SLAM) e SLAM by Joint Loss (SLAM-JL).

5.1. Small Loss Approach Multilabel (SLAM): Descrição Matemática

Dado um dataset $D = \{x_i + \vec{y}_i\}_{i=1}^{z}$, onde x_i é a *i-th* amostras de *D* e $\vec{y}_i$ é o vetor de classe multilabel $\vec{y}_i = [c_i^1, c_i^2, \ldots., c_i^M], c_i \in [0,1]$, em que -0- indica ausência de classe e -1- indica presença para uma das *M* classes em treino. Dado um modelo $P(y|x:\theta)$ com parâmetros θ que para uma dada entrada *x* retorna a probabilidade desta pertencer as classes dos rótulos $\vec{y}$. O vetor da *cross entropy* $\overrightarrow{CE}_M^i$ para a amostras x_i é dado por (38):

$$\overrightarrow{CE}_M(x_i) = \vec{y}_i \odot \log(P(y|x_i:\theta)) \tag{38}$$

Onde $\odot$ representa um multiplicador *element-wise* e log também é aplicado elemento a elemento. Dessa forma, $\overrightarrow{CE}_M(x_i)$ é um vetor de custo para todas as classes da instância x_i, $\overrightarrow{CE}_M(x_i) = [l_i^1, l_i^2, \ldots, l_i^M]$, onde l_i é o custo da *i-th* classe. Os vetores $\overrightarrow{CE}_M$ para todas as amostras de um batch de tamanho *B* de um dataset *D* podem ser concatenados e representados como uma matriz *MC* com B linhas e M colunas, MC_{BM}:

$$MC_{BM} = \begin{bmatrix} l_{11} & \cdots & l_{l1M} \\ \vdots & \ddots & \vdots \\ l_{B1} & \cdots & l_{BM} \end{bmatrix} \quad (39)$$

No SLAM, são selecionados os *k* máximos valores de cada coluna da matriz MC_{BM}, onde cada coluna representa uma das classes de $\vec{y}$. O valor *k* é estimado com base no ruído presente no conjunto de dados, $k = B\,\tau$, onde τ é a porcentagem de ruído presente. O vetor de índices para os *k* máximos valores para a coluna *m* de MC_{BM} é dado por:

$$\vec{Y}(m) = arg\,max_k(\,MC_{1:B,m}\,) \quad (40)$$

Portanto, $\vec{Y}(m) = [\,id_m^1, id_m^2, \ldots, id_m^k\,]$, onde id_m é o index da matriz MC_{BM} para a classe m. O operador $arg\,max_k$ retorna os índices dos k máximos valores do vetor de entrada. $MC_{1:B,m}$ representa um vetor com todas as linhas da coluna *m* da matriz MC_{BM}.

Os índices do vetor $\vec{Y}(m)$ são equivalentes aos índices das amostras *x* no *batch* de tamanho *B*. No SLAM calcula-se $\vec{Y}(m)$ para todas as M classes presentes no *dataset*. Por fim, atribui-se novos rótulos para cada classe *m* das amostras do *batch* B, onde os índices são os do vetor $\vec{Y}(m)$. Para atribuir o novo rótulo, da classe ruidosa *m* da amostra *x*, simplesmente invertemos o rótulo de *1* para *0* ou vice--versa.

5.2. Modelo Learning By Small Loss Approach Multilabel

O modelo Learning By Small Loss Apporach Multilabel foi inspirado nos conceitos do modelo Co-teaching para amostras ruidosas em problemas multiclasse. Dessa forma, o modelo proposto opera com duas redes treinando de forma simultânea e substitui a técnica SLA pela SLAM. Além disso, os novos rótulos atribuídos por cada rede são utilizados para o treinamento da outra rede.

Na Figura 36, o fluxo do treino é ilustrado. Seguindo o Co-teaching, as duas redes (A e B) são treinadas sobre os mesmos dados simultaneamente, onde cada rede identifica as amostras e classes ruidosas dos *batchs* individualmente usando SLAM. As amostras com os novos rótulos assinalados por cada rede são utilizadas no treinamento da outra. Com os rótulos novos atribuídos, cada rede calcula a CE considerando os

novos rótulos. O algoritmo completo do modelo está apresentado abaixo em Algoritmo 4. No procedimento do modelo é adicionado um hiperparâmetro *start_epoch*. Esse parâmetro visa permitir que a rede aprenda características bases do conjunto de dados antes de atribuir novos rótulos as amostras ruidosas.

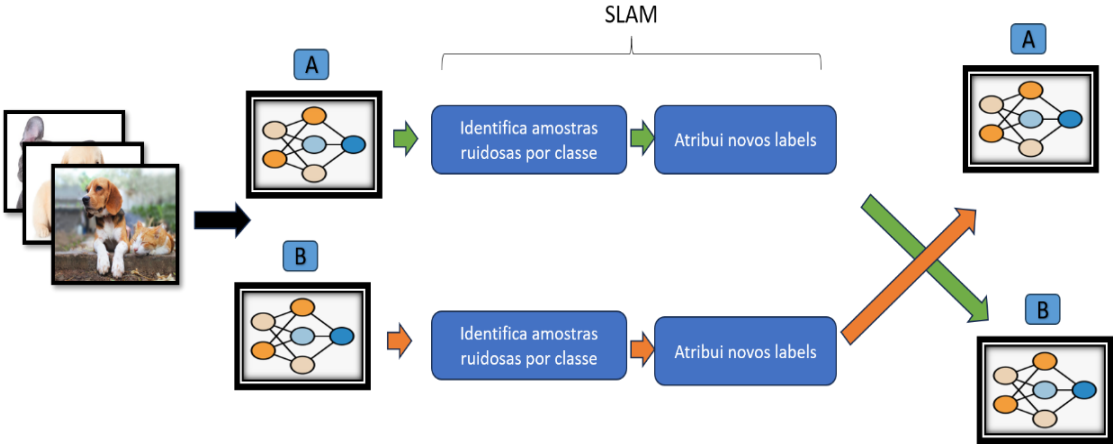


Figura 36 Fluxo do Treinamento do modelo learning By Small Loss Approach

Pseudocódigo do Modelo Learning by SLA Multilabel

Inputs: Modelo $w_1(\theta_1)$, Modelo $w_2(\theta_2)$, learning rate η, ruído estimado τ, número de épocas $Tmax$, Dataset ruidoso $D = \{x_i + \vec{y}_i\}_{i=1}^{z}$, start_epoch, tamanho do batch b_z,k

Passo 1: **For** : t=1,2... $Tmax$:

Passo 2: Shuffle D

Passo 3: Calcule o número de iterações: $\frac{D}{b_z}$

Passo 4: **For** n=1, 2,..., $\frac{D}{b_z}$

Passo 5: Crie o batch D_n de D

Passo 6: **If** t> start_epoch **Então:**

Passo 7: Crie a copia $D_n^{w_1}$ de D_n

Passo 8: Crie a copia $D_n^{w_2}$ de D_n

Passo 9: Calcule $\overrightarrow{CE}_M(x_i) \forall\ x_i\ \in\ D_n$ com $w_1(\theta_1)$

Passo 10: Calcule $\overrightarrow{CE}_M(x_i) \forall\ x_i\ \in\ D_n$ com $w_2(\theta_2)$

Passo 11: Obtenha a matriz $MC_{BM}^{w_1}$ com os velores $\overrightarrow{CE}_M(x_i)$ do Passo 9

Passo 12: Obtenha a matriz $MC_{BM}^{w_2}$ com os velores $\overrightarrow{CE}_M(x_i)$ do Passo 10

Passo 13: Calcule $Y(m)^{w_1} \forall\ m \in [1, \dots M]$ usando $MC_{BM}^{w_1}$ e k

Passo 14: Calcule $Y(m)^{w_2} \forall\ m \in [1, \dots M]$ usando $MC_{BM}^{w_2}$ e k

Passo 15: **For** m =1,2,...,M:

Passo 16: Atualize Os rótulos $y_i[m] \forall\ i\ \in\ Y(m)^{w_1}$ para as amostras x_i de $D_n^{w_1}$

- A atualização é feita pelo operador $logical_{not}$

$$logical_{not}\ 0 \rightarrow 1; 1 \rightarrow 0$$

Passo 17: Atualize Os rótulos $y_i[m] \forall\ i\ \in\ Y(m)^{w_2}$ para as amostras x_i de $D_n^{w_2}$

- A atualização é feita pelo operador $logical_{not}$

$$logical_{not}\ 0 \rightarrow 1; 1 \rightarrow 0$$

Passo 18: **Fim For**

Passo 20: **Fim If**

Passo 21: **If** t> start_epoch **Então:**

Passo 22: **Atualize** $\theta_1 = \theta_1 + \eta \nabla L(w_1, D_n^{w_2})$

Passo 23: **Atualize** $\theta_2 = \theta_2 + \eta \nabla L(w_2, D_n^{w_1})$

Passo 24: **Se não:**

Passo 25: **Atualize** $\theta_1 = \theta_1 + \eta \nabla L(w_1, D_n)$

Passo 26: **Atualize** $\theta_2 = \theta_2 + \eta \nabla L(w_2, D_n)$

Passo 27: **Fim if**

Passo 28: **Fim For**

Algoritmo 4 Modelo Learning by SLA Multilabel

5.3. Modelo SLAM by Joint Loss

O modelo Small Loss Approach Multilabel by Joint Loss (SLAM-JL) combina a Joint Loss, apresentada no trabalho [15], para problemas multiclasse, com a técnica SLAM. A *joint loss*, como apresentada originalmente, é calculada por amostra e não por classe. Frente a isso, é preciso ajustá-la para tornar possível a identificação das classes limpas e ruidosas por amostras presentes no conjunto de dados.

Embora a técnica SLAM possa melhorar a capacidade de generalização dos modelos de *DL* sobre presença de amostras ruidosas, ela ainda sofre efeitos de memorização em rótulos ruidosos [75]. Como proposto em [15], aproximar o aprendizado das duas redes, seguindo o princípio do *max agreement*, pela *joint loss*, reduz a exposição das amostras ruidosas às redes durante o treinamento. Consequentemente, mitiga os efeitos negativos da memorização das amostras. O princípio do *max agreement* aumenta a confiança nos novos rótulos das amostras ruidosas. Uma vez que dois classificadores tendem a concordar na maioria das amostras limpas, mas discordar na maioria das amostras ruidosas.

No modelo SLAM-JL, é calculado a *Joint Loss* para cada classe de forma independente. Em seguida, o processo do SLAM é aplicado sobre a *Joint Loss*, em vez da CE. Na Figura 37, esse procedimento de treino é apresentado.

Seguindo o modelo Co-teaching, são utilizadas duas redes neurais no treinamento, calcula-se a CE para cada amostra em treino, preservando o resultado para cada classe, como no SLAM. Em seguida, a *Jason Shannon Divergence* é aplicada, como termo *contrastive*, entre as previsões classe a classe dos modelos. Por fim, é realizado uma soma, classe a classe, do termo *contrastive* e das CE, gerando a função de custo *Joint Loss* adequada para multilabel. Em cima dessa função de custo o modelo continua com o procedimento do SLAM.

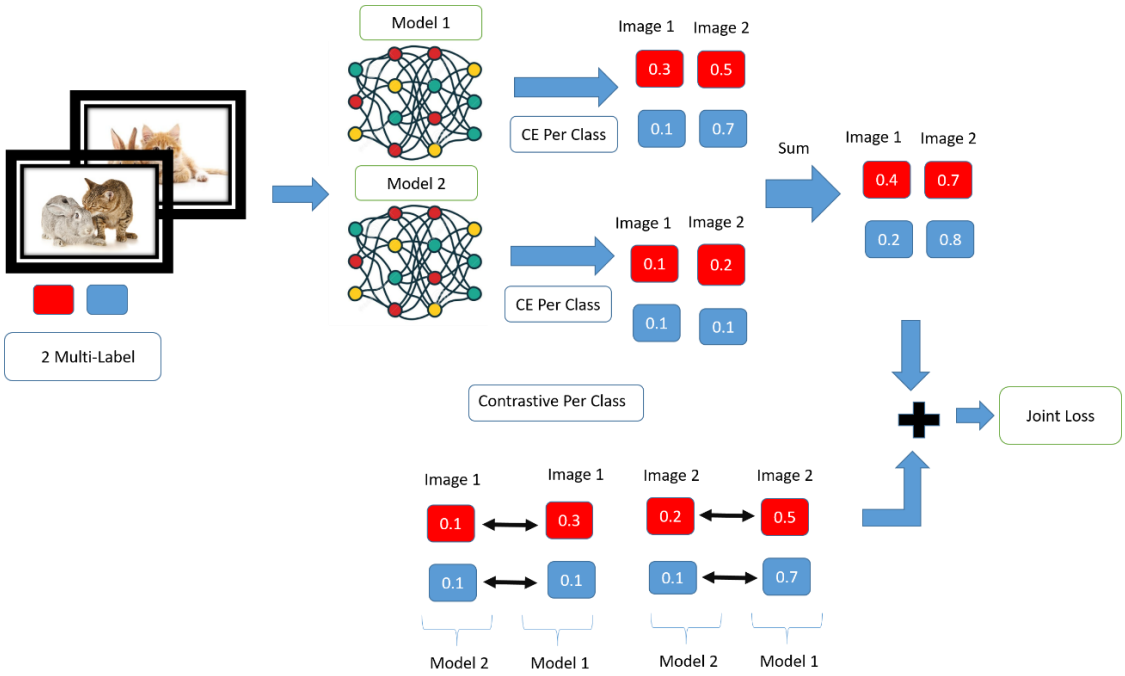


Figura 37 Procedimento do cálculo da Joint Loss Multilabel para o modelo SLAM-JL. Calcula-se a CE classe a classe, em seguida calcula-se o termo Contrastive dado pela JS. Realiza-se em seguida a soma elemento a elemento dos resultados da JD e da CE.

5.4. Modelo SLAM by Joint Loss: Descrição Matemática

Para as duas redes em treinamento no modelo calcula-se a matriz MC_{BM}, equação (39), gerando as matrizes MC_{BM1} e MC_{BM2}. Soma-se as duas matrizes:

$$MC_{BMt} = MC_{BM1} + MC_{BM2} \tag{41}$$

Com o termo *contrastive*, foi utilizado a *JS Divergence*. Esse termo aumenta a similaridade entre as previsões dos dois modelos durante o treino. Aplica-se a JS entre as previsões $\overrightarrow{CE}_M(x_i)$ entre o modelo 1 e modelo 2. Como feito no modelo Jocor, a JS foi definida em termos da KL *divergence*, equação (25). Portanto, o termo contrastive é dado por:

$$\vec{C}_{cont}(x_i) = KL(\overrightarrow{CE}_{M1}(x_i)||\overrightarrow{CE}_{M2}(x_i)) + KL(\overrightarrow{CE}_{M2}(x_i)||\overrightarrow{CE}_{M1}(x_i)) \tag{42}$$

Onde $\overrightarrow{CE}_{M1}$ é o $\overrightarrow{CE}_M(x_i)$, equação (38), para o modelo 1 e $\overrightarrow{CE}_{M2}$ é o $\overrightarrow{CE}_M(x_i)$ referente ao modelo 2. Portanto, $\vec{C}_{cont}(x_i)$ é um vetor com os termos *contrastive* de todas as classes da amostra x_i, sendo $\vec{C}_{cont}(x_i) = [lc_i^1, lc_i^2, \dots, lc_i^M]$,onde lc_i é o termo *contrastive* para a *i-th* classe da amostra x_i, entre o modelo 1 e modelo 2. Todas os $\vec{C}_{cont}(x_i)$, para um *batch* de tamanho *B,* de um *Dataset D,* podem ser concatenados, gerando uma matriz G, com B linhas e M colunas, G_{BM}:

$$G_{BM} = \begin{bmatrix} lc_{11} & \cdots & lc_{1M} \\ \vdots & \ddots & \vdots \\ lc_{B1} & \cdots & lc_{BM} \end{bmatrix} \quad 43$$

A *Joint Loss* Multilabel é dada pela soma de MC_{BMt} e G_{BM} sendo dada por:

$$JL_{BM} = MC_{BMt} + G_{BM} \quad 44$$

Por fim, todo o procedimento final do modelo SLAM, a partir da equação (38), é replicado em cima da JL_{BM} em vez da MC_{BM}. As duas redes nesse modelo são treinadas utilizando o procedimento de redes siamesas seguindo o modelo Jocor para problemas multiclasse. Dessa forma, com os novos rótulos assinalados, a função de custo final, utilizada para treinar as duas redes siamesas, *Loss Multinoise Label* (L_{MNL}), é soma entre a CE do modelo 1 e a CE do modelo 2 sobre os novos rótulos:

$$L_{MNL} = \sum_{n=1}^{z} \overrightarrow{-y_i} \odot \log(P1(y|x_i:\theta)) + \sum_{n=1}^{z} \overrightarrow{-y_i} \odot \log(P2(y|x_i:\theta)) \quad 45$$

No Algoritmo 5 o passo a passo do treino é apresentado.

	Learning by SLA Multilabel by Joint Loss
Input:	Modelo $w_1(\theta_1)$, Modelo $w_2(\theta_2)$, learning rate η, ruído estimado τ, número de épocas $Tmax$, Dataset ruidoso $D = \{x_i + \vec{y}_i\}_{i=1}^{z}$, start_epoch, tamanho do batch b_z,k
Passo 1	**For** t=1,2,… $Tmax$:
Passo 2	Shuffle D
Passo 3	Calcule o número de iterações: $\frac{D}{b_z}$
Passo 4	**For** *n=1,2,..,*$\frac{D}{b_z}$:
Passo 5	Crie o batch D_n de D
Passo 6	**IF** t> start_epoch :
Passo 7	Calcule $\overrightarrow{CE}_M(x_i)\ \forall\ x_i \in D_n$ com $w_1(\theta_1)$
Passo 8	Calcule $\overrightarrow{CE}_M(x_i)\ \forall\ x_i \in D_n$ com $w_2(\theta_2)$
Passo 9	Obtenha a matriz $MC_{BM}^{w_1}$ com os velores $\overrightarrow{CE}_M(x_i)$ do Passo 7
Passo 10	Obtenha a matriz $MC_{BM}^{w_2}$ com os velores $\overrightarrow{CE}_M(x_i)$ do Passo 8

Passo 11 Calcule $MC_{BM_t} = MC_{BM}^{w_1} + MC_{BM}^{w_2}$, equação (41)
Passo 12 Calcule $\vec{C}_{cont}(x_i)\ \forall\ x_i \in D_n$ com $\overrightarrow{CE}_M(x_i)$ do passo 7 e 8
Passo 13 Calcule a matriz G_{BM} equação 43
Passo 14 Calcule JL_{BM} equação 44
Passo 15 Calcule $Y(m)\ \forall\ m \in [1, \dots M]$ usando JL_{BM} e k
Passo 17 **For** m =1,2,…,M:
Passo 18 Atualize Os rótulos $y_i[m] \forall\ i \in Y(m)$ para as amostras x_i de D_n

- A atualização é feita pelo operador $logical_{not}$

$$logical_{not}\ 0 \rightarrow 1; 1 \rightarrow 0$$

Passo 19 **Fim For**
Passo 20 Calcule L_{MNL}, equação 45, com $w_1(\theta_1)$ e $w_2(\theta_2)$ com os novos labels de D_n(passo 18)
Passo 21 **Fim If**
Passo 22 **If** t> start_epoch **Então:**
Passo 23 **Atualize** $\theta_1 = \theta_1 + \eta\nabla L_{MNL}$
Passo 24 **Atualize** $\theta_2 = \theta_2 + \eta\nabla L_{MNL}$
Passo 25 **Se não:**
Passo 26 **Atualize** $\theta_1 = \theta_1 + \eta\nabla L(w_1, D_n)$
Passo 27 **Atualize** $\theta_2 = \theta_2 + \eta\nabla L(w_2, D_n)$
Passo 28 **Fim if**
Passo 29 **Fim For**

Algoritmo 5 Learning by SLA Multilabel by Joint Loss

6
Resultados e Discussões

Nessa seção serão apresentados e discutidos os resultados obtidos nos experimentos e os detalhes experimentais.

6.1.
Experimental

Foi realizada uma comparação entre os modelos RDS-J, RDS-C, RDS-Contrastive e os modelos do estado da arte, SAT [58], Co-teaching+ [14], Jocor [15], Decoupling [13], além do modelo *standard* que se refere a um modelo *DL* sem nenhuma abordagem para lidar com amostras ruidosas.

Para uma comparação justa, todos os modelos foram treinados utilizando a mesma rede neural com os mesmos hiperparâmetros. Os detalhes da rede estão apresentados na Tabela 1. A taxa de aprendizado η utilizada foi de 0.001 e o otimizador foi o *Adam* [76] com momentum 0.9 – todos escolhidos empiricamente. O hiperparâmetro T_k utilizado na função *R(τ)* - equação (24) - foi de $T_k = 10$, para os modelos RDS-J, RDS-C, RDS-Contrastive, Jocor e Co-teaching+ como indicado no trabalho em [5]. O número de épocas utilizado foi de 200. Para os modelos RDS-C, RDS-J e RDS-Contrastive o número de transformações *n* foi de 2 e o *threshold* μ utilizado foi de 0.80. Para o RDS-Contrastive utilizou-se o hiperparâmetro *temp=0.25*. O hiperparâmetro $\gamma = 0.25$ para os modelos RDS.

O protocolo para avaliação foi o mesmo utilizado nos trabalhos [5] [14] [15], sendo utilizado a acurácia de teste dado por:

$$Acurácia\ de\ teste = (\#inferências\ corretas)/(\#conjunto\ \ teste\ do\ dataset) \quad (46)$$

O conjunto de teste é composto apenas de amostras limpas. Ainda seguindo o protocolo adotado em [5], [14], [15], os experimentos realizados foram com os datasets CIFAR-10, CIFAR-100 e MNIST com ruídos inseridos manualmente. Os ruídos inseridos foram de $\tau = 20$ e $\tau = 50$ para o ruído simétrico e $\tau = 45$ para o ruído *pair flip*. O conjunto de teste foi selecionado utilizando a separação padrão do *Tensorflow* [77], ou seja,

utilizando a função *split* com configuração padrão do pacote *keras tensorflow.keras.datasets* versão 2.4, para os *datasets* CIFAR-10, CIFAR-100 e MNIST.

Para os modelos RDS-C RDS-J e RDS-Constrative utilizou-se ainda a acurácia RDS como métrica, dada por:

$Acurácia\ RDS =$ (# de *pseudolabels* corretos) / (# conjunto Relabel total) (47)

Sendo o Relabel total definido no tópico anterior 3.2. Além disso, também foi realizado a visualização gráfica do *relabel* total ao longo das épocas para visualizar a quantidades de amostras recuperadas. Avaliou-se também como o desempenho dos modelos RDS-C, RDS-J e RDS-Contrastive são afetados com o número de transformações *n* e o *threshold* μ.

Para situar os modelos RDS-C, RDS-J e RDS-Contrastive de forma ampla no estado da arte, foi realizado uma avaliação dos modelos no *dataset* Clothing1M [78]. Este *dataset* foi formado por busca ampla na internet de imagens de vestimentas, onde a legenda da imagem é associada a imagem coletada, e é apontado no trabalho [78] como contendo 40% de amostras ruidosas. Esse conjunto de dados é comumente utilizado para comparar os modelos do SOTA na área de *Noisy Labels*, e neste trabalho, estamos seguindo as mesmas condições do trabalho [22] para realizar uma comparação justa dos modelos.

Tabela 1 Detalhes do modelo CNN utilizado nos experimentos conduzidos

	Cifar 10	**Cifar 100**	**Mnist**
Input Shape	32x32x3	32x32x3	28x28x3
CNN Layer	Filters=128, kernel size=(3x3) strides=(1, 1) activation function = LeakyRelu	Filters=128, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=128, kernel size=(3x3) strides=(1, 1) activation function = LeakyRelu
Batch Normalization	Batch Normalization	Batch Normalization	Batch Normalization
CNN Layer	Filters=128, kernel size =(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=128, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=128, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu
Batch Normalization	Batch Normalization	Batch Normalization	Batch Normalization
CNN Layer	Filters=128, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=128, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=128, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu
Batch Normalization	Batch Normalization	Batch Normalization	Batch Normalization
Max pooling	Pool size=(2x2) strides=(2,2)	Pool size=(2x2) strides=(2,2)	Pool size=(2x2) strides=(2,2)
Dropout layer	rate=0.25	rate=0.25	rate=0.25
CNN Layer	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu
Batch Normalization	Batch Normalization	Batch Normalization	Batch Normalization
CNN Layer	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu
Batch Normalization	Batch Normalization	Batch Normalization	Batch Normalization
CNN Layer	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu
Batch Normalization	Batch Normalization	Batch Normalization	Batch Normalization
Max pooling	Pool size=(2x2) strides=(2,2)	Pool size=(2x2) strides=(2,2)	Pool size=(2x2) strides=(2,2)
Dropout layer	rate=0.25	rate=0.25	rate=0.25
CNN Layer	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu
Batch Normalization	Batch Normalization	Batch Normalization	Batch Normalization
CNN Layer	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu
Batch Normalization	Batch Normalization	Batch Normalization	Batch Normalization
CNN Layer	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu	Filters=256, kernel size=(3x3) strides= (1, 1) activation function = LeakyRelu
Batch Normalization	Batch Normalization	Batch Normalization	Batch Normalization
GlobalAverage Pooling	GlobalAverage Pooling	GlobalAverage Pooling	GlobalAverage Pooling
Dense Layer	10 Classes, activation=softmax	100 Classes, activation=softmax	10 Classes, activation=softmax

6.2.
Experimental para Problemas Multilabel

Os modelos comparados com os SLAM e SLAM by Joint Loss foram os Co-teaching, Jocor e o modelo standard nas suas versões multilabel conforme propostos em [79] . Para avaliar a performance dos modelos multilabel foi necessário elaborar dois *datasets* multilabels ruidosos. Dessa forma, foi adicionado ruído nos *dataset* multilabel UcMerced e TreeSatAI. Esses *datasets* estão disponíveis em https://github.com/ICA-PUC, permitindo que trabalhos futuros realizem comparações nas mesmas condições.

O ruído inserido foi por classe, sendo aditivo ou subtrativo [79]. O ruído aditivo é quando se adiciona uma classe que não existe na amostra, enquanto o ruído subtrativo exclui uma classe que existe na amostra. O ruído foi inserido da seguinte forma: (1) para todas as classes, seleciona-se aleatoriamente 25% das amostras do conjunto treino; (2) se a amostra selecionada pertencer a classe abordada, essa classe é removida (ruído subtrativo), se não, a classe é adicionada (ruído aditivo).

Foi utilizado como *backbone* a VGG-16 com os pesos pré treinados da *ImageNet*. Apenas as últimas quatro camadas da rede foram otimizadas. Foi utilizado o pré-processamento padrão do *Tensorflow* versão 2.4 para a VGG-16 nas imagens de entrada. O otimizador utilizado foi o *Adam,* o *learning rate* foi 0.00025, $T_k = 40$. O $start_{epoch}$ usado no SLAM foi 1, o ruído estimado foi de 0.25, o *batch-size* de 64. O número de épocas para o *dataset* UcMerced foi 50 e para o *dataset* TreeSatAi foi 30.

Utilizou-se a métrica F1-Score sobre o conjunto de teste, onde o conjunto de teste contém apenas amostras limpas, para comparar os modelos. Além disso, foi desenvolvido uma segunda métrica para avaliar a acurácia dos novos rótulos assinalados pelos modelos desenvolvidos nessa tese. Essa métrica foi chamada de Acurácia SLA Multilabel, sendo calculada ao final de todas época de treino sobre o conjunto treino. A Acurácia SLA Multilabel é dada por:

$$Acc_{SLAMultiLabel}(m) = \frac{\#\ de\ rótulos\ corretos}{\#total\ de\ rótulos} \quad (48)$$

Essa métrica informa a porcentagem de rótulos corretos assinalados por classe pelo modelo.

6.3. CIFAR-10

Em todos os gráficos dessa seção a linha sólida representa o resultado médio da referida métrica obtida pelos modelos ao longo das épocas de treinamento. A região sombreada da mesma cor da curva média representa o desvio padrão da métrica referida obtida pelos modelos ao longo das épocas.

Na Figura 38, está apresentado o gráfico da acurácia de teste ao longo das épocas para o *dataset* Cifar 10 com ruído *pair flip* e $\tau = 45$.

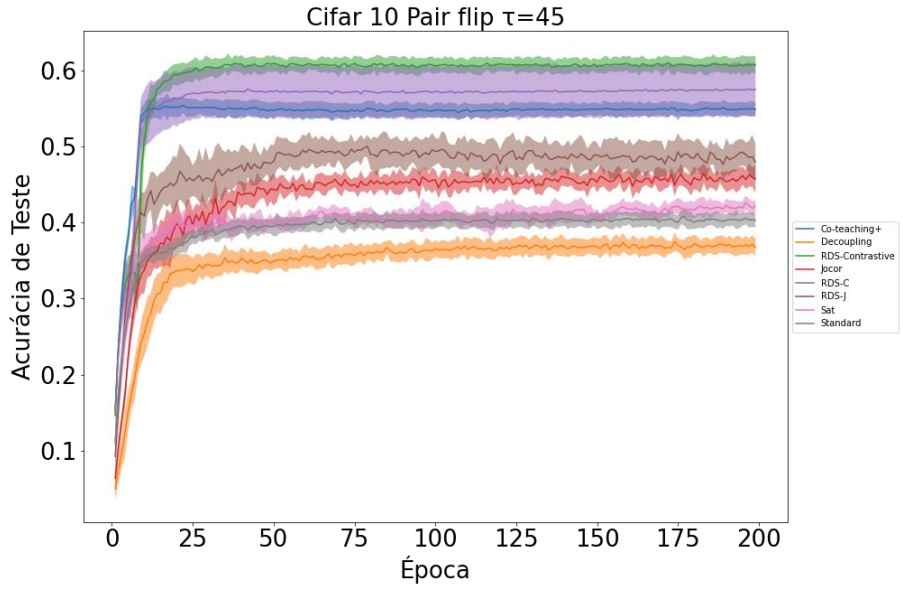


Figura 38 Gráfico da acurácia de teste para o dataset *Cifar 10 com ruído Pair Flip t=45*

Observa-se que o modelo RDS-Contrastive teve o melhor desempenho seguido do RDS-C. O RDS-J teve um resultado intermediário, apesar disso, destaca-se que o RDS-J teve um desempenho melhor do que o modelo JOCOR, assim como o RDS-C está acima do Co-teaching+. Esse primeiro resultado é um indicativo dos ganhos que a técnica RDS traz aos modelos, pois todos os modelos com RDS obtiveram melhoras no desempenho, quando comparados com os mesmos modelos sem o uso da técnica. A técnica RDS, essencialmente, possibilita que a rede aprenda características importantes do conjunto dos dados que estavam sendo excluídas pela SLA, esses primeiros resultados indicam que recuperar essas características ao treinamento tornam o treino da rede mais robusto.

Na Figura 39 está apresentado o gráfico da Acurácia RDS para o *dataset* Cifar 10 com ruído *pair flip* e $\tau = 45$. Esse gráfico está informando a porcentagem de *pseudolabels* corretos que estão retornando ao treino pelo processo RDS. Ou seja, ele

informa a porcentagem de *pseudolabels* que foram atribuídos de forma correta pelo RDS-Label. Observa-se que o modelo RDS-Contrastive teve o melhor desempenho, seguido do RDS-C e por último o RDS-J. Destaca-se que para a acurácia de teste os modelos reproduziram essa mesma ordem de desempenho.

O resultado da Acurácia RDS ilustra a capacidade de cada modelo de atribuir pseudolabels corretos ao longo das épocas. Intuitivamente, espera-se que modelos com melhor desempenho na Acurácia RDS sejam mais robustos. Pois *pseudolabels* corretos levam a rede a associar as características, anteriormente excluídas pelo SLA, as classes corretas. Por outro lado, pseudolabels incorretos guiam a rede a associar as novas características aprendidas as classes incorretas. De fato, os resultados até então obtidos ao se analisar a Figura 39 em conjunto com a Figura 38 reforçam esse indicativo.

Na Figura 40 está apresentado o gráfico do Relabel total, esse gráfico informa quantas amostras estão sendo recuperadas pelo processo RDS. Ou seja, informa a quantidade de amostras que tiveram *pseudolabels* válidos pelo sistema RDS-Label. É importante ressaltar a interligação entre a Acurácia RDS e o Relabel total. A porcentagem da Acurácia RDS diz respeito a quantidade de amostras do Relabel total. Por exemplo, na época 100, o modelo RDS-Contrastive teve uma Acurácia RDS na ordem de 0.6 e o Relabel total na ordem de 18.000 amostras. Esses dois dados, em conjunto, informam que o modelo RDS-Contrastive na época 100 atribuiu 18.000 *pseudolabels* válidos, entretanto apenas 10.800 (60%) desses *pseudolabels* estavam corretos.

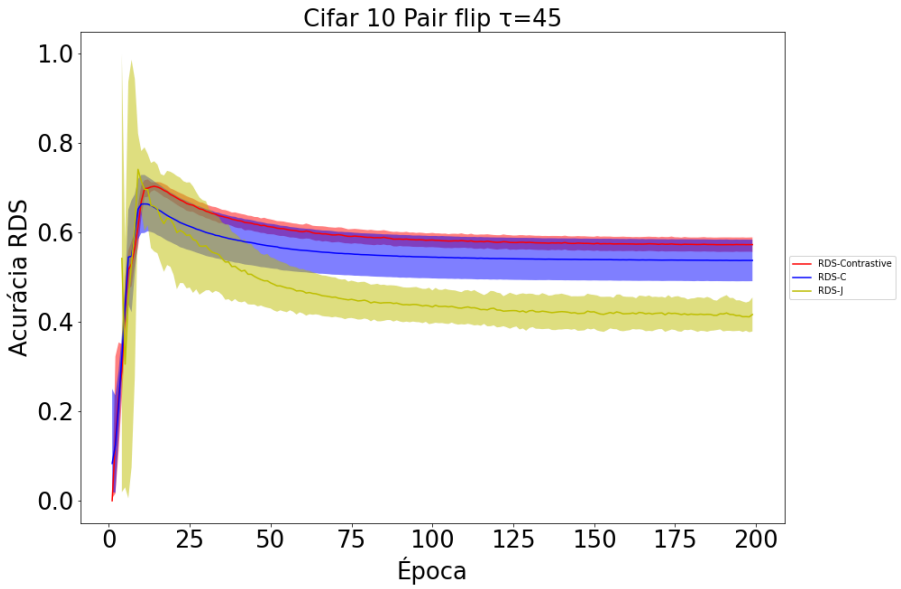


Figura 39 Gráfico da Acurácia RDS para o dataset Cifar 10 com ruído Pair Flip t=45

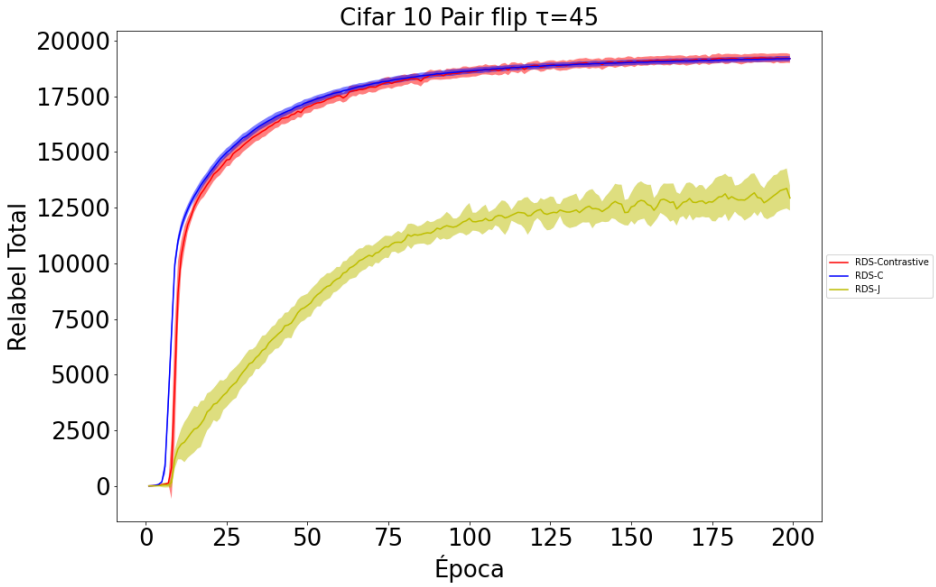


Figura 40 Gráfico do Relabel Total para o dataset Cifar 10 com ruído Pair Flip t=45

A informação que o gráfico Relabel Total traz para compreender o desempenho do modelo é essencial, pois modelos de DL necessitam de grandes quantidades de dados para um aprendizado robusto [43]. Assim, espera-se que quanto mais amostras sejam recuperadas pelo RDS maior a possibilidade do aprendizado das novas características anteriormente excluídas pela SLA. O ideal seria um modelo capaz de recuperar todas as amostras excluídas pelo SLA e que todas essas amostras tenham o pseudolabel correto, i.e., Acurácia RDS de 100%. Esse cenário seria equivalente a um treinamento livre de ruídos.

Dessa forma, o modelo com o melhor desempenho, ao se utilizar o RDS, deve ter grande quantidade de amostras no Relabel Total e alta Acurácia RDS. Outro ponto importante para o desempenho final de cada modelo é o quão robusto a ruído é o processo utilizado no treinamento com as amostras limpas oriundas da SLA. Por exemplo, o modelo RDS-J utiliza o procedimento descrito no modelo Jocor e o modelo RDS-C o procedimento do Co-teaching+. Nesse cenário - *dataset* Cifar 10 com ruído *pair flip* 45 -, o procedimento do RDS-C teve um desempenho melhor em relação ao do RDS-J.

Tabela 2 Comparação dos modelos do Estado da Arte sobre as métricas Acurácia (Ac.), Acurácia RDS (Ac. RDS) e Relabel Total (Rel.) para o dataset Cifar-10. O melhor resultado de cada coluna está destacado em negrito. Os resultados apresentados referem-se à performance do modelo na última época de treinamento.

Modelo	Pair Flip t=45			Simétrico t=20			Simétrico t = 50		
	Ac.	Ac. RDS	Rel.	Acc.	Ac. RDS	Rel.	Ac.	Ac. RDS	Rel.
Co-teaching+	0.5486	-	-	0.6804	-	-	0.5835	-	-
Decoupling	0.3692	-	-	0.5592	-	-	0.3142	-	-
RDS-Contrastive	**0.6065**	0.7298	**22141**	0.7042	0.6670	**8590**	**0.6313**	**0.4109**	**18700**
Jocor	0.45735	-	-	0.7041	-	-	0.6078	-	-
RDS-C	0.5743	0.7000	22100	0.6964	0.6548	8560	0.6101	0.4034	18690
RDS-J	0.4863	**0.7709**	17020	**0.7160**	**0.7300**	8080	0.6252	0.5059	12500
SAT	0.4206	-	-	0.6567	-	-	0.4126	-	-
Standard	0.4035	-	-	0.6054	-	-	0.3673	-	-

Na Tabela 2 está apresentado um resumo geral da comparação dos modelos do estado da arte sobre as métricas Acurácia (Ac.), Acurácia RDS (Ac. RDS) e Relabel Total (Rel.) quando aplicáveis. Na tabela estão presentes os resultados para os ruídos simétrico com $\tau = 20{,}50$, e *par-flip* com $\tau = 45$. As curvas de todas as métricas ao longo do treinamento dos modelos estão apresentadas em detalhes no anexo B.

Na Tabela 2, na seção Simétrico t=20, referente a acurácia, o modelo RDS-J teve o melhor desempenho seguido do RDS-Contrastive. Novamente, o RDS-J teve um desempenho superior ao Jocor, assim como o RDS-C teve um desempenho melhor que o Co-teaching+. Além disso, observa-se que, o RDS-J teve o melhor desempenho na Acurácia RDS, já no Relabel total, o melhor desempenho foi do RDS-Contrastive.

Na Tabela 2, na seção Simétrico t=50, referente a acurácia, o modelo com melhor desempenho foi do RDS-Contrastive seguido do RDS-J e do RDS-C. Assim como nos casos anteriores, o RDS-C teve um desempenho melhor que o Co-teaching+ e o RDS-J teve um desempenho melhor que o Jocor. Observa-se, que o RDS-Contrastive teve o melhor desempenho na Acurácia RDS e no Relabel Total.

Os resultados indicam uma tendência de aumento no desempenho da acurácia de teste quando o resultado do modelo no Relabel Total e na Acurácia RDS aumentam. No tópico 6.7, essa possibilidade será amplamente explorada ao ser realizado uma análise do desempenho dos modelos em relação ao número de data augmentation *n* e do threshold μ. Em suma, a técnica RDS possibilita que os modelos aprendam novas características que anteriormente estavam sendo excluídas pelo SLA e isto possibilita uma melhora no desempenho dos modelos.

Como os modelos RDS-C e RDS-J possuem interdependência com os modelos Co-teaching+ e Jocor respectivamente, foi realizado um teste de hipótese *pair-t* [80] para demostrar que os resultados obtidos pelos modelos possuem diferença estatística

significativa. A comparação foi realizada entre o modelo RDC-C e Co-teaching+ e entre o modelo RDS-J e Jocor. Nesse teste, o *p-value* [80] acima de 0.05 informa que os dados pertencem a mesma distribuição e um *p-value* abaixo de 0.05 informa que pertencem a distribuição distinta. A hipótese nula é que não há diferença entre os resultados obtidos. O valor *p-value* calculado é menor do que 0.05, confirmando que a hipótese nula pode ser rejeitada e, consequentemente, os diferentes resultados obtidos pelos dois modelos podem ser considerados estatisticamente significativos.

Na Tabela 3 estão apresentados os resultados obtidos para o *dataset* Cifar-10, onde a coluna *t* informa a porcentagem de ruído presente. Observa-se que todos os experimentos apontaram para distribuições distintas. No anexo A também acrescentamos os *boxplots* para os experimentos realizados.

Os resultados do teste de hipótese reforçam as observações anteriormente realizadas que a técnica RDS de fato está influenciando nos resultados dos modelos de forma positiva. Uma vez que é apontando distribuição distinta para todos os experimentos e como o modelo RDS-J e RDS-C apontaram resultados melhores quando comparados ao Jocor e Co-teaching+, respectivamente, fica evidente a melhoria dos modelos ao se utilizar a técnica RDS.

Tabela 3 Teste de hipótese para os para os diferentes modelos sobre o dataset Cifar-10 na época 150 sobre o conjunto de teste

Modelo1	Modelo2	Tipo de Ruído	t	p value	distribuição	DATASET
Co-teaching+	RDS-C	Pair flip	45	0.001846098	Distintas	Cifar 10
Co-teaching+	RDS-C	Simétrico	20	2.74E-17	Distintas	Cifar 10
Co-teaching+	RDS-C	Simétrico	50	1.34E-18	Distintas	Cifar 10
Jocor	RDS-J	Simétrico	20	0.000702597	Distintas	Cifar 10
Jocor	RDS-J	Simétrico	50	3.05E-05	Distintas	Cifar 10
Jocor	RDS-J	Pair flip	45	0.002115271	Distintas	Cifar 10

6.4. CIFAR 100

Na Tabela 4, está apresentado um resumo geral, referente ao *dataset* Cifar 100, da comparação dos modelos do estado da arte sobre as métricas Acurácia (Ac.), Acurácia RDS (Ac. RDS) e Relabel Total (Rel.) quando aplicáveis. Na Tabela 4 estão presentes os resultados para os ruídos simétrico com $\tau = 20{,}50$ e par-flip com $\tau = 45$. As curvas de todas as métricas ao longo do treinamento dos modelos estão apresentadas em detalhes no anexo B.

Tabela 4 Comparação dos modelos do estado da arte sobre as métricas Acurácia (Ac.), Acurácia RDS (Ac. RDS) e Relabel Total (Rel.) para o dataset Cifar-100. O melhor resultado de cada coluna está destacado em negrito. Os resultados apresentados referem-se à performance média dos modelos na última época de treinamento.

Modelo	Pair Flip t=45			Simétrico t=20			Simétrico t = 50		
	Ac.	Ac. RDS	Rel.	Acc.	Ac. RDS	Rel.	Ac.	Ac. RDS	Rel.
Co-teaching+	0.2054	-	-	0.3484	-	-	0.2700	-	-
Decoupling	0.1618	-	-	0.2406	-	-	0.1101	-	-
RDS-Contrastive	0.1891	**0.1865**	1150	0.3600	**0.5109**	5890	**0.3126**	**0.4706**	20980
Jocor	0.2209	-	-	0.3754	-	-	0.2903	-	-
RDS-C	0.2159	0.1712	**17192**	0.3716	0.4729	**8350**	0.3089	0.4670	21856
RDS-J	**0.2280**	0.1678	15680	**0.3883**	0.4890	7790	0.3104	0.5614	14564
SAT	0.1869	-	-	0.2990	-	-	0.1415	-	-
Standard	0.1928	-	-	0.2836	-	-	0.1384	-	-

Na Tabela 4, o modelo RDS-J teve o melhor desempenho na acurácia, referente ao ruído *Pair Flip*=45, seguido do modelo Jocor. O modelo RDS-C teve um desempenho melhor que o modelo Co-teaching+, mostrando mais uma vez a melhora que o sistema RDS traz aos modelos. O modelo RDS-Contrastive teve um desempenho intermediário. Observa-se que na métrica Acurácia RDS o modelo RDS-Contrastive teve desempenho superior ao RDS-C e RDS-J, porém esse desempenho não é seguido na acurácia de teste. Referente ao Relabel Total, para o *dataset* Cifar 100, com ruído *Pair Flip* t=45 o modelo RDS-Contrastive teve um desempenho muito abaixo em relação aos modelos RDS-J e RDS-C. Esse resultado indica a importância de um bom desempenho no RDS-Relabel Total para o desempenho final do modelo. Essa relação será aprofundada no tópico 6.7.

Na seção referente ao Simétrico e $\tau = 20$ o modelo RDS-J teve o melhor desempenho, seguido do modelo JOCOR bem próximo do RDS-C. Quando referente a Acurácia RDS o modelo RDS-Contrastive teve o melhor desempenho seguido do RDS-J e por último o modelo RDS-C. Novamente, essa relação não é seguida no desempenho da acurácia de teste do modelo. Como já mencionado, para um indicativo melhor do desempenho final do modelo é preciso também analisar o Relabel Total. Na métrica do Relabel Total, para simétrico t =20, observa-se que o modelo RDS-C teve o melhor desempenho seguido do RDS-J. Na seção referente ao ruído simétrico e $\tau = 50$, o modelo RDS-J teve um desempenho superior ao Jocor, assim como o RDS-C foi superior ao Co-teaching+.

Como realizado para os experimentos do dataset Cifar-10, também foi realizado um teste de hipótese para o Cifar-100 com o pair-t [80], comparando os modelos RDS-C e Co-teaching+ e o RDS-J com o Jocor. Na Tabela 5 estão apresentados os resultados obtidos. A coluna t informa a porcentagem de ruído presente. Novamente, como nos experimentos do Cifar-10 os resultados do teste de hipótese nos

experimentos do Cifar-100 reforçam as observações anteriormente realizadas que a técnica RDS de fato está influenciando nos resultados dos modelos de forma positiva. No anexo A também acrescentamos os *boxplots* para os experimentos realizados.

Tabela 5 Teste de hipótese para os para os diferentes modelos sobre o dataset Cifar-100 na época 150 sobre o conjunto de teste

Modelo1	Modelo2	Tipo de Ruído	t	p value	distribuição	Dataset
Co-teaching+	RDS-C	Simétrico	50	1.44E-26	Distintas	Cifar 100
Co-teaching+	RDS-C	Simétrico	20	5.44E-19	Distintas	Cifar 100
Co-teaching+	RDS-C	Pair flip	45	9.80E-08	Distintas	Cifar 100
Jocor	RDS-J	Pair flip	45	0.043138103	Distintas	Cifar 100
Jocor	RDS-J	Simétrico	20	4.83E-05	Distintas	Cifar 100
Jocor	RDS-J	Simétrico	50	1.96E-06	Distintas	Cifar 100

6.5. Resultados Mnist

Na Tabela 6, está apresentado um resumo geral, referente ao dataset Mnist, da comparação dos modelos do estado da arte sobre as métricas Acurácia (Ac.), Acurácia RDS (Ac. RDS) e Relabel Total (Rel.) quando aplicáveis. As curvas de todas as métricas ao longo do treinamento dos modelos estão apresentadas em detalhes no anexo B.

Tabela 6 Comparação dos modelos do Estado da Arte sobre as métricas Acurácia (Ac.), Acurácia RDS (Ac. RDS) e Relabel Total (Rel.) para o dataset Mnist. O melhor resultado de cada coluna está destacado em negrito. Os resultados apresentados referem-se à performance na última época de treino.

Modelo	Pair Flip t=45			Simétrico t=20			Simétrico t = 50		
	Ac.	Ac. RDS	Rel.	Acc.	Ac. RDS	Rel.	Ac.	Ac. RDS	Rel.
Co-teaching+	0.9337	-	-	0.9876	-	-	0.9745	-	-
Decoupling	0.5732	-	-	0.9658	-	-	0.5746	-	-
RDS-Contrastive	**0.9605**	0.**9620**	**24600**	**0.9900**	**0.9882**	**12506**	**0.9835**	**0.9860**	**29080**
Jocor	0.9265	-	-	0.9863	-	-	0.9706	-	-
RDS-C	0.9456	0.9500	24118	0.9890	0.9800	12487	0.9812	0.9815	28200
RDS-J	0.9355	0.9600	23102	0.9868	0.9750	11702	0.9742	0.9900	26300
SAT	0.5904	-	-	0.9789	-	-	0.8383	-	-
Standard	0.5649	-	-	0.9211	-	-	0.6140	-	-

Na Tabela 6, seção *Pair Flip* t=45, o modelo RDS-Contrastive teve o melhor desempenho seguido do RDS-C. Conservando-se o padrão dos outros *dataset*s, o modelo RDS-C teve um desempenho melhor que o Co-teaching+ e o RDS-J teve um desempenho melhor que o Jocor. Destaca-se novamente a importância do alto desempenho do Relabel Total e da Acurácia RDS para um bom desempenho na acurácia de teste. Observa-se que o modelo RDS-J e RDS-Contrastive na Acurácia RDS tiveram desempenhos similares, entretanto no Relabel Total o modelo RDS-Contrastive

teve um desempenho superior. Assim, o modelo RDS-Contrastive teve um resultado melhor na acurácia de teste. Como será visto no tópico 6.7 essa relação é um importante indicativo para o desempenho final do modelo.

Como realizado para os experimentos do *dataset* Cifar-10 e Cifar 100, também foi realizado um teste de hipótese para o Mnist com o *pair-t* [80], comparando os modelos RDS-C e Co-teaching+ e o RDS-J com o Jocor. Na Tabela 7 estão apresentados os resultados obtidos, a coluna *t* informa a porcentagem de ruído presente.

Observa-se que todos os experimentos apontaram para distribuições distintas, com exceção do Mnist com ruído simétrico 20%. O resultado indica que ambos os modelos Jocor e RDS-J obtiveram desempenho estatisticamente iguais para o experimento com o Mnist com ruído simétrico 20%. Esse resultado pode ser explicado, pela simplicidade do *dataset* Mnist e pelo fato do ruído simétrico com t=20% ser o caso mais simples a ser tratado como apontando em [5]. Dessa forma a margem de melhoria para a performance do modelo Jocor é extremamente diminuta, uma vez que este apontou um ótimo desempenho sem a técnica RDS. No anexo A também acrescentamos os *boxplots* para os experimentos realizados.

Tabela 7 Teste de hipótese para os para os diferentes modelos sobre o dataset Mnist na época 150 sobre o conjunto de teste

Modelo1	Modelo2	Tipo de Ruído	t	p value	distribuição	Dataset
Co-teaching+	RDS-C	Simétrico	20	0.000372815	Distintas	Mnist
Co-teaching+	RDS-C	Simétrico	50	2.72E-16	Distintas	Mnist
Co-teaching+	RDS-C	Pair flip	45	1.38E-07	Distintas	Mnist
Jocor	RDS-J	Pair flip	45	0.025994423	Distintas	Mnist
Jocor	RDS-J	Simétrico	20	0.344815249	Iguais	Mnist
Jocor	RDS-J	Simétrico	50	0.024642231	Distintas	Mnist

Os experimentos realizados sobre os *dataset*s Cifar-10, Cifar-100 e Mnist ilustram os ganhos que a técnica RDS traz aos modelos Co-teaching+ e Jocor, pois em todos os cenários os modelos RDS-C foi superior ao Co-teaching+ e o RDS-J foi superior ao Jocor. Os ganhos na acurácia de teste chegaram à margem de 6% a depender do modelo.

Dessa forma, reaproveitar as amostras excluídas no treinamento pela SLA é uma forma consistente de implementar os resultados dos modelos do estado da arte que utilizam a SLA ao lidar com amostras ruidosas. Como já mencionado, a técnica RDS, essencialmente, possibilita que a rede aprenda características importantes do conjunto dos dados que estavam sendo excluídas pela SLA. Os resultados obtidos reforçam a

possibilidade dessas características serem importantes para o aprendizado da rede, pois todos os modelos tiveram ganhos ao utilizar o RDS.

Outro importante indicativo dos resultados, apontado pela métrica Relabel Total, é que a quantidade de amostras recuperadas contribui para o aprendizado, estando de acordo com a notória observação que modelos de DL necessitam de grande quantidade de dados no processo de treinamento. Além disso, é essencial que essas amostras contenham *pseudolabels* corretos, pois *pseudolabels* corretos vão guiar a rede a atribuir as características novas aprendidas as classes corretas. Sendo então completamente indesejável os *pseudolabels* incorretos, pois estes são equivalentes a ruídos inseridos a rede e guiam a rede a atribuir as novas características aprendidas as classes erradas.

Esses pontos evidenciam que quanto mais amostras recuperadas corretamente melhor o desempenho do modelo. Idealmente, espera-se que todas as amostras tenham *pseudolabels* corretos e que todas as amostras excluídas pela SLA sejam recuperadas no processo. Esse cenário seria equivalente a realizar um treinamento livre de amostras ruidosas. Então, pode-se concluir que a técnica RDS aproxima os modelos treinados com *datasets* ruidosos a *datasets* sem ruídos e de forma ideal levaria a um treino livre de ruídos.

Segundo os resultados, o modelo RDS-Contrastive, em alguns cenários, é mais robusto a ruídos em relação aos outros modelos aqui abordados, isso é constatado pois apesar de resultados similares a modelos como RDS-C e RDS-J nas métricas Acurácia RDS e Relabel Total, o desempenho final foi superior na acurácia de teste. Para compreender esse comportamento é preciso ressaltar que tanto a técnica SLA e a RDS não são livres de ruído, sendo a quantidade de ruído retornado ao treino mensurado pela métrica Acurácia RDS. Por isso, é importante que o procedimento de treinamento nas amostras limpas seja robusto a ruídos e o modelo RDS-Contrastive se mostrou eficaz em diversos cenários, superando o RDS-J e o RDS-C.

6.6. Resultados Dataset Clothing1M

Nesse tópico apresentamos a performance dos modelos RDS-C, RDS-J e RDS-Contrastive sobre o *large-scale real-world dataset*: Clothing1M. Este *dataset* é amplamente utilizado no SOTA para comparar os diferentes tipos de modelo desenvolvidos para lidar com amostras ruidosas. Ele consiste 10^6 amostras com 14

classes com ruído aproximado de 40%. O ruído presente neste *dataset* é da categoria *open-set* [22], ou seja, existem amostras que não pertencem a nenhuma das classes em treinamento presentes no *dataset*.

Na Tabela 8 apresentamos os resultados da performance. Observa-se que o nosso melhor modelo foi o RDS-J sendo superado somente pelo modelo do SOTA Jo-SRC. O modelo Jo-SRC é voltado para ruído em *datasets open-set*, sendo especificamente voltado para o dataset Clothing1M. Apesar desse modelo ser útil para esse tipo de *dataset*, ele contém muitos hiperparâmetros dependentes, sendo então, inapropriados para aplicações reais. Portanto, nossos modelos, apresentaram resultados sólidos, demostrando robustez contra esse tipo de ruído, mesmo não sendo este o foco desta tese.

Tabela 8 Comparação do desempenho dos modelos no Dataset Clothing1M. Resultados diretamente extraídos do trabalho [22].

Modelo	**Acurácia de Teste**	**BackBone**
RDS-C	70.60 %	Resnet18
RDS-J	71.10 %	Resnet18
Jo-SRC	**71.78 %**	**Resnet18**
Standard	67.22 %	Resnet18
Decoupling	68.48 %	Resnet18
Co-teaching	69.21 %	Resnet18
Co-teaching+	59.32 %	Resnet18
JoCoR	70.30 %	Resnet18
RDS-Contrastive	70.28 %	Resnet18
F-Correction	68.93	Resnet 18

6.7. Análise de *Data Augmentation n* e o *Threshold* μ

Nessa seção é analisado como os modelos se comportam em relação a variação dos hiperparâmetros de *Data Augmentation n* e *Threshold* μ, sobre as métricas acurácia de teste, Acurácia RDS e Relabel Total. A análise foi feita sobre o *dataset* Cifar-100 com ruído simétrico t=50, utilizando-se da mesma rede descrita no capítulo 5 e demais hiperparâmetros. Nos experimentos realizados, o *data augmentation* variou sobre os valores *n=1, n=2 e n=4.*E o *threshold* variou de $\mu = 0.6$, $\mu = 0.8$ e $\mu = 0.96$.

No experimento 1 utilizou-se o modelo RDS-C, onde foi treinado 3 redes. Em todas as redes manteve-se fixo o $\mu = 0.80$ e variou-se o *n* em: *n=1, n=2 e n=4.* Os detalhes do experimento 1 estão indicados na Tabela 9.

Tabela 9 Cifar-100, model RDS-C, resultados para os hiperparâmetros: μ = 0.80 e n = 1, 2, 4

Model	*Época*	*μ*	*n*	*Acurácia teste*	*Acurácia RDS*	*Relabel*
RDS-C	*150*	*0.80*	*1*	*0.3077*	*0.3611*	*20259*
RDS-C	*150*	*0.80*	*2*	*0.3004*	*0.4047*	*16333*
RDS-C	*150*	*0.80*	*4*	*0.26175*	*0.7178*	*484*

As redes treinadas com n=1 e n=2 tiveram resultados em relação a acurácia de teste similares sendo o n=1 superior, enquanto a rede treinada com n=4 teve um resultado inferior a ambos. Em relação a Acurácia RDS a rede com n=4 teve um desempenho superior à das redes com n=1 e n=2 por uma margem considerável. Na métrica Relabel a rede com n=4 teve um desempenho abaixo das demais.

Conclui-se do experimento 1 que o aumento do valor de *n* tende a aumentar a confiança dos *pseudolabels* (alta acurácia RDS), entretanto ocorre uma redução dos valores de *pseudolabels* validados no processo do RDS (baixo Relabel Total). Além disso, observa-se que o desempenho final do modelo depende tanto da acurácia RDS quanto do Relabel Total.

Dessa forma, é preciso buscar um ponto ótimo entre o aumento de *pseudolabels* corretos atribuídos pelo modelo RDS e a redução de amostras válidas ao seu escolher o hiperparâmetro *n.* A busca desse ponto ótimo é importante, pois segundo as observações dos experimentos anteriores o aumento de amostras recuperadas contribui para o aprendizado do modelo, entretanto é desejável que estes tenham pseudolabels corretos. Pois o aumento de pseudolabels incorretos guiam o modelo a associar as novas características aprendidas as classes incorretas.

No experimento 2 utilizou-se novamente o modelo RDS-C. Onde foi treinado 3 redes. Em todas as redes manteve-se fixo o $n = 4$ e variou-se o μ em $\mu = 0.6$, $\mu = 0.8$ e $\mu = 0.96$. Os detalhes dos experimentos 2 estão indicados na Tabela 10.

Tabela 10 Cifar-100, model RDS-C, resultados para os hiperparâmetros: μ = 0.6, 0.8, 0.96 e n = 4

Model	*Época*	*μ*	*n*	*Acurácia teste*	*Acurácia RDS*	*Relabel*
RDS-C	*150*	*0.60*	*1*	*0.2991*	*0.4865*	*7996*
RDS-C	*150*	*0.80*	*2*	*0.2617*	*0.7178*	*484*
RDS-C	*150*	*0.96*	*4*	*0.2660*	*0.8778*	*130*

Nesse cenário a rede com $\mu = 0.60$ teve o melhor desempenho em relação a Acurácia teste e as redes com $\mu = 0.8, 0.96$ com resultados similares. Com tabela 2

observa-se uma relação similar ao Experimento 1. As redes com alto desempenho na Acurácia RDS tiveram baixo desempenho no Relabel Total e consequentemente na acurácia de teste.

Com o Experimento 2 observa-se que o aumento do μ aumenta a confiança da rede nos pseudolabels. Sendo o mesmo efeito observado com o aumento do *n*. Entretanto também ocorre uma tendência de redução no Relabel total.

O Experimento 1 e 2 indicam que o Relabel Total é importante para o resultado do modelo. Dessa forma, é preciso buscar um ponto ótimo entre o alto grau de pseudolabels corretos e o alto número de Relabel total. Pois tanto o aumento de *n* e μ tendem a aumentar a Acurácia RDS, porém também reduzem o Relabel Total.

Assim, a escolha dos hiperparâmetros *n* e μ estão diretamente associados ao controle da quantidade de amostras retornadas ao treino e da confiança dos pseudolabels dessas amostras. Em que, idealmente, busca-se recuperar todas as amostras excluídas pela SLA, além de todos os pseudolabels corretos, chegando a um cenário equivalente a um treinamento com dataset limpo.

No experimento 3, utilizou-se o modelo RDS-J. Onde foi treinado 3 redes. Em todas as redes manteve-se fixo o $\mu = 0.96$ e variou o *n* em: *n=1, n=2 e n=4.* Os detalhes dos experimentos 3 estão indicados na Tabela 11.

Tabela 11 Cifar-100, model RDS-C, resultados para os hiperparâmetros: μ = 0.80 e n = 1, 2, 4

Model	*Época*	μ	*n*	*Acurácia teste*	*Acurácia RDS*	*Relabel*
RDS-J	*150*	*0.80*	*1*	*0.3099*	*0.4503*	*13505*
RDS-J	*150*	*0.80*	*2*	*0.3077*	*0.5187*	*10593*
RDS-J	*150*	*0.86*	*4*	*0.2891*	*0.7248*	*510*

As redes treinadas com n=2 e n=1 tiverem resultados similares em relação a Acurácia teste e a rede com n=4 teve um desempenho inferior. Os resultados do Experimento 3 seguem a mesma linha do Experimento 1, onde que o aumento do *n* resulta em um aumento da Acurácia RDS, entretanto também ocorre uma redução do Relabel Total.

No experimento 4 utilizou-se o modelo RDS-J. Onde foi treinado 3 redes. Em todas as redes manteve-se fixo o $n = 4$ e variou-se o μ em $\mu = 0.6$, $\mu = 0.8$ e $\mu = 0.96$. Os detalhes dos experimentos 4 estão indicados na Tabela 12

Tabela 12 Cifar-100, model RDS-C, resultados para os hiperparâmetros: μ = 0.6, 0.8, 0.96 e n = 4

Model	*Época*	*μ*	*n*	*Acurácia teste*	*Acurácia RDS*	*Relabel*
RDS-J	*150*	*0.60*	*4*	*0.3066*	*0.5867*	*5504*
RDS-J	*150*	*0.80*	*4*	*0.2891*	*0.7248*	*510*
RDS-J	*150*	*0.96*	*4*	*0.2885*	*0.8394*	*205*

Os resultados do Experimento 4 refletem os mesmos resultados do Experimento 2. O aumento do μ aumenta a confiança da rede nos pseudolabels, assim como o aumento do *n*, entretanto também ocorre uma tendência de redução no Relabel total.

Os resultados dos Experimentos 1,2,3 e 4 indicam uma tendência de aumento da Acurácia RDS com o aumento μ e do *n*, entretanto também há uma tendência de redução do Relabel Total. É preciso buscar um ponto de equilibro entre a alta performance da Acurácia RDS, e o valor do Relabel Total, pois os resultados também indicam que tanto a Acurácia RDS, quanto o Relabel Total influenciam no desempenho da acurácia de teste.

Frente a essas observações registra-se a possibilidade de trabalho futuro: buscar técnicas mais eficientes para substituir o RDS-Relabel com o objetivo de retornar melhores valores na Acurácia RDS e Relabel Total. Visando chegar o mais próximo possível de um cenário ideal, em que todas as amostras excluídas pela SLA sejam recuperadas com pseudolabels corretos. Por fim, os resultados obtidos reforçam as observações que o aumento de amostras recuperadas é benéfico ao treinamento do modelo, pois permite o aprendizado de novas características do banco de dados. Entretanto, é preciso que estas amostras tenham pseudolabels corretos, permitindo assim que as novas características aprendidas sejam associadas as classes corretas.

6.8. Estudo de Caso Real

É notória a importância da preservação ambiental, o que inclui a regeneração e restauração dos mais diversos ecossistemas, visando recuperar ecossistemas que foram degradados ou destruídos pela ação humana. Ao longo da exploração de petróleo e gás natural em águas profundas na orla brasileira, ocorreram impactos no ecossistema das algas calcárias [24]. Visando a restauração dessas espécies marinhas a empresa Petrobras adotou uma série de medidas de sustentabilidade para esse setor. Uma dessas medidas é realizar um levantamento da população atual de algas calcárias

presentes no assoalho marinho, outras medidas podem ser encontradas no Relatório de Sustentabilidade de 2021 da Petrobras[1].

Para realizar o levantamento da região com algas calcárias a empresa solicitou o desenvolvimento de uma ferramenta para realizar a classificação automática das algas calcárias presentes no assoalho marinho a partir de imagens coletadas por um Veículo Operado Remotamente (ROV, do inglês *Remotely Operated Vehicle*). A ferramenta solicitada deve ser capaz de identificar o tipo de alga calcária presente na imagem, sendo os tipos: Rodolito, BioConcreção, Granulado e Laje.

6.8.1. Motivação

A motivação para solicitação do desenvolvimento deste classificador surgiu da enorme quantidade de imagens necessárias para classificação. Segundo os especialistas do setor, a quantidade de imagens a serem classificadas supera a casa de 1 milhão e o número continua a crescer com novas coletas de dados pelo ROV.

O uso de um modelo de *DL* para essa tarefa reduz o custo humano no trabalho, pois é necessário o gasto com especialistas. Além disso, o modelo de *DL* consegue realizar a classificação das imagens muito mais rápido do que os especialistas, permitindo que estes trabalhem em outros pontos do projeto de suma importância. Outro objetivo da empresa é instalar o modelo de *DL* diretamente no ROV permitindo inferências em tempo real, acelerando o processo de inspeção e levantamento das espécies de algas calcárias no assolho marinho.

6.8.2. Base de Dados de Algas Calcárias

Algas calcárias são compostas basicamente de carbonato de cálcio e carbonato de magnésio, sendo utilizadas para diversas aplicações: agricultura (maior volume), potabilização de águas para consumo, indústria de cosméticos, dietética, implantes em cirurgia óssea, nutrição animal e tratamento da água em lagos [24]. Os diferentes tipos

[1] Relatório de Sustentabilidade 2021 | Petrobras

de formações de algas calcárias de interesse são Rodolito, BioConcreção, Granulado e Laje. Uma descrição de cada um deles é apresentado na Tabela 13 abaixo.

Tabela 13 Descrição dos tipos de Algas Calcárias

Algas Calcárias	
Tipo	**Características**
Granulado	Nódulos pequenos, aglomerados ou esparsos, com diâmetros ≤ 3 cm;
Rodolitos	Nódulos esféricos, aglomerados ou esparsos, com diâmetros > 3 cm;
Laje	Maciças, contínuas ou intercaladas com sedimento;
Bioconcreção	Maciças contínuas ou intercaladas com sedimento, porém diferenciam-se das lajes por possuírem uma maior complexidade tridimensional, principalmente em altura.

A distribuição das classes no *dataset* está apresentando na Figura 41:

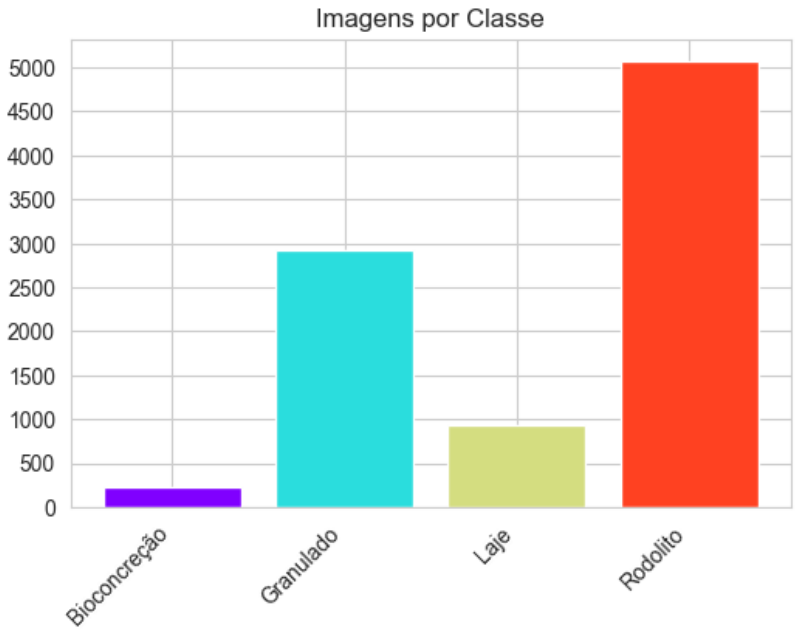


Figura 41 Distribuição de amostras do dataset por classe

6.8.3. Restrições de Dados

A base de dados utilizada no trabalho é confidencial. Além disso, não foi liberada a publicação dos resultados realizados nos testes em campo pelo modelo

implementado, nem os detalhes da implementação. Dessa forma serão apresentados apenas os resultados internos do modelo.

6.8.4.
Ruído no Dataset

Ao longo do trabalho, a equipe observou a presença de amostras ruidosas na base de imagens fornecida, o que, posteriormente, foi confirmado por especialistas. Frente a isso, utilizou-se técnicas desenvolvidas nessa proposta de tese para o treinamento do modelo de *DL*. O ruído estimado pela equipe foi de 10%.

Observou-se que a confusão entre as classes ruidosas se dava prioritariamente entre as classes Laje e Rodolito devido à proximidade entre as classes e entre Rodolito e Granulado em menor grau, conforme a Figura 42.

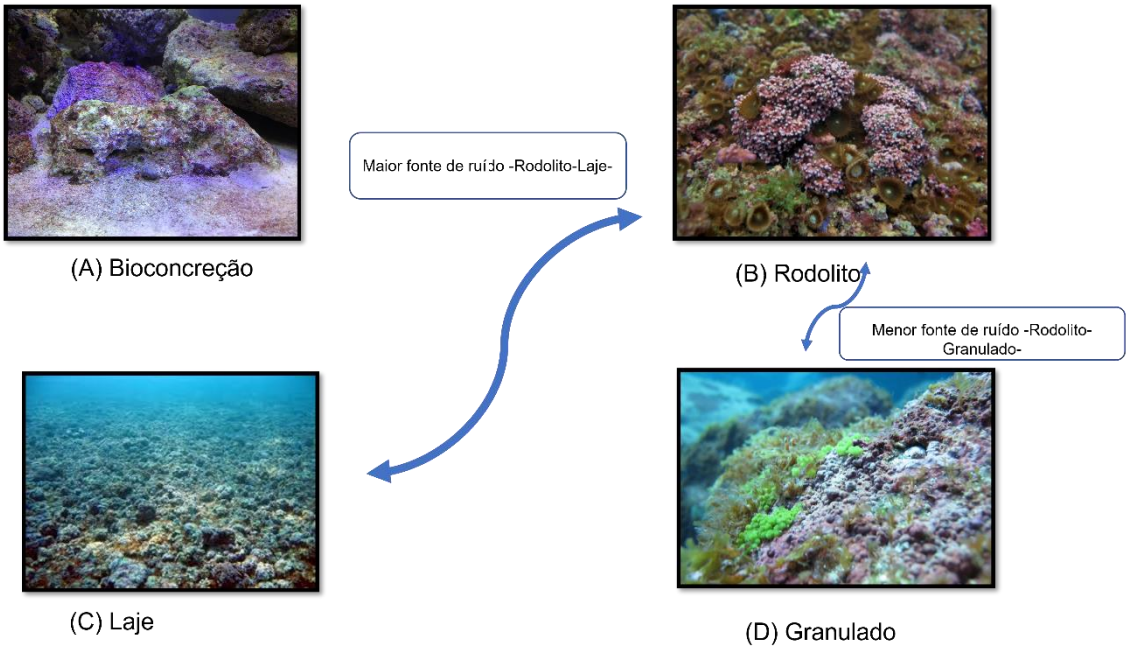


Figura 42 Confusão entre as classes ruidosas (imagens disponíveis livremente na internet)

6.8.5.
Detalhes Treinamento

Para justificar o uso dos modelos desenvolvidos, realizou-se uma comparação no desempenho dos modelos RDS-C, RDS-J e RDS-Contrastive em relação aos modelos

do SOTA: Co-teaching+, Decoupling, Jocor, SAT, Standard (STD), ou seja, um modelo de DL treinado sem nenhuma técnica para lidar com amostras ruidosas.

Para essa comparação utilizou-se a rede RESNET-50 [81]. A taxa de aprendizado η utilizada foi de 0.00001, o otimizador utilizado ao longo do treino foi o Adam, com momentum 0.9. O hiperparâmetro T_k utilizado na função *R(t)* - equação 31 - foi de $T_k = 10$. O número de épocas utilizado foi de 80, o número de data augmentation *n* foi de 2 e o threshold μ utilizado foi de 0.80 e o batch-size 32.

Para realizar avaliação dos modelos utilizou-se o F1-Score sobre o conjunto teste. Seguindo o protocolo dos trabalhos [5] [14] [15] , separou-se a base de dados em treino (90%) e teste (10%). O conjunto teste foi amplamente revisado para garantir a presença apenas amostras limpas. A distribuição de amostras seguiu conforme a Figura 43 para o conjunto treino, onde este contém amostras ruidosas; e para o conjunto teste conforme a Figura 44 com amostras somente limpas.

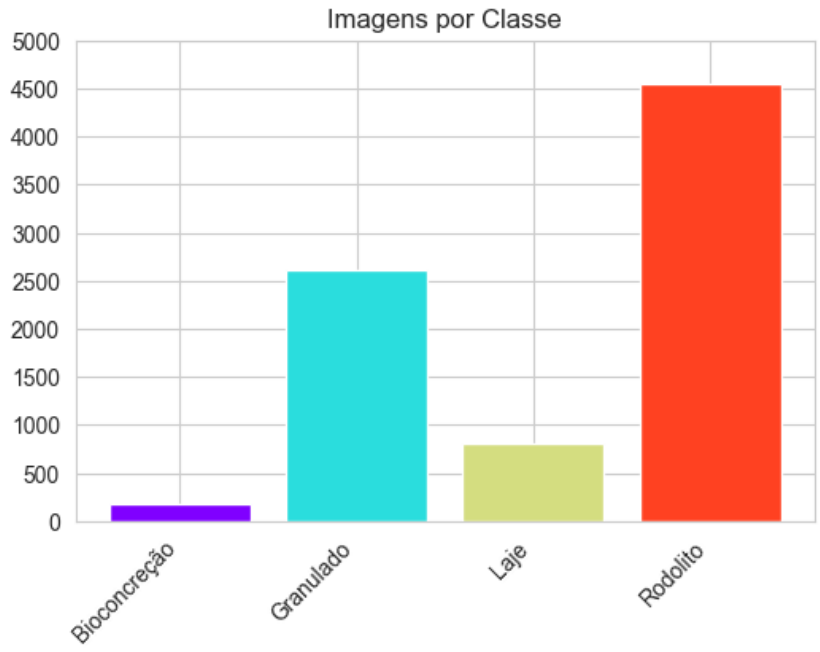


Figura 43 Distribuição de amostras por classe do conjunto treino da base de algas calcárias

Figura 44 Distribuição de amostras por classe do conjunto de teste da base de algas calcárias

6.8.6. Resultados

Na Tabela 14 estão apresentados os melhores resultados obtidos pelos modelos. Observa-se que o modelo RDS- Contrastive teve o melhor resultado.

Tabela 14 Resultados de F1 Score, Recall e Precision para o Estudo de Caso

Modelo	F1-Score	Recall	Precision	Acurácia
Co-teaching +	0.8030	0.7558	0.8766	0.8414
Decoupling	0.8144	0.7831	0.8721	0.8409
Jocor	0.8332	0.7973	0.8860	0.8737
RDS-C	0.8444	0.8385	0.8551	0.8469
RDS-Contrastive	**0.8611**	**0.8718**	**0.8552**	**0.8618**
RDS-J	0.8333	0.7968	0.9023	0.8727
SAT	0.7700	0.7188	0.8785	0.8316
Standard	0.8139	0.7726	0.8758	0.8489

A melhora do desempenho em relação ao modelo *standard* é um indicativo que as amostras ruidosas estão de fato tendo uma influência negativa no aprendizado do modelo. O modelo o *standard*, por não realizar nenhum tratamento especial em cima dessas amostras, acaba por “aprender” as amostras com os rótulos errados

prejudicando a generalização. Já o modelo RDS-Contrastive, ao eliminar as amostras ruidosas e corrigir o rótulo de uma parcela dessas amostras através da técnica RDS, acaba por ter um desempenho melhor. Os outros modelos RDS-C e RDS-J também apresentaram ganho de desempenho, porém abaixo do RDS-Contrastive. A melhora de desempenho pode ser explicada pelo termo contrastive estar atuando em conjunto com o RDS. Este *dataset* apresenta características complexas e este termo ajuda o modelo a extrair mais detalhes do conjunto de dados.

Na Figura 45 está apresentada a matriz confusão referente ao modelo RDS-Contrastive sobre o conjunto teste. Observa-se que o modelo foi capaz de aprender características de todas as classes. Como se trata de um conjunto de dados desbalanceado, esse resultado aponta que o modelo é capaz de aprender a diversidade das classes mesmo em cenários ruidosos desbalanceados.

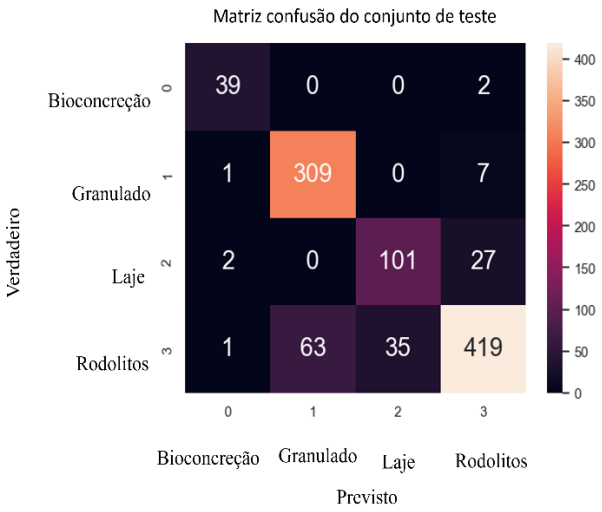


Figura 45 Matriz confusão do Modelo RDS-Contrastive sobre o conjunto teste

6.9. Resultados para o Problema Multilabel

Nessa seção serão apresentados os resultados dos modelos Multilabel. Em todos os gráficos dessa seção a linha sólida representa o resultado médio da referida métrica obtida pelos modelos ao longo das épocas de treinamento. A região sombreada da mesma cor da curva média representa o desvio padrão da métrica referida obtida pelos modelos ao longo das épocas.

Na Figura 46, está apresentado o gráfico do F1-Score referente ao *dataset* UcMerced para todos os modelos em comparação. O modelo SLAM-JL (curva azul) apresentou um ganho de performance de 17% quando comparado com outros modelos do SOTA e de 2.4% quando comparado com o modelo SLAM (curva verde) também desenvolvido nessa tese.

Na Figura 47, está apresentado o gráfico da métrica Acurácia SLA Multilabel para todas as classes durante o período de treino. No gráfico, a linha sólida preta representa a porcentagem inicial de amostras limpas para todas as classes, do qual é equivalente a 75%, uma vez que a porcentagem de ruído introduzido para todas as classes é de 25 %. Quando as outras curvas, referentes as classes em treino, estão acima dessa linha preta indica que o modelo está reduzindo a porcentagem de ruído presente no dataset para esta classe. Por outro lado, uma curva abaixo dessa linha indica que o modelo está aumentando a porcentagem de ruído presente no dataset para a classe em questão.

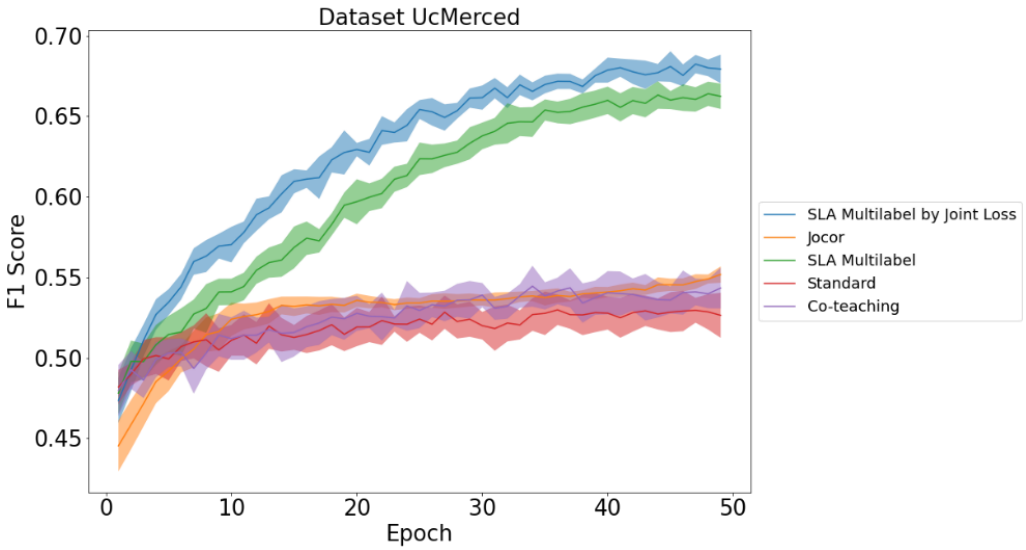


Figura 46 F1-Score por época sobre o conjunto teste para o dataset UcMerced

Ao longo das épocas de treino observa-se que o modelo está significativamente reduzindo o ruído presente no dataset. O modelo SLAM-JL aumentou a porcentagem de amostras limpas presentes na margem de 95% para a maioria das classes, apenas para as classes “*water*”, “*tree*” e “*grass*” ocorreu aumento de ruído. Esse comportamento

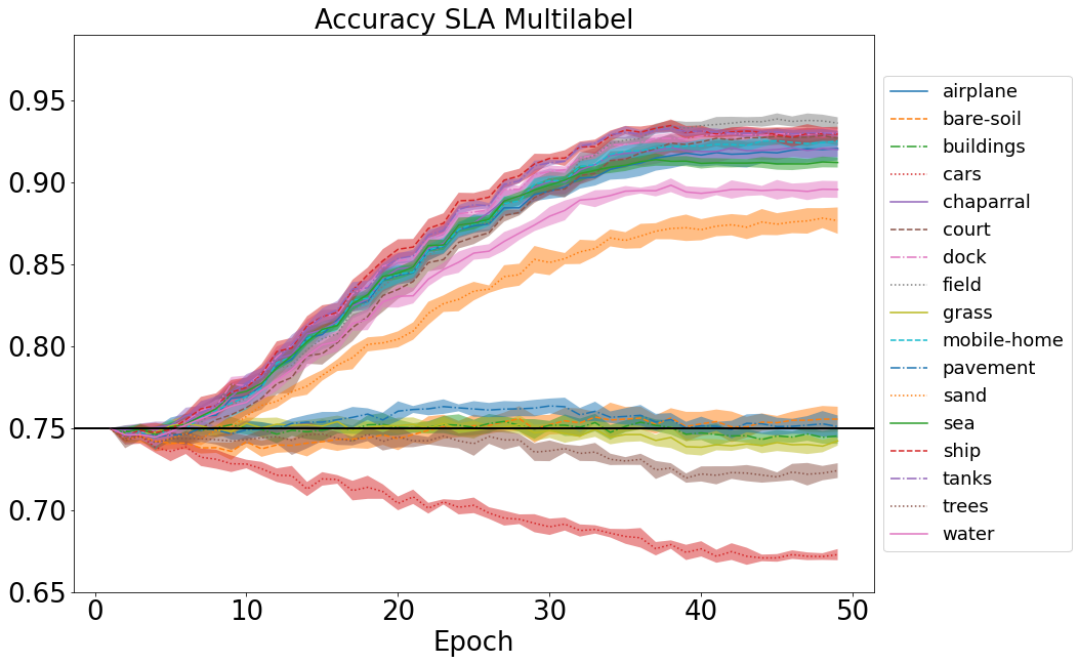


Figura 47 Acurácia SLA Multilabel Para o Dataset UcMerced referente ao modelo SLAM-JL

apresentado pelo modelo continua sendo um ponto de pesquisa em aberto. Intuitivamente, existe a possibilidade de uma conexão com o desbalanceamento das classes do dataset, como ilustrado na Figura 48. Entretanto, este comportamento não foi observado nos experimentos realizados nos demais dataset abordados nessa tese que também são desbalanceados.

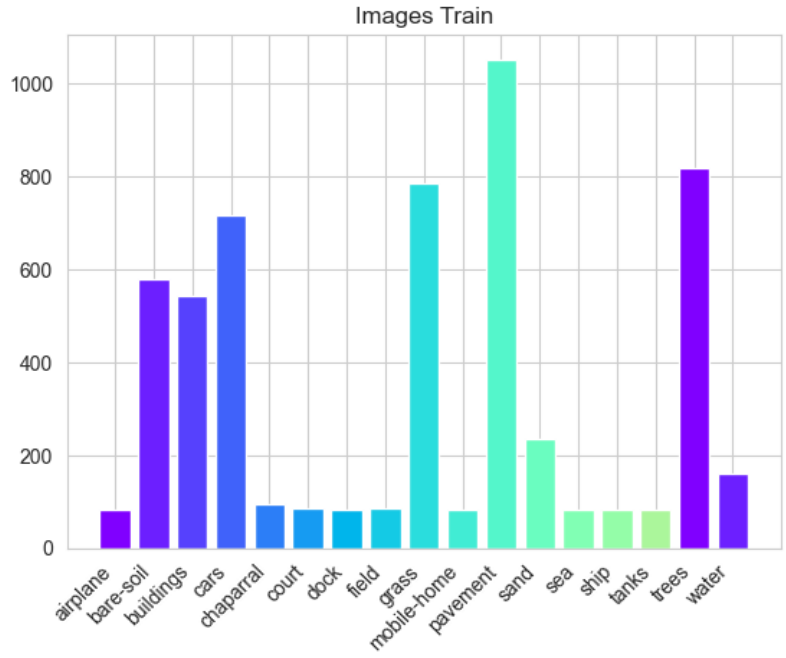


Figura 48 Balanceamento das Classes no Dataset UcMerced

Um efeito similar foi observado ao analisar a métrica Acurácia SLA Multilabel para o modelo SLAM sobre o *dataset* UcMerced, conforme Figura 49. As classes com pior desempenho referentes ao modelo SLAM, nesta métrica, são as classes mais comuns no *dataset* UcMerced. Frente a isso, observa-se uma possibilidade das classes mais comuns em *dataset* multilabel serem mais sensíveis a ruído em cenário multilabel, contudo este ainda é um resultado inconclusivo, uma vez que não se repete em outros experimentos.

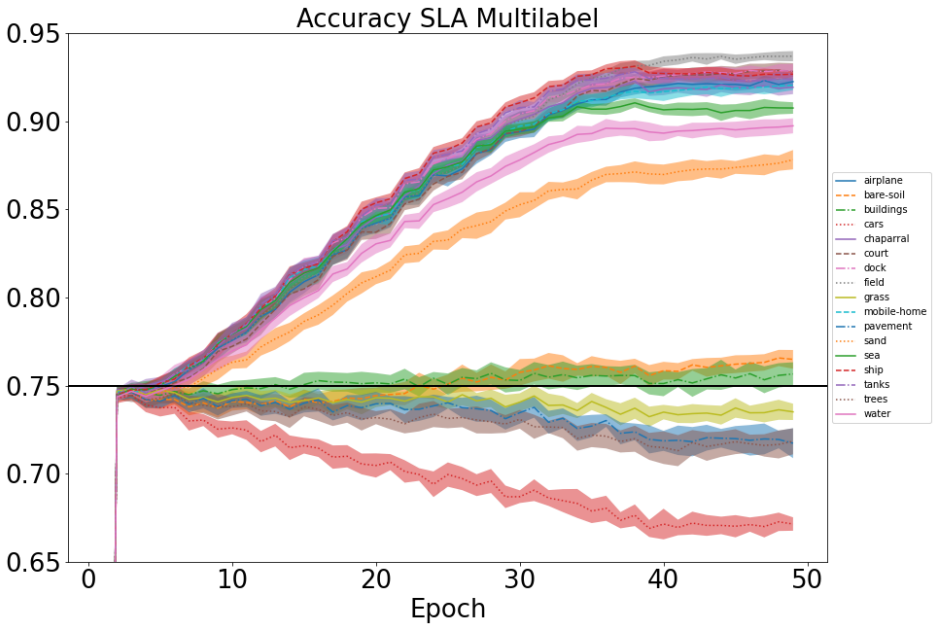


Figura 49 Acurácia SLA Multilabel Para o Dataset UcMerced referente ao modelo SLAM-JL

Na Tabela 15, apresentamos os resultados de performance F1-Score dos modelos em relação ao dataset TreeSatAI. A performance dos modelos SLAM-JL e SLAM foram similares, sendo superior aos modelos do SOTA em 6 %. Na Figura 50, apresentamos os gráficos da métrica Acurácia SLA Multilabel para os modelos SLAM-JL e SLAM referente ao dataset TreeSatAI. Observa-se que os modelos reduziram a presença de ruído para todas as classes, não repetindo o efeito observado no dataset UcMerced.

Tabela 15 F1-Score para os modelos em comparação sobre o dataset TreeSatAI. Resultados referentes a época final de treino.

Modelo	Época	F1-Score
SLAM	30	39.18%
SLAM-JL	30	39.30%
Jocor	30	23.00%
Co-teaching	30	33.00%
Standard	30	32.01%

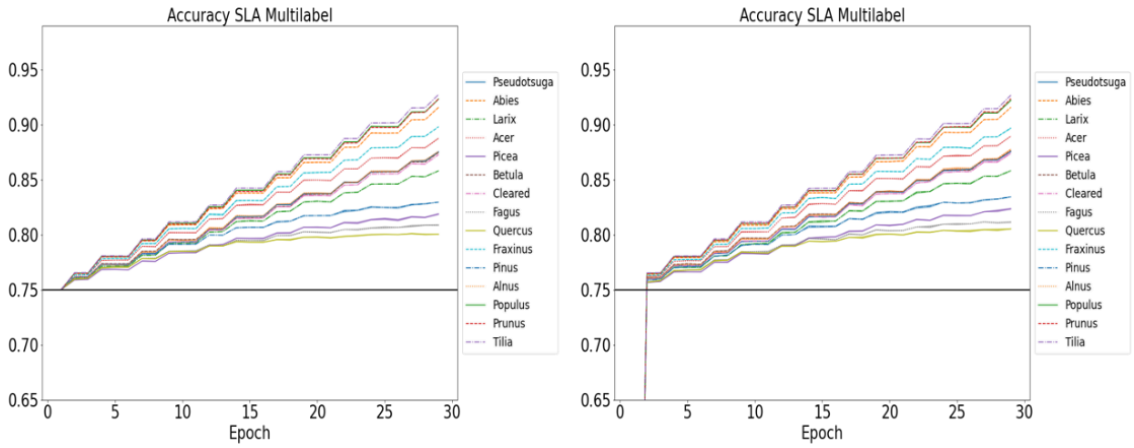


Figura 50 Acurácia SLA Multilabel para os modelos SLAM-JL e SLAM. O gráfico da esquerda é referente ao modelo SLAM-JL e o da direita ao SLAM

O dataset TreeSatAi também é desbalanceado, conforme Figura 51. Dessa forma, a hipótese de o desbalanceamento das classes reduzir a performance dos modelos na métrica Acurácia SLA MultiLabel é inconclusiva.

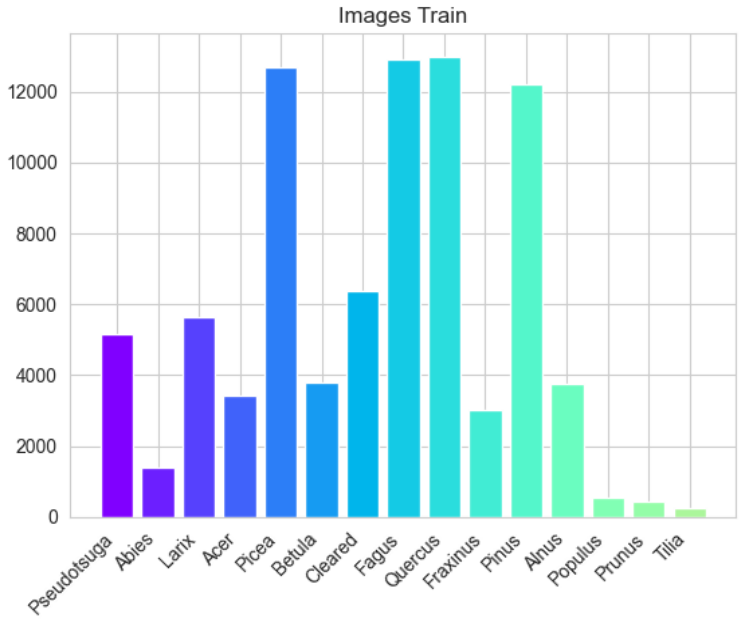


Figura 51 Balanceamento das Classes no Dataset TreeSatAi

Os resultados apresentados reforçam que a técnica SLAM é eficaz no tratamento de amostras ruidosas no cenário multiclasse. Os ganhos em relação aos outros modelos do SOTA foram de até 17%, demostrando a robustez da técnica desenvolvida. O modelo SLAM-JL teve uma performance melhor sobre a métrica Acurácia SLA Multilabel e esse ganho de performance é refletido no F1-Score.

6.9.1. Análise de Sensibilidade T_k SLAM

Nesta seção analisamos a performance dos modelos sobre a métrica F1-Score em função do hiperparâmetro $\boldsymbol{T_k}$. Para o cenário multiclasse o trabalho em [5] apontou $\boldsymbol{T_k = 10}$ como o valor mais adequado. Na Tabela 16 apresentamos os resultados do F1-Score no conjunto teste referente ao Dataset TreeSatAI para os modelos SLAM e SLAM-JL, treinados com $\boldsymbol{T_k = 10, 20, 30, 40, 50}$.

Observa-se na Tabela 16 que a performance dos modelos pouco varia sobre o hiperparâmetro $\boldsymbol{T_k}$. Sendo os valores $\boldsymbol{T_k = 30{,}40}$ ligeiramente superior aos outros resultados.

Tabela 16 F1-Score para os SLAM e SLAM-JL sobre o dataset TreeSatAI. Resultados referentes a época final de treino (30). Hiperparâmetro $T_k = 10{,}20{,}30{,}40$

Modelo	T_k	F1-Score
SLAM	10	39.06%
SLAM-JL	10	39.10%
SLAM	20	39.06%
SLAM-JL	20	39.11%
SLAM	30	39.19%
SLAM-JL	30	39.29%
SLAM	**40**	39.18%
SLAM-JL	**40**	39.30%
SLAM	50	39.09%
SLAM-JL	50	39.26%

6.9.2. Análise de Sensibilidade $start_{epoch}$ SLAM

Nesta seção analisamos a performance dos modelos sobre a métrica F1-Score em função do hiperparâmetro $\boldsymbol{start_{epoch}}$. Como mencionado este hiperparâmetro visa permitir que a rede aprenda características bases do conjunto de dados antes de atribuir novos rótulos as amostras ruidosas. Na Tabela 17 apresentamos os resultados do F1-Score no conjunto teste referente ao Dataset TreeSatAI para os modelos SLAM e SLAM-JL, treinados com $\boldsymbol{start_{epoch} = 1, 5, 10, 15.}$ Observa-se resultados similares de performance, sendo $\boldsymbol{start_{epoch}}$**=1,5** ligeiramente melhores.

Tabela 17 F1-Score para os SLAM e SLAM-JL sobre o dataset TreeSatAI. Resultados referentes a época final de treino (30). Hiperparâmetro $start_{epoch} = 1{,}5{,}10{,}15$

Modelo	$start_{epoch}$	F1-Score
SLAM	**1**	39.18%
SLAM-JL	**1**	39.30%
SLAM	**5**	39.18%
SLAM-JL	**5**	39.30%
SLAM	10	39.17%
SLAM-JL	10	39.29%
SLAM	15	39.05%
SLAM-JL	15	39.13%

6.9.3. Estudo de Caso Multilabel

A empresa Petrobras vem realizando um estudo em conjunto com o ICA para automatizar a inspeções de dutos utilizados na Bacia de Campos. A inspeção atualmente é realizada com o auxílio de um submarino ROV com o qual imagens dos dutos e equipamentos são coletadas. Essas imagens são analisadas com o objetivo de classificar as condições dos dutos para assinalar futuros reparos e manutenções necessárias.

Atualmente um *dataset* Multilabel está sendo elaborado para automatizar o processo de inspeção através de um modelo de *DL*. As imagens coletadas, nas inspeções antigas, estão sendo utilizadas para a elaboração do dataset onde são possíveis a presença de até 50 classes simultaneamente (cenário multilabel). Um exemplo desse cenário está apresentando na Figura 52, na figura é apresentado um exemplo fictício do *dataset* com classes as reduzidas, em que na imagem estão presentes três classes simultaneamente, *pipeline, rope e end fitting*.

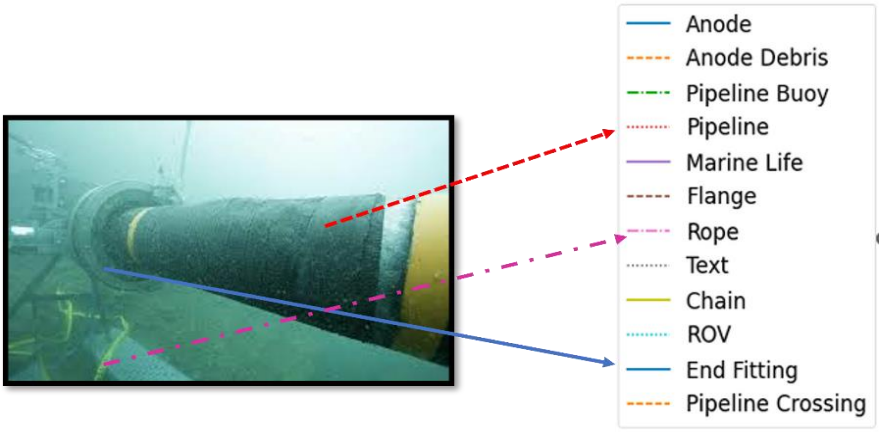


Figura 52 Exemplo de Dataset Multilabel de Inspeções Submarinas. Na imagem estão presentes simultaneamente três classes: Pipeline, Rope e End Fitting

Devido ao grande número de classes a serem anotadas e sua complexidade, esse *dataset* apresentou amostras ruidosas. Foi realizado um experimento em um conjunto de dados reduzidos do *dataset* contendo 12 classes e 1200 amostras para avaliar a viabilidade de utilizar os modelos desenvolvidos nessa tese. Esse dataset foi chamado de *Underwater Inspections Dataset*. As classes do dataset são: 'Anode', 'Anode Debris', 'Pipeline Buoy', 'Pipeline', 'Marine Life', 'Flange', 'Rope', 'Text', 'Chain', 'End Fitting', 'Pipeline Crossing'. O conjunto reduzido de 1200 amostras teve suas anotações cuidadosamente verificadas para garantir apenas rótulos limpos. Na Figura 53, está apresentado a distribuição das classes do *dataset*.

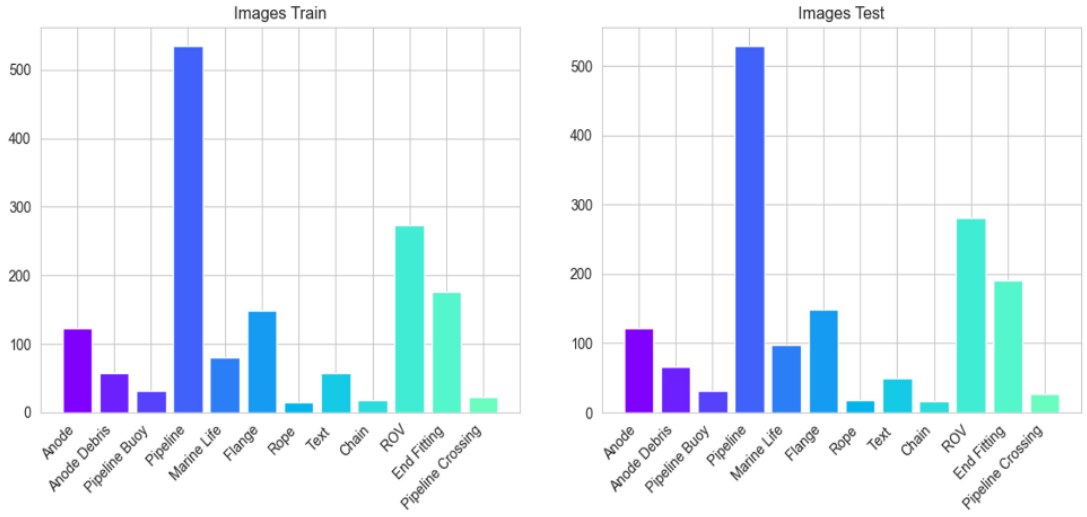


Figura 53 Distribuição das classes no dataset Underwater Inspections. A direita, o conjunto treino e a esquerda, o conjunto teste

O experimento consistiu em inserir 25 % de ruído em todas as classes de forma mixed [65] como realizado na elaboração dos *datasets* UcMerced e TreeSatAI no conjunto treino. Os modelos comparados foram os mesmos utilizados nos *dataset* TreeSatAI e UcMerced. Onde que a métrica utilizada foi o F1-Score em cima do conjunto teste ao longo das épocas, em que o conjunto teste consiste apenas de amostras limpas.

O *backbone* utilizado para avaliar os modelos e o pré-processamento nas amostras foi o mesmo dos experimentos dos datasets UcMerced e TreeSatAI. O *learning rate* foi de 0.01, o número de épocas foi de 200, $T_k = 30$, $start_{epoch} = 5$, o ruído estimado foi de 0.25, batch size 32 e o otimizador foi o Adam.

Na Figura 54 estão apresentados os resultados do F1-score. Os resultados obtidos reforçam os bons resultados dos nossos modelos encontrados no *dataset* UcMerced multilabel e TreeSatAI. Novamente, observa-se que o modelo SLAM-JL teve o melhor desempenho seguido do modelo Jocor multilabel. O modelo SLAM teve resultado inferior ao modelo Jocor.

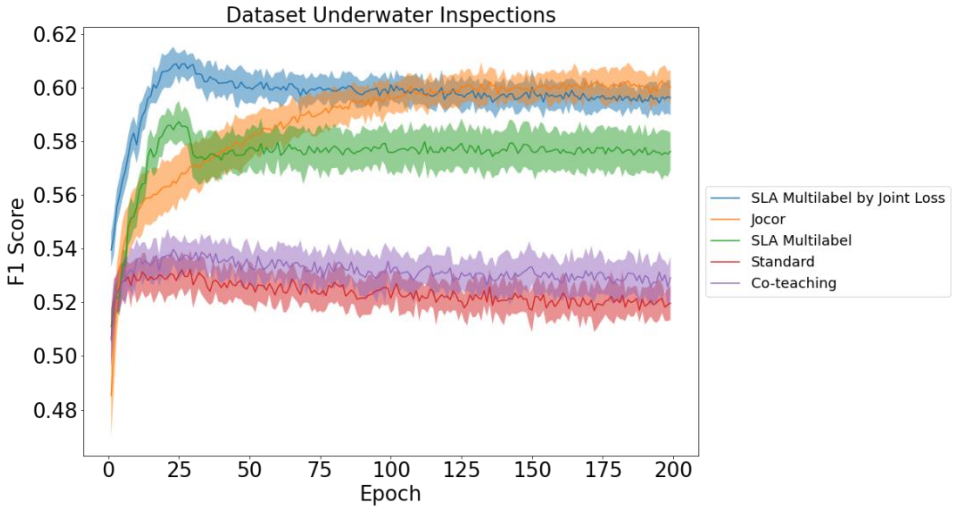


Figura 54 Métrica F1-Score para o Dataset Underwater Inspections

Observa-se que o pico de performance dos nossos modelos SLAM-JL e SLAM (linha azul e linha verde) ocorrem na época 25. É interessante notar que este pico coincide com o pico de performance na métrica Acurácia SLA Multilabel (Figura 55).

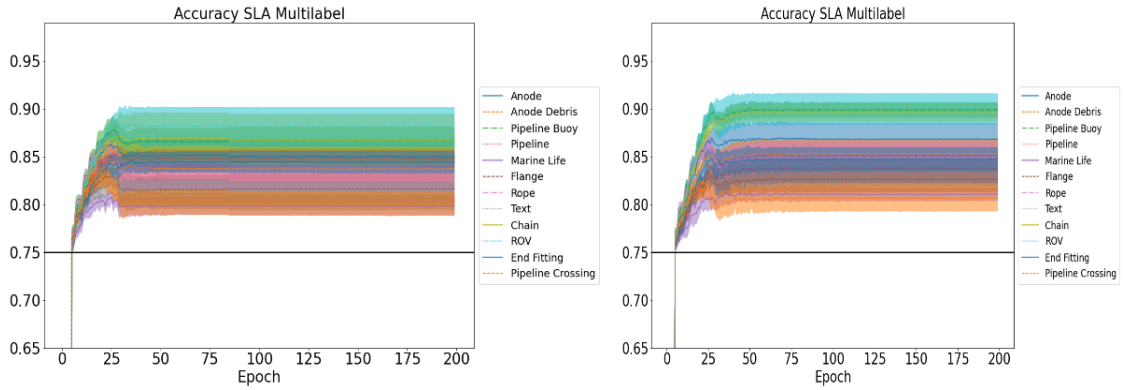


Figura 55 Acurácia SLA Multilabel. O gráfico da esquerda refere-se ao modelo SLAM-JL e o da direita ao modelo SLAM

Essa é uma característica específica dos modelos baseados em SLAM nesse experimento, onde o pico de desempenho está próximo do pico da métrica Acurácia SLA Multilabel. Uma possibilidade interessante seria fixar os rótulos atribuídos nesta época, pois poderia levar a uma melhoria no desempenho do modelo, uma vez que as redes estariam expostas a menos ruído. No entanto, adicionar um hiperparâmetro para controlar esse processo seria extremamente dependente do conjunto de dados, uma vez que tal comportamento não foi observado nos experimentos com o conjunto de dados UcMerced e TreeSAI. Portanto, ajustar este hiperparâmetro pode ser um desafio para aplicações industriais do mundo real.

Embora a incorporação de tal hiperparâmetro para controlar este processo possa não ser adequada para aplicações reais de DL, esta observação continua sendo uma característica interessante dos modelos que utilizam a técnica SLAM. Sendo este um ponto interessante para trabalhos futuros.

Os resultados obtidos apontam que o modelo SLAM-JL é adequado para aplicações do mundo real, demonstrando robustez em uma variedade de cenários distintos. Este modelo não apenas superou o estado da arte, mas também apresentou ganhos expressivos, com ganhos de até 17% em relação ao F1-Score. O modelo Learning by SLAM também demonstrou robustez consistente em todas as aplicações avaliadas apesar de ter sido superado pelo SLAM-JL.

7
Conclusão e Trabalhos Futuros

Nessa Tese foi apresentada uma nova técnica para lidar com amostras ruidosas chamada RDS. Os resultados indicam uma melhora no desempenho dos modelos que utilizam a Small Loss Approach de até 6% na acurácia do conjunto de teste. A nova técnica proposta pode ser utilizada para o desenvolvimento de novos modelos, além dos apresentados nessa proposta, sendo esta uma grande contribuição desse trabalho para a literatura.

Os modelos desenvolvidos utilizando o RDS foram o RDS-C, RDS-J e o RDS-Contrastive. O modelo RDS-C foi desenvolvido a partir do modelo Co-teaching+, e o RDS-J a partir do Jocor. Em todos os experimentos a técnica se mostrou capaz de melhorar o desempenho dos modelos. O modelo RDS-Contrastive teve ganhos de até 4% em relação aos modelos do estado da arte. Esses modelos desenvolvidos foram utilizados para a solução de uma demanda real de uma empresa de óleo e gás. Os resultados apontaram ganhos significativos, com ganhos de performance de até 6% em relação ao *F1-Score*.

Além disso, nessa tese foi expandida a técnica SLA para o cenário multilabel, sendo desenvolvido a técnica SLAM. Com o uso dessa técnica foram desenvolvidos dois novos modelos, o Learning by SLAM e o SLAM-JL. Esses modelos superaram o estado da arte em até 17% em relação ao F1-Score. Esses modelos também reduziram a presença de ruído no conjunto de dados em até 95%. O cenário de amostras ruidosas para multiclasse é pouco explorado na literatura, dessa forma, a técnica SLAM pode expandir a atenção para esse tema importante na literatura.

Ao longo do doutorado foi realizado quatro publicações, sendo dois periódicos internacionais, um congresso nacional e um congresso internacional. Ainda está prevista uma última publicação, referente ao modelo RDS-Contrastive, sendo destinada a um periódico ou congresso internacional, concluindo assim cinco publicações. A técnica RDS e os modelos RDS-C e RDS-J foram publicados no periódico (A2) Neural Computations and Aplications. A técnica SLAM foi apresentada em congresso internacional. Os princípios da técnica RDS foram apresentados em congresso nacional. O modelo Slam by Joint Loss foi publicado no período (A2) Neural Computations and Aplications.

A presença de amostras ruidosas em dataset de aplicações reais continua sendo um obstáculo significativo na expansão de aplicações práticas envolvendo modelos de DL. Esse cenário exige abordagens inovadoras para melhorar a eficácia dos modelos. Nesta tese foi possível constatar alguns caminhos promissores para trabalhos futuros.

Os resultados apresentados no tópico 6.7 apontam que uma maior acurácia nos novos rótulos atribuídos para as amostras ruidosas pela técnica RDS-Label melhoram a eficácia dos modelos. Dessa forma, estudos visando melhorar o desempenho do RDS-Label podem aprimorar a performance desses modelos.

Uma limitação da área de estudos envolvendo DL com amostras ruidosas é a falta de aplicações reais. Os trabalhos mais recentes se concentram na melhoraria da performance dos modelos em datasets benchmarks, sem avaliar os modelos em aplicações reais. A particularidade de cada domínio pode trazer inovações interessantes para esta área de estudo, assim é preciso realizar mais estudos voltados para datasets com amostras ruidosas de problemas reais.

Um campo de aplicação real que pode trazer inovação na área de amostras ruidosas é o uso da técnica SLA em imagens provenientes de satélites. É possível utilizar esse tipo de imagem para identificar vazamento de óleo em águas oceânicas, ou identificar pontos de desmatamento em florestas. Devido à complexidade desses problemas, é comum o uso da técnica de segmentação para identificar o ponto exato dos eventos. Dessa forma, a aplicação da técnica SLA nesse cenário pode trazer inovações, como por exemplo, expandir a técnica SLA para segmentação.

Atualmente, os modelos do SOTA são baseados na SLA. Uma das limitações dessa técnica é a necessidade de estimar a porcentagem do ruído presente no dataset. Esse valor é estimado por amostragem, sendo um processo dependente de especialistas. É preciso buscar técnicas para automatizar a estimativa de ruído presente no dataset. Além disso, é possível explorar técnicas envolvendo confiança e incerteza para desenvolver novos modelos com desempenhos melhores sobre amostras ruidosas.

Referências Bibliográficas

[1] J. Chai, H. Zeng, A. Li, e E. W. T. Ngai, “Deep learning in computer vision: A critical review of emerging techniques and application scenarios”, *Machine Learning with Applications*, vol. 6, p. 100134, 2021, doi: 10.24433/CO.0411648.v1.

[2] M. M. Lopez e J. Kalita, “Deep Learning applied to NLP”, mar. 2017, [Online]. Disponível em: http://arxiv.org/abs/1703.03091

[3] Y. Li, H. Zhang, X. Xue, Y. Jiang, e Q. Shen, “Deep learning for remote sensing image classification: A survey”, *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, nº 6. Wiley-Blackwell, 1º de novembro de 2018. doi: 10.1002/widm.1264.

[4] M. Lai, “Deep Learning for Medical Image Segmentation”, maio 2015, [Online]. Disponível em: http://arxiv.org/abs/1505.02000

[5] B. Han *et al.*, “Co-teaching: Robust Training of Deep Neural Networks with Extremely Noisy Labels”, abr. 2018, [Online]. Disponível em: http://arxiv.org/abs/1804.06872

[6] B. Frénay e M. Verleysen, “Classification in the presence of label noise: A survey”, *IEEE Trans Neural Netw Learn Syst*, vol. 25, nº 5, p. 845–869, 2014, doi: 10.1109/TNNLS.2013.2292894.

[7] W. Liu, Y.-G. Jiang, J. Luo, e S.-F. Chang, “Noise Resistant Graph Ranking for Improved Web Image Search”, 2011. doi: 10.1109/CVPR.2011.5995315.

[8] P. Welinder, S. Branson, S. Belongie, e P. Perona, “The Multidimensional Wisdom of Crowds”, 2010.

[9] William B. Rouse e Sandra H. Rouse, “Analysis and classification of human error”, *IEEE Trans Syst Man Cybern*, 1983, doi: 10.1109/TSMC.1983.6313142.

[10] T. Sanderson e C. Scott, “Class Proportion Estimation with Application to Multiclass Anomaly Rejection”, 2014.

[11] B. Van Rooyen e R. C. Williamson, “A Theory of Learning with Corrupted Labels”, 2018. [Online]. Disponível em: http://jmlr.org/papers/v18/16-315.html.

[12] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, e L. Fei-Fei, “MentorNet: Learning Data-Driven Curriculum for Very Deep Neural Networks on Corrupted Labels”, 2018.

[13] E. Malach e S. Shalev-Shwartz, “Decoupling ‘when to update’ from ‘how to update’”, jun. 2017, [Online]. Disponível em: http://arxiv.org/abs/1706.02613

[14] X. Yu, B. Han, J. Yao, G. Niu, I. W. Tsang, e M. Sugiyama, “How does Disagreement Help Generalization against Label Corruption?”, 2019.

[15] H. Wei, L. Feng, X. Chen, e B. An, “Combating Noisy Labels by Agreement: A Joint Training Method with Co-Regularization”, 2020.

[16] M. Grandini, E. Bagli, e G. Visani, “Metrics for Multi-Class Classification: an Overview”, ago. 2020, [Online]. Disponível em: http://arxiv.org/abs/2008.05756

[17] H. Pham, Z. Dai, Q. Xie, e Q. V Le, “Meta Pseudo Labels”, 2021.

[18] T. Chen, S. Kornblith, M. Norouzi, e G. Hinton, “A Simple Framework for Contrastive Learning of Visual Representations”, 2020. [Online]. Disponível em: https://github.com/google-research/simclr.

[19] L. Deng, “The MNIST database of handwritten digit images for machine learning research”, *IEEE Signal Process Mag*, vol. 29, nº 6, p. 141–142, 2012, doi: 10.1109/MSP.2012.2211477.

[20] A. Krizhevsky, “Learning Multiple Layers of Features from Tiny Images”, 2009.

[21] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, e Xiaogang Wang, “Learning from massive noisy labeled data for image classification”, em *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, jun. 2015, p. 2691–2699. doi: 10.1109/CVPR.2015.7298885.

[22] Y. Yao *et al.*, “Jo-SRC: A Contrastive Approach for Combating Noisy Labels”, mar. 2021, [Online]. Disponível em: http://arxiv.org/abs/2103.13029

[23] L. Huang, C. Zhang, e H. Zhang, “Self-Adaptive Training: Bridging Supervised and Self-Supervised Learning”, *IEEE Trans Pattern Anal Mach Intell*, p. 1–17, 2022, doi: 10.1109/TPAMI.2022.3217792.

[24] GilbertoT. M. Dias, “GRANULADOS BIOCLÁSTICOS-ALGAS CALCÁRIAS”, 2001. doi: 10.1590/S0102-261X2000000300008.

[25] H. Cevikalp, B. Benligiray, O. N. Gerek, e H. Saribas, “Semi-Supervised Robust Deep Neural Networks for Multi-Label Classification”, 2020. doi: 10.1016/j.patcog.2019.107164.

[26] T. Burgert, M. Ravanbakhsh, e B. Demir, “On the Effects of Different Types of Label Noise in Multi-Label Remote Sensing Image Classification”, *IEEE Transactions on Geoscience and Remote Sensing*, 2022, doi: 10.1109/TGRS.2022.3226371.

[27] Y. Yang e S. Newsam, “Bag-of-visual-words and spatial extensions for land-use classification”, em *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, New York, NY, USA: ACM, nov. 2010, p. 270–279. doi: 10.1145/1869790.1869829.

[28] S. Ahlswede *et al.*, “*TreeSatAI Benchmark Archive* : a multi-sensor, multi-label dataset for tree species classification in remote sensing”, *Earth Syst Sci Data*, vol. 15, nº 2, p. 681–695, fev. 2023, doi: 10.5194/essd-15-681-2023.

[29] B. Mahesh, “Machine Learning Algorithms-A Review Machine Learning Algorithms-A Review View project Six Stroke Engine View project Batta Mahesh Independent Researcher Machine Learning Algorithms-A Review”, *International Journal of Science and Research*, 2018, doi: 10.21275/ART20203995.

[30] Ian Goodfellow and Yoshua Bengio and Aaron Courville, *Deep Learning*, vol. 1. MIT Press, 2016.

[31] X. P. Qiu, T. X. Sun, Y. G. Xu, Y. F. Shao, N. Dai, e X. J. Huang, “Pre-trained models for natural language processing: A survey”, *Science China Technological Sciences*, vol. 63, nº 10. Springer Verlag, p. 1872–1897, 1º de outubro de 2020. doi: 10.1007/s11431-020-1647-3.

[32] S. Ekins *et al.*, “Exploiting machine learning for end-to-end drug discovery and development”, *Nature Materials*, vol. 18, nº 5. Nature Publishing Group, p. 435–441, 1º de maio de 2019. doi: 10.1038/s41563-019-0338-z.

[33] N. O’ Mahony *et al.*, “Deep Learning vs. Traditional Computer Vision”, 2015. doi: 10.1007/978-3-030-17795-9_10.

[34] Z. Ghahramani, “Probabilistic machine learning and artificial intelligence”, *Nature*, vol. 521, nº 7553. Nature Publishing Group, p. 452–459, 27 de maio de 2015. doi: 10.1038/nature14541.

[35] L. Nanni, S. Ghidoni, e S. Brahnam, “Handcrafted vs. non-handcrafted features for computer vision classification”, *Pattern Recognit*, vol. 71, p. 158–172, nov. 2017, doi: 10.1016/j.patcog.2017.05.025.

[36] X. Liu *et al.*, “Self-supervised Learning: Generative or Contrastive”, *IEEE Trans Knowl Data Eng*, p. 1–1, 2021, doi: 10.1109/TKDE.2021.3090866.

[37] X. Liu *et al.*, “Self-supervised Learning: Generative or Contrastive”, *IEEE Trans Knowl Data Eng*, p. 1–1, 2021, doi: 10.1109/TKDE.2021.3090866.

[38] D. M. Kline e V. L. Berardi, “Revisiting squared-error and cross-entropy functions for training neural network classifiers”, *Neural Comput Appl*, vol. 14, nº 4, p. 310–318, dez. 2005, doi: 10.1007/s00521-005-0467-y.

[39] I. J. Myung, “Tutorial on maximum likelihood estimation”, *J Math Psychol*, vol. 47, nº 1, p. 90–100, 2003, doi: 10.1016/S0022-2496(02)00028-7.

[40] T. Van Erven e P. Harrëmos, “Rényi divergence and kullback-leibler divergence”, *IEEE Trans Inf Theory*, vol. 60, nº 7, p. 3797–3820, 2014, doi: 10.1109/TIT.2014.2320500.

[41] C. C. J. Kuo, “Understanding convolutional neural networks with a mathematical model”, *J Vis Commun Image Represent*, vol. 41, p. 406–413, nov. 2016, doi: 10.1016/j.jvcir.2016.11.003.

[42] Sebastian. Raschka e Vahid. Mirjalili, *Python machine learning : machine learning and deep learning with Python, scikit-learn, and TensorFlow*. Packt Publishing, 2017.

[43] Ian Goodfellow, Yoshua Bengio, e Aaron Courville, “Deep Learning”, *MIT press*, 2016.

[44] Y. Pang, M. Sun, X. Jiang, e X. Li, “Convolution in convolution for network in network”, *IEEE Trans Neural Netw Learn Syst*, vol. 29, nº 5, p. 1587–1597, maio 2018, doi: 10.1109/TNNLS.2017.2676130.

[45] V. Rasche, R. Proksa, R. Sinkus, P. Börnert, e H. Eggers, “Resampling of Data Between Arbitrary Grids Using Convolution Interpolation”, 1999.

[46] W. T. Rhodes, “Acousto-Optic Signal Processing: Convolution and Correlation Invited Paper”, 1981.

[47] W. D. Jia Deng, Richard Socher, Li-Jia Li, Kai Li, e Li Fei-Fei, *ImageNet: A Large-Scale Hierarchical Image Database*. IEEE, 2009.

[48] O. Russakovsky *et al.*, “ImageNet Large Scale Visual Recognition Challenge”, *Int J Comput Vis*, vol. 115, nº 3, p. 211–252, dez. 2015, doi: 10.1007/s11263-015-0816-y.

[49] Tulasi Krishna e Hemantha Kumar Kalluri, “Deep Learning and Transfer Learning Approaches for Image Classification”, 2019. [Online]. Disponível em: https://www.researchgate.net/publication/333666150

[50] B. Van Rooyen, A. K. Menon, e R. C. Williamson, “Learning with Symmetric Label Noise: The Importance of Being Unhinged”, 2015.

[51] S. Abubakar, I. Etikan, e R. S. Alkassim, “Comparision of Snowball Sampling and Sequential Sampling Technique Related papers”, 2015, doi: 10.15406/bbij.2015.03.00055.

[52] H. Song, M. Kim, e J.-G. Lee, “SELFIE: Refurbishing Unclean Samples for Robust Deep Learning”, 2019.

[53] A. Krizhevsky, “Learning Multiple Layers of Features from Tiny Images”, 2009.

[54] A. Baldominos, Y. Saez, e P. Isasi, “A survey of handwritten character recognition with MNIST and EMNIST”, *Applied Sciences (Switzerland)*, vol. 9, nº 15. MDPI AG, 1º de agosto de 2019. doi: 10.3390/app9153169.

[55] Chapelle, Olivier, Bernhard ’Scholkopf ’, e Alexander Zien, “ Semi-supervised learning”, *IEEE Transactions on Neural Networks* , 2009.

[56] Blum Avrim e Tom Mitchell, “Combining Labeled and Unlabeled Data with Co-Training*”, 1998.

[57] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, e S. Wang, “Learning Dynamic Siamese Network for Visual Object Tracking”, 2017.

[58] L. Huang, C. Zhang, e H. Zhang, “Self-Adaptive Training: Bridging Supervised and Self-Supervised Learning”, jan. 2022, [Online]. Disponível em: http://arxiv.org/abs/2101.08732

[59] G. Patrini, A. Rozza, A. K. Menon, R. Nock, e L. Qu, “Making Deep Neural Networks Robust to Label Noise: a Loss Correction Approach”, 2017.

[60] Y. Liu, H. Cheng, e K. Zhang, “Identifiability of Label Noise Transition Matrix”, fev. 2022, [Online]. Disponível em: http://arxiv.org/abs/2202.02016

[61] D. Cheng *et al.*, “Instance-Dependent Label-Noise Learning with Manifold-Regularized Transition Matrix Estimation”, 2022.

[62] G. Patrini, A. Rozza, A. Menon, R. Nock, e L. Qu, “Making Deep Neural Networks Robust to Label Noise: a Loss Correction Approach”, set. 2016.

[63] I. S. D. P. R. Nagarajan Natarajan, “Learning with noisy labels”, *NIPS’13: Proceedings of the 26th International Conference on Neural Information Processing Systems*, vol. 1, p. 1196–1204, dez. 2013.

[64] N. Inoue, E. Simo-Serra, T. Yamasaki, e H. Ishikawa, “Multi-label Fashion Image Classification with Minimal Human Supervision”, em *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, IEEE, out. 2017, p. 2261–2267. doi: 10.1109/ICCVW.2017.265.

[65] T. Burgert, M. Ravanbakhsh, e B. Demir, “On the Effects of Different Types of Label Noise in Multi-Label Remote Sensing Image Classification”, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, p. 1–13, 2022, doi: 10.1109/TGRS.2022.3226371.

[66] X. X. H. Z. Y. Z. S. G. T. L. Shikun Li, “Estimating noise transition matrix with label correlations for noisy multi-label learning”, *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, vol. 35, p. 24184–24198, 2022.

[67] D. Berthelot *et al.*, “MixMatch: A Holistic Approach to Semi-Supervised Learning”, 2019. [Online]. Disponível em: https://github.com/google-research/mixmatch

[68] C. Shorten e T. M. Khoshgoftaar, “A survey on Image Data Augmentation for Deep Learning”, *J Big Data*, vol. 6, nº 1, dez. 2019, doi: 10.1186/s40537-019-0197-0.

[69] K. Ohri e M. Kumar, “Review on self-supervised image recognition using deep neural networks”, *Knowl Based Syst*, vol. 224, jul. 2021, doi: 10.1016/j.knosys.2021.107090.

[70] M. Assran *et al.*, “Semi-Supervised Learning of Visual Features by Non-Parametrically Predicting View Assignments with Support Samples”, 2021.

[71] E. Zheltonozhskii, C. Baskin, A. Mendelson, A. M. Bronstein, e O. Litany, “Contrast to Divide: Self-Supervised Pre-Training for Learning with Noisy Labels”, 2022.

[72] M. Caron *et al.*, “Emerging Properties in Self-Supervised Vision Transformers”, 2021. [Online]. Disponível em: https://github.com/facebookresearch/dino

[73] D. T. Pham, S. S. Dimov, e C. D. Nguyen, “Selection of K in K-means clustering”, *Proc Inst Mech Eng C J Mech Eng Sci*, vol. 219, nº 1, p. 103–119, jan. 2005, doi: 10.1243/095440605X8298.

[74] E. Jang, S. Gu, e B. Poole, “Categorical Reparameterization with Gumbel-Softmax”, nov. 2016, [Online]. Disponível em: http://arxiv.org/abs/1611.01144

[75] C. Zhang, S. Bengio, M. Hardt, B. Recht, e O. Vinyals, “Understanding deep learning (still) requires rethinking generalization”, *Commun ACM*, vol. 64, nº 3, p. 107–115, mar. 2021, doi: 10.1145/3446776.

[76] D. P. Kingma e J. Ba, “Adam: A Method for Stochastic Optimization”, dez. 2014, [Online]. Disponível em: http://arxiv.org/abs/1412.6980

[77] M. Abadi *et al.*, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems”, mar. 2016, [Online]. Disponível em: http://arxiv.org/abs/1603.04467

[78] T. Xiao, T. Xia, Y. Yang, C. Huang, e X. Wang, “Learning from Massive Noisy Labeled Data for Image Classification”, 2015.

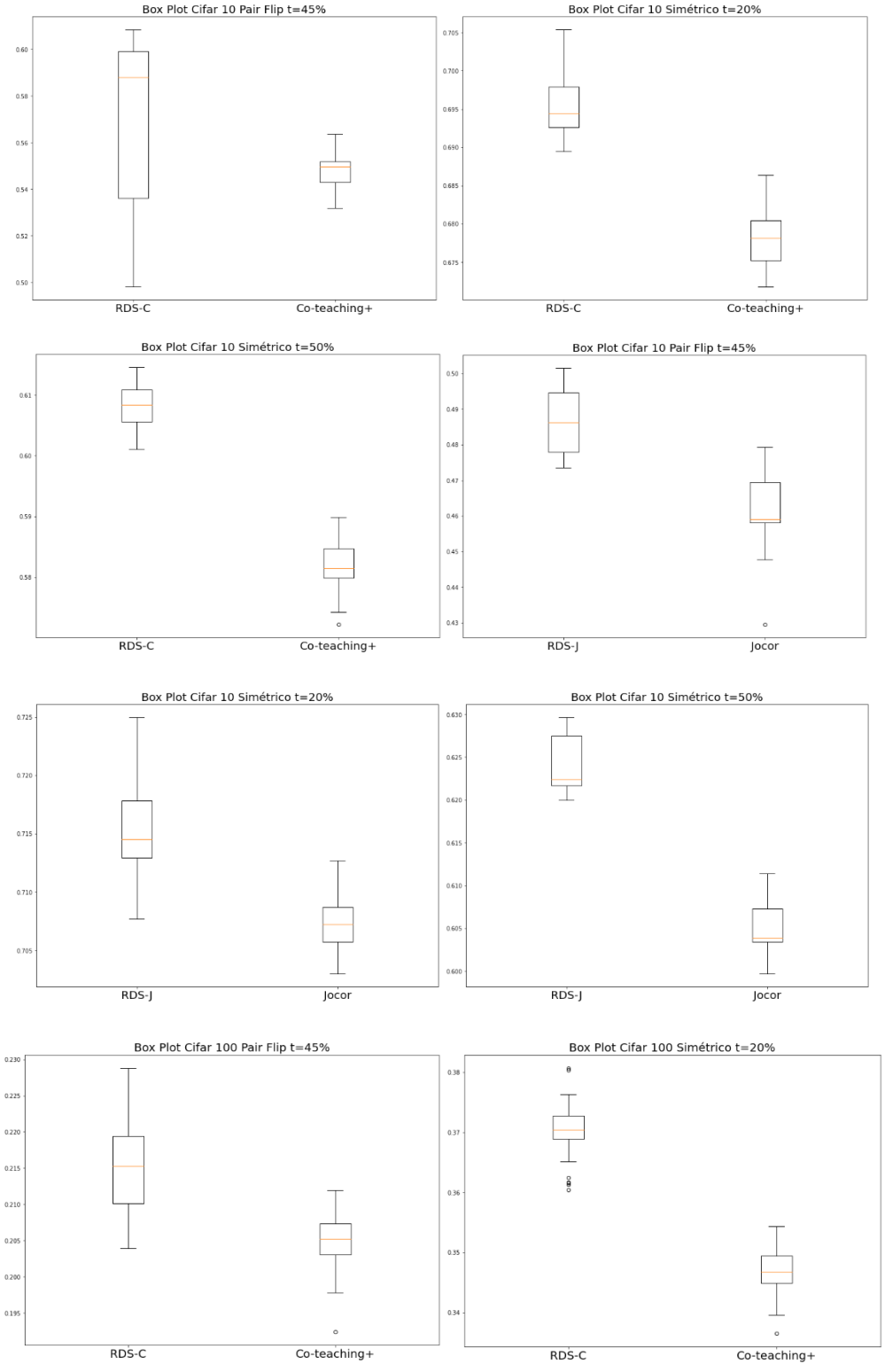
[79] T. Burgert, M. Ravanbakhsh, e B. Demir, “On the Effects of Different Types of Label Noise in Multi-Label Remote Sensing Image Classification”, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, 2022, doi: 10.1109/TGRS.2022.3226371.

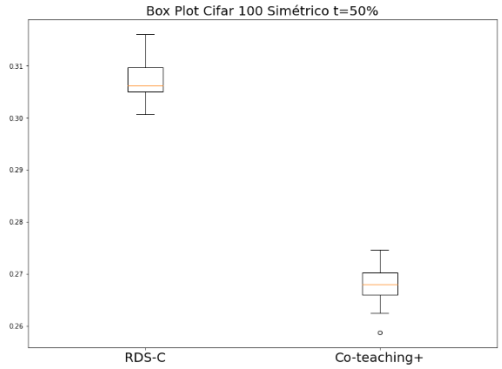
[80] M. Xu, D. Fralick, J. Z. Zheng, B. Wang, X. M. Tu, e C. Feng, “The differences and similarities between two-sample t-test and paired t-test”, *Shanghai Arch Psychiatry*, vol. 29, nº 3, p. 184–188, jun. 2017, doi: 10.11919/j.issn.1002-0829.217070.

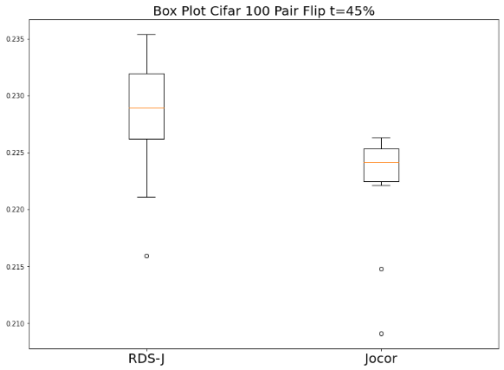
[81] K. He, X. Zhang, S. Ren, e J. Sun, “Deep Residual Learning for Image Recognition”, dez. 2015, [Online]. Disponível em: http://arxiv.org/abs/1512.03385

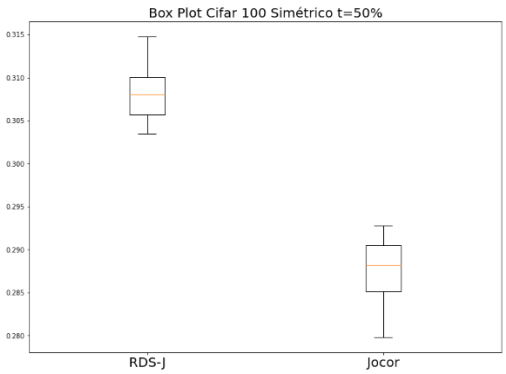
Anexo A

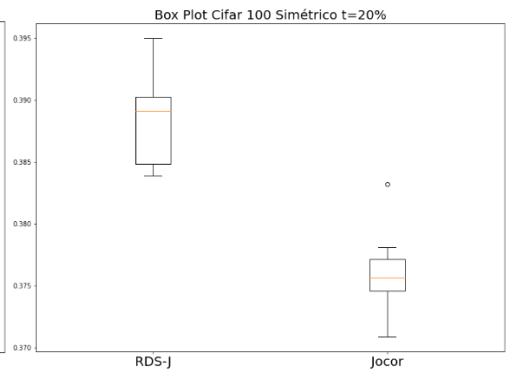
Boxplots para todos os experimentos do dataset Cifar-10, Cifar-100 e Mnist, para a época 150 do conjunto teste. Os modelos comparados foram o RDS-C com o Co-teaching+ e o modelo RDS-J com o Jocor.

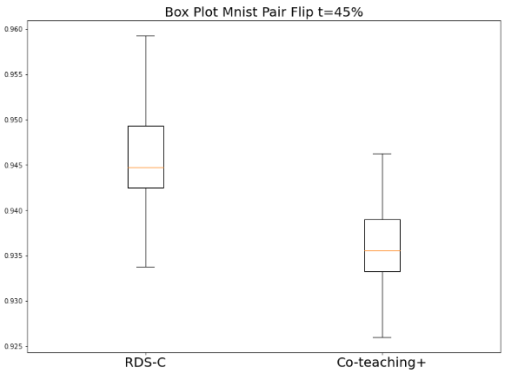


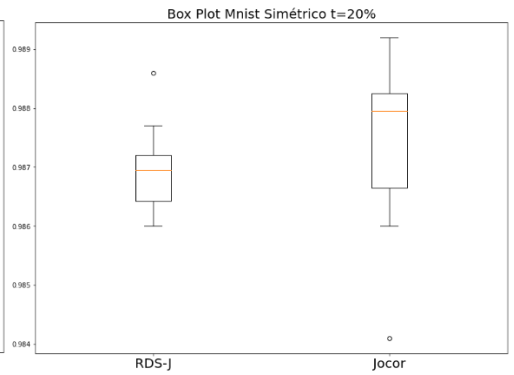

Box Plot Cifar 100 Simétrico t=50%
RDS-C
Co-teaching+

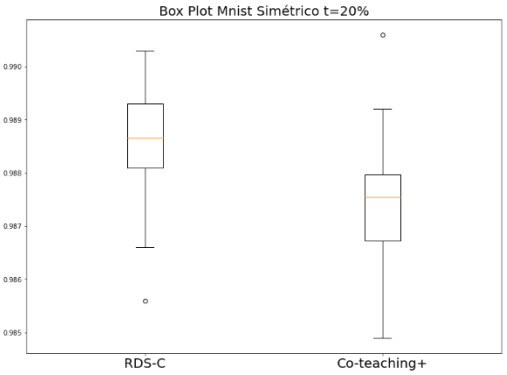

Box Plot Cifar 100 Pair Flip t=45%
RDS-J
Jocor

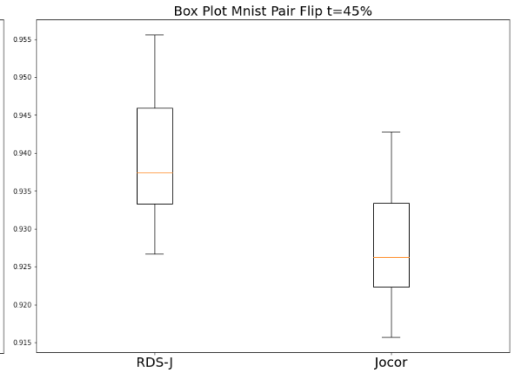

Box Plot Cifar 100 Simétrico t=50%
RDS-J
Jocor

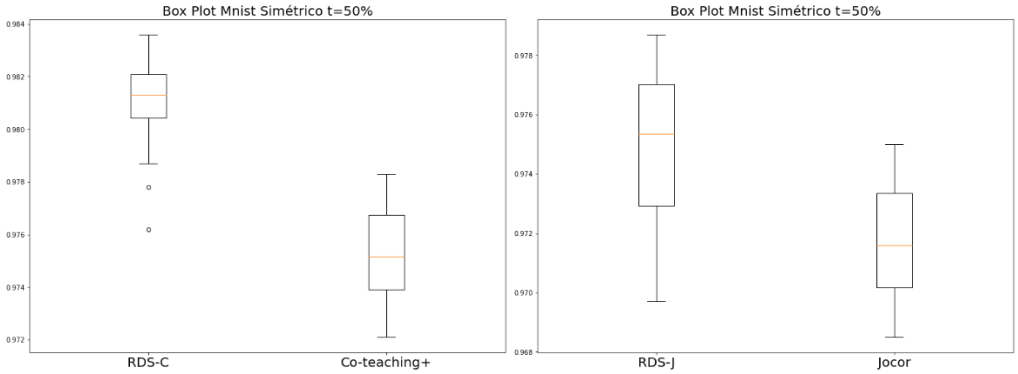

Box Plot Cifar 100 Simétrico t=20%
RDS-J
Jocor


Box Plot Mnist Pair Flip t=45%
RDS-C
Co-teaching+


Box Plot Mnist Simétrico t=20%
RDS-J
Jocor


Box Plot Mnist Simétrico t=20%
RDS-C
Co-teaching+


Box Plot Mnist Pair Flip t=45%
RDS-J
Jocor


Box Plot Mnist Simétrico t=50%
0.984
0.982
0.980
0.978
0.976
0.974
0.972
RDS-C
Co-teaching+
Box Plot Mnist Simétrico t=50%
0.978
0.976
0.974
0.972
0.970
0.968
RDS-J
Jocor

Anexo B

Gráficos para o Dataset Cifar-10: Acurácia Teste, Acurácia RDS e Relabel Total.
Ruído Pair Flip $\tau = 45$. Simétrico $\tau = 20{,}50$

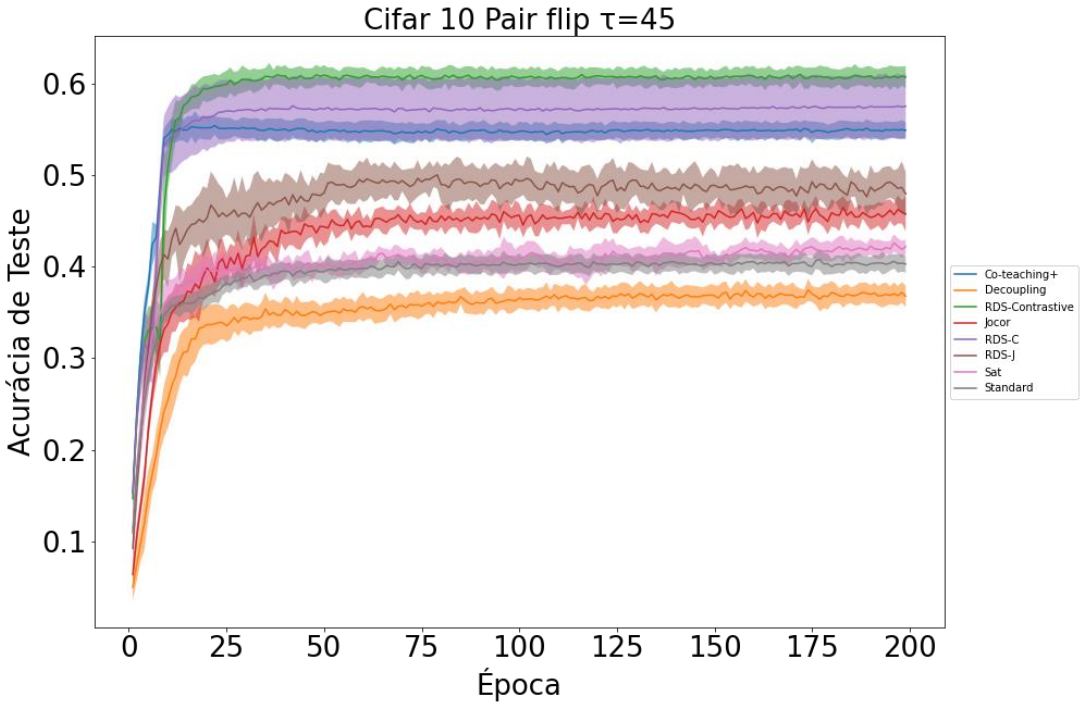


Figura 56 Acurácia de teste para o dataset Cifar 10 com ruído Pair Flip t=45

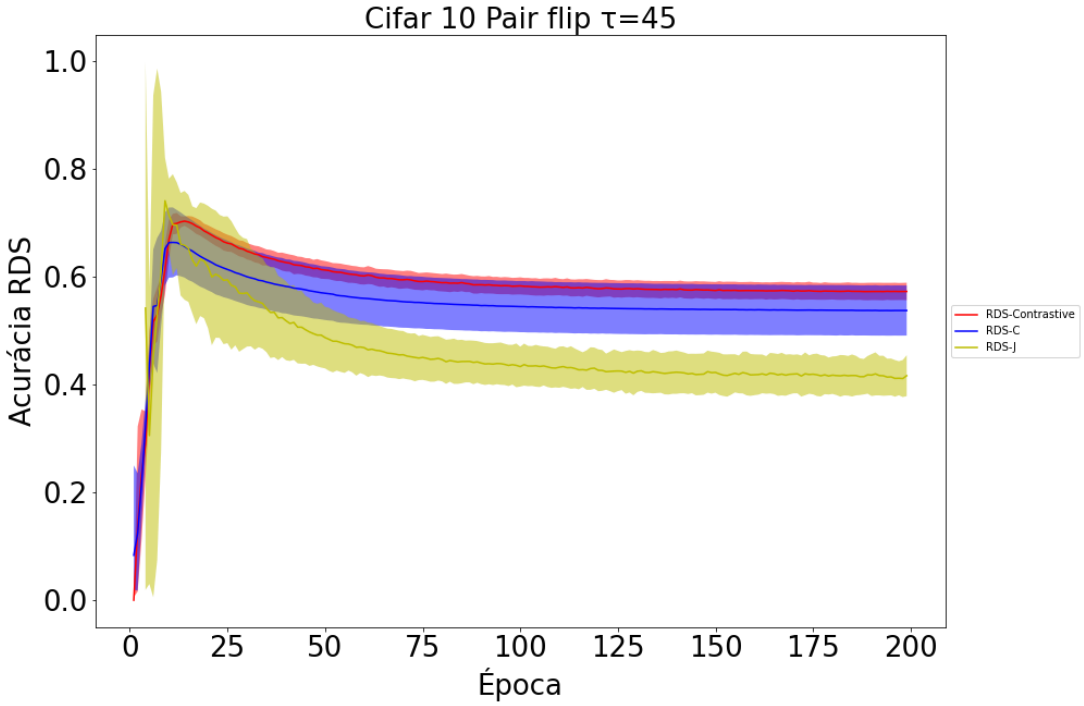


Figura 57 Acurácia RDS Precision para o dataset Cifar 10 com ruído Pair Flip t=45

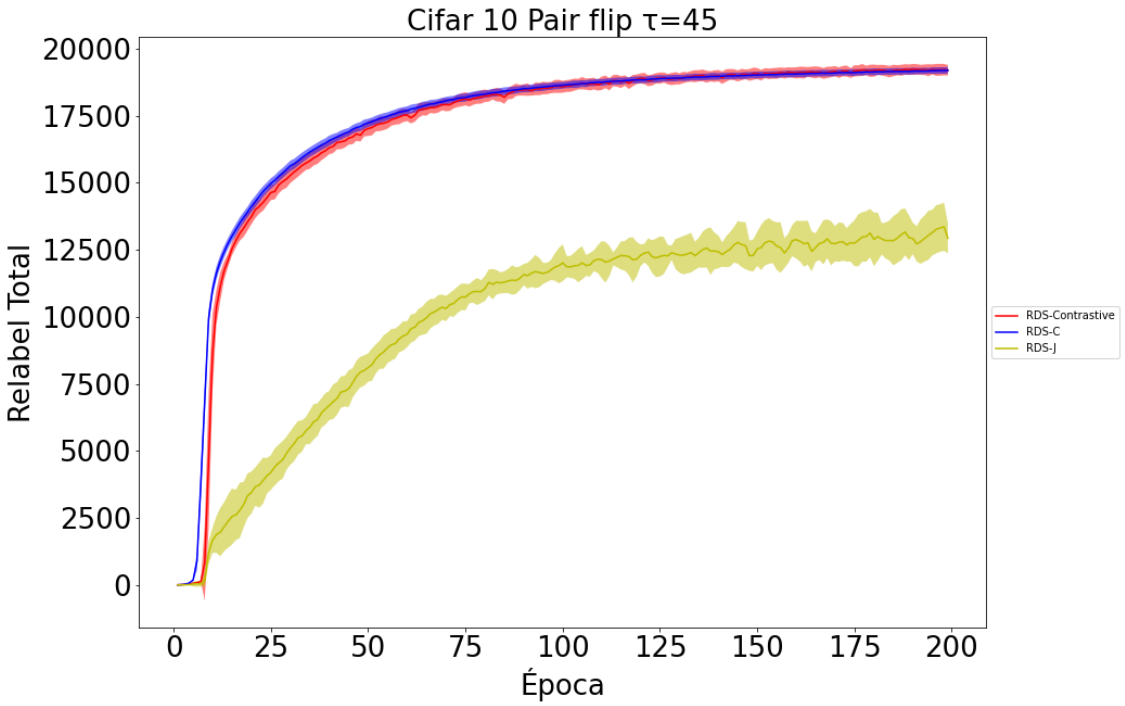


Figura 58 Relabel Total para o dataset Cifar 10 com ruído Pair Flip t=45

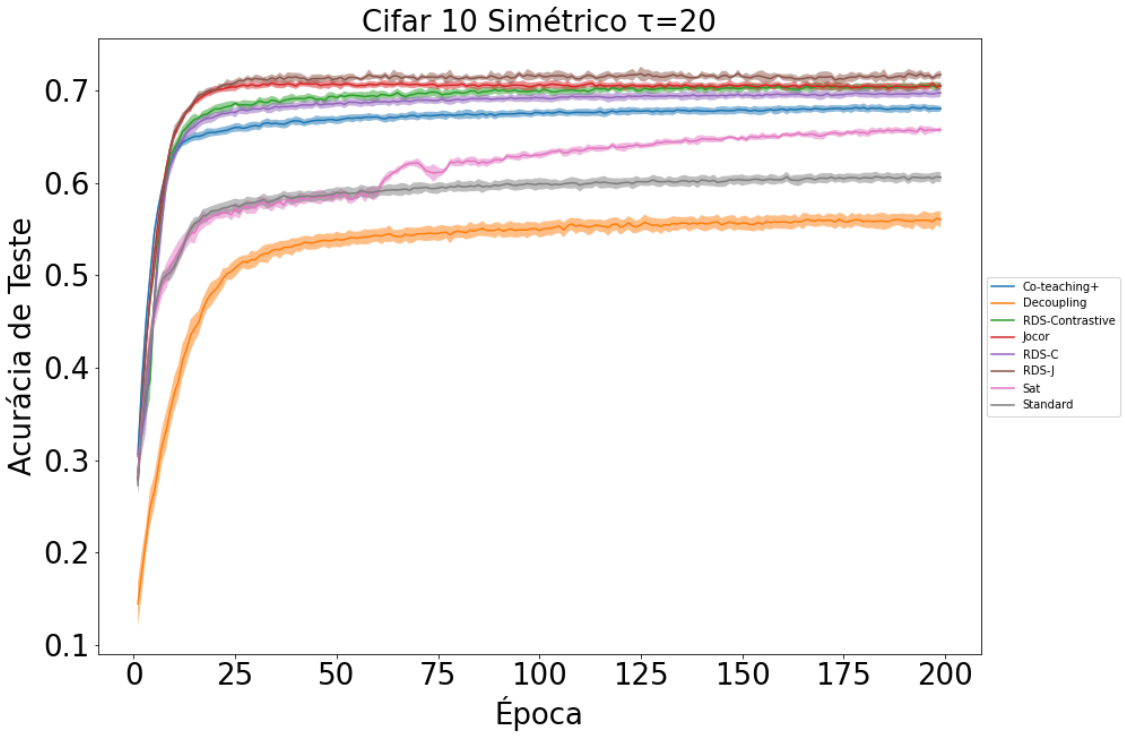


Figura 59 Acurácia de teste para o dataset Cifar 10 com ruído Simétrico t=20

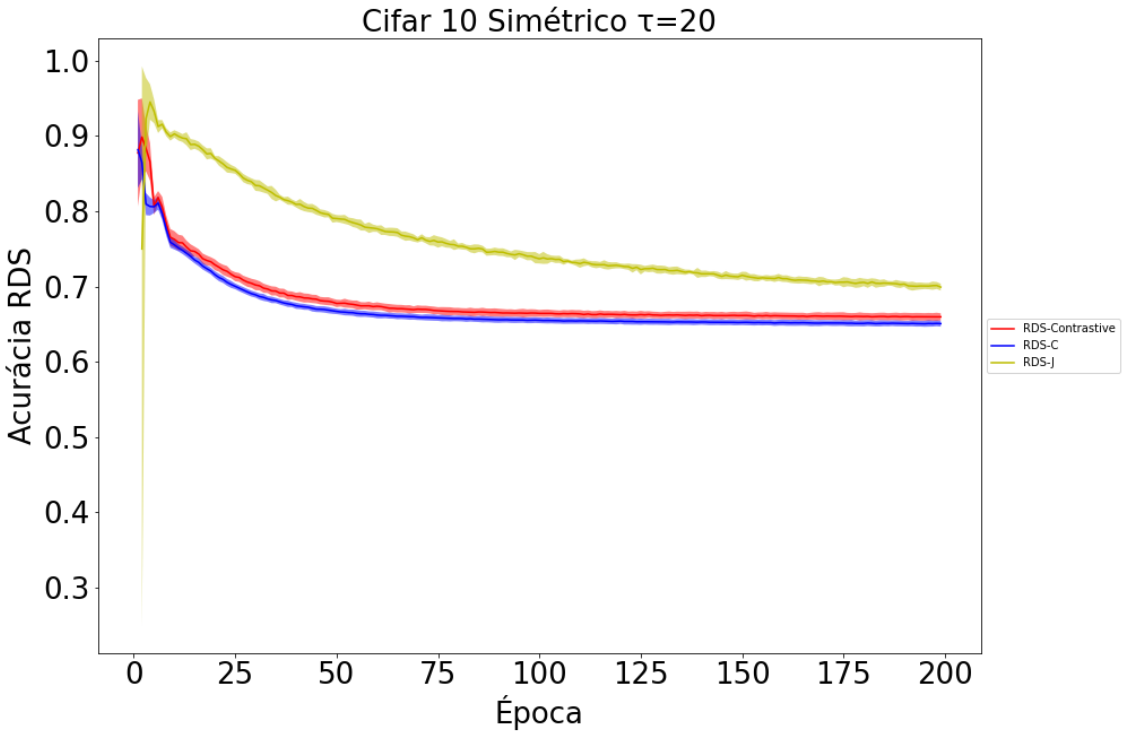


Figura 60 Acurácia RDS para o dataset Cifar 10 com ruído Simétrico t=20

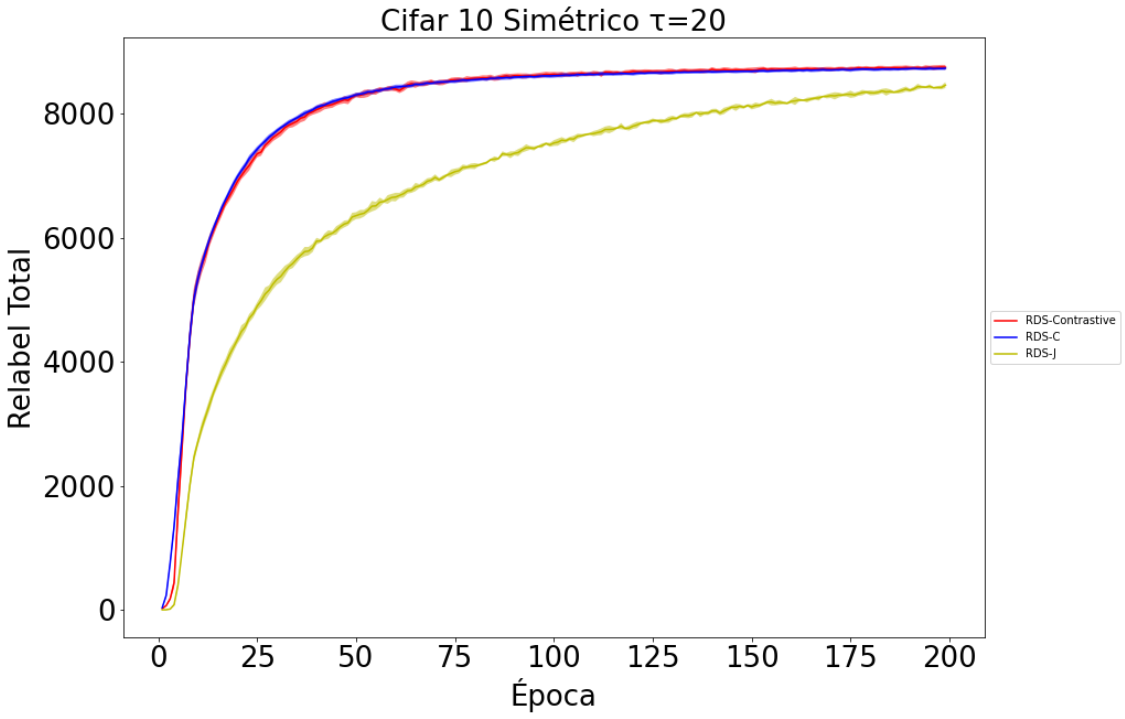


Figura 61 Relabel Total para o dataset Cifar 10 com ruído Simétrico t=20

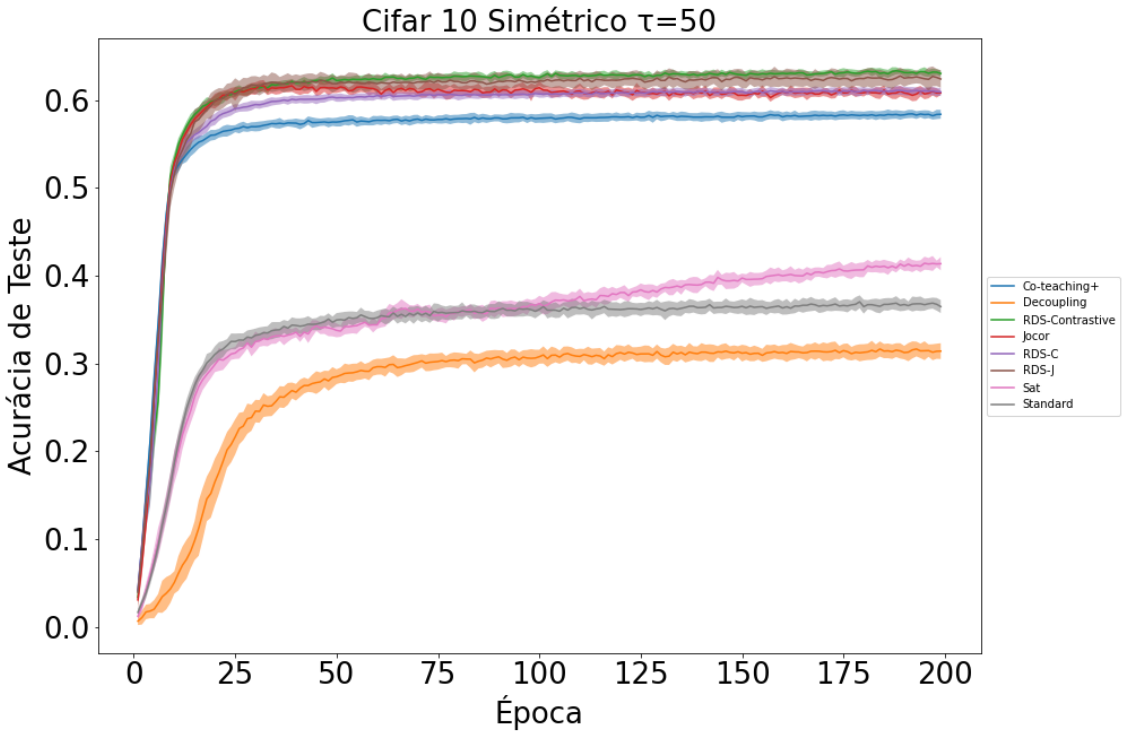


Figura 62 Acurácia de teste para o dataset Cifar 10 com ruído Simétrico t=50

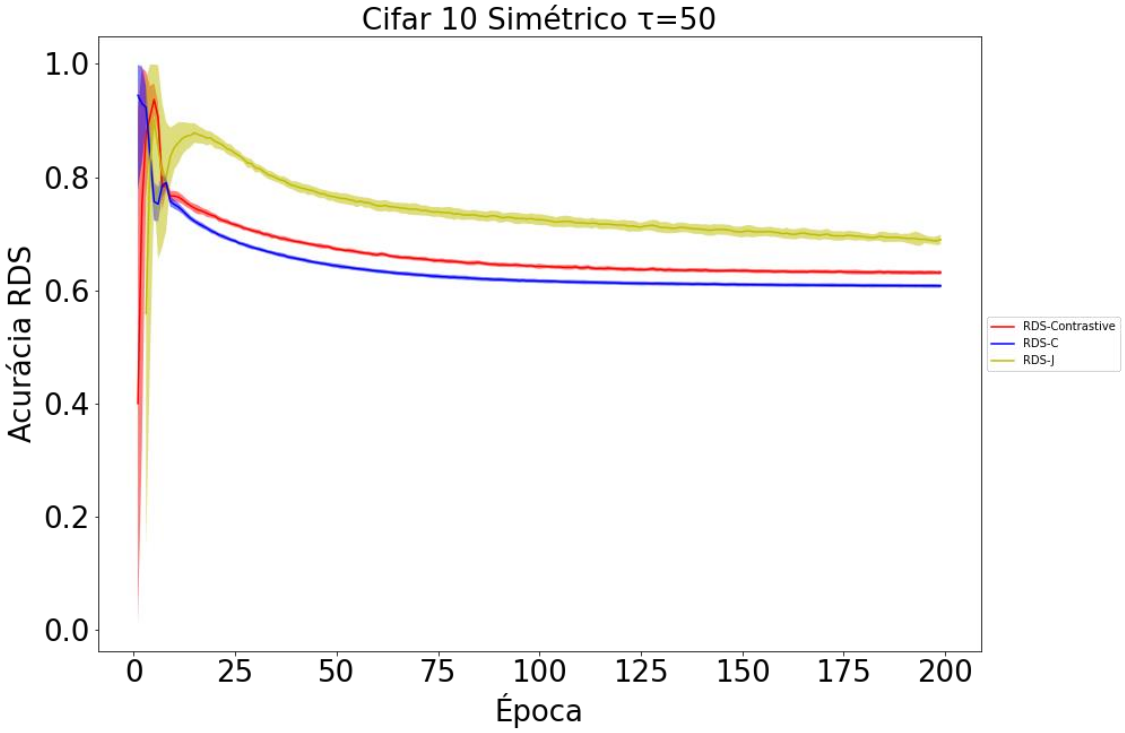


Figura 63 Acurácia RDS para o dataset Cifar 10 com ruído Simétrico t=50

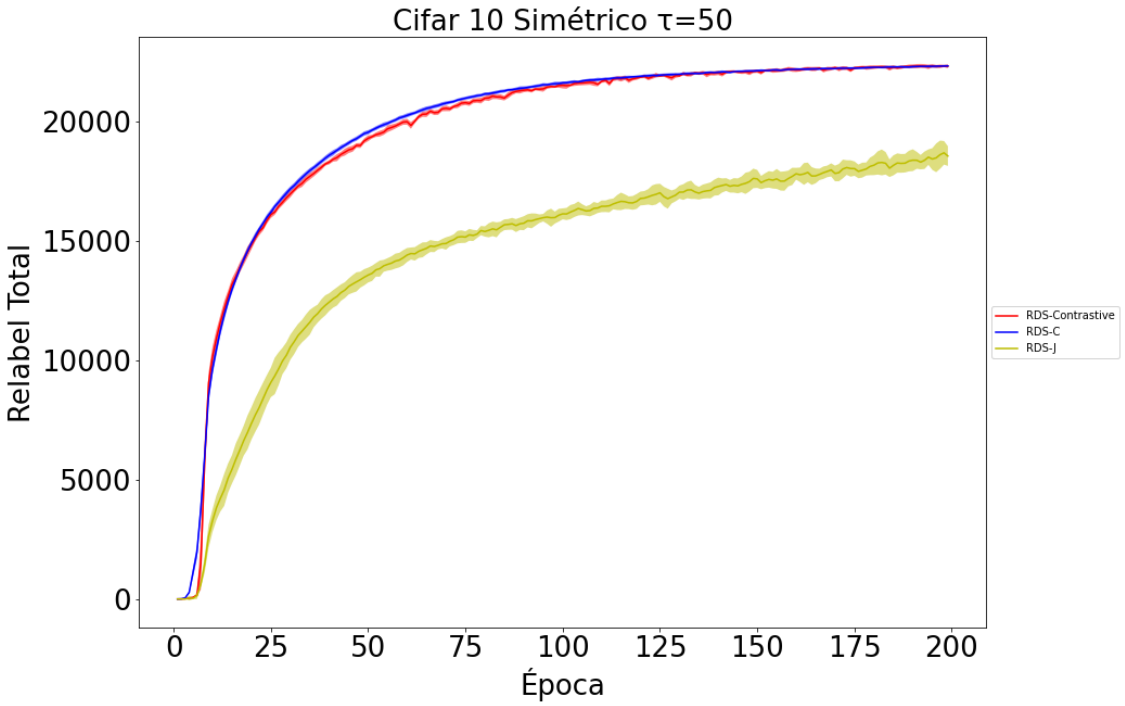


Figura 64 Relabel Total para o dataset Cifar 10 com ruído Simétrico t=50

Gráficos para o Dataset Cifar-100-: Acurácia Teste, Acurácia RDS e Relabel Total. Ruído Pair Flip $\tau = 45$. Simétrico $\tau = 20{,}50$

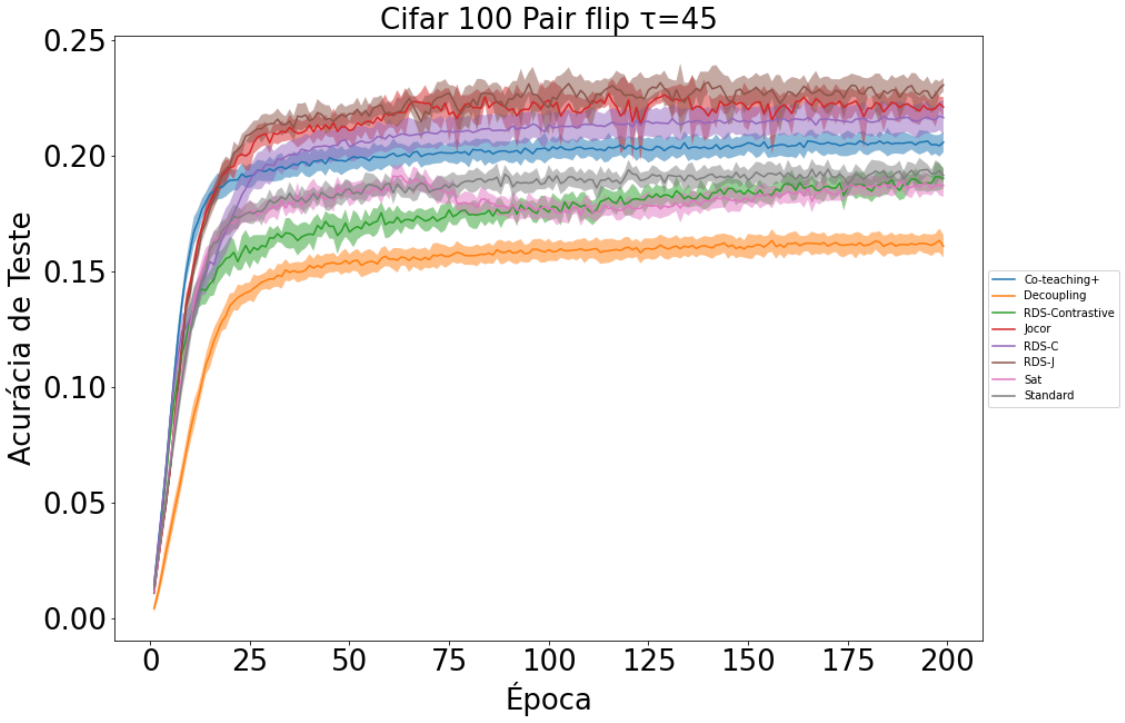


Figura 65 Acurácia de teste para o dataset Cifar 100 com ruído Pair Flip t=45

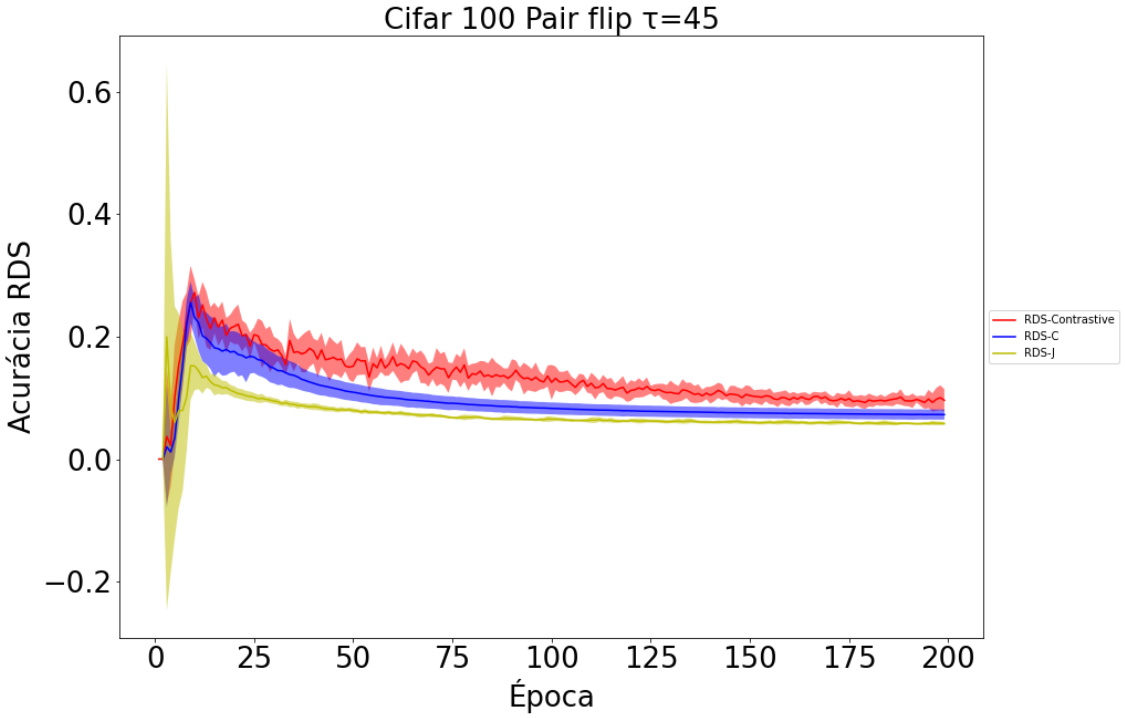


Figura 66 Acurácia RDS para o dataset Cifar 100 com ruído Pair Flip t=45

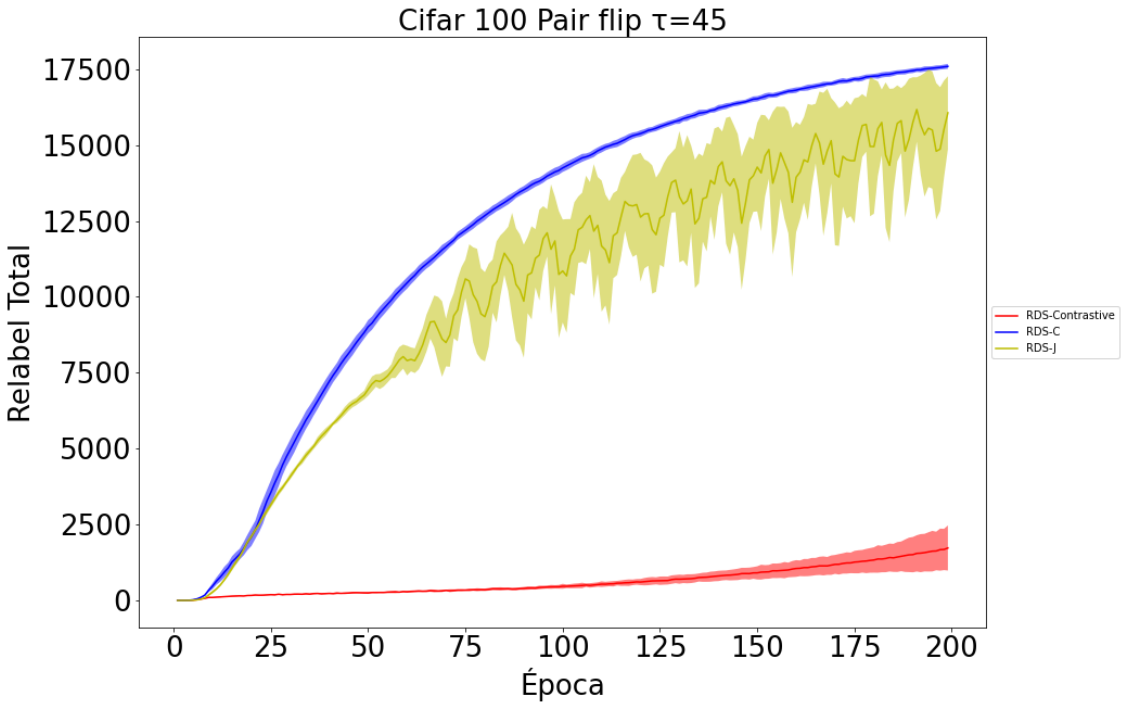


Figura 67 Relabel Total para o dataset Cifar 100 com ruído Pair Flip t=45

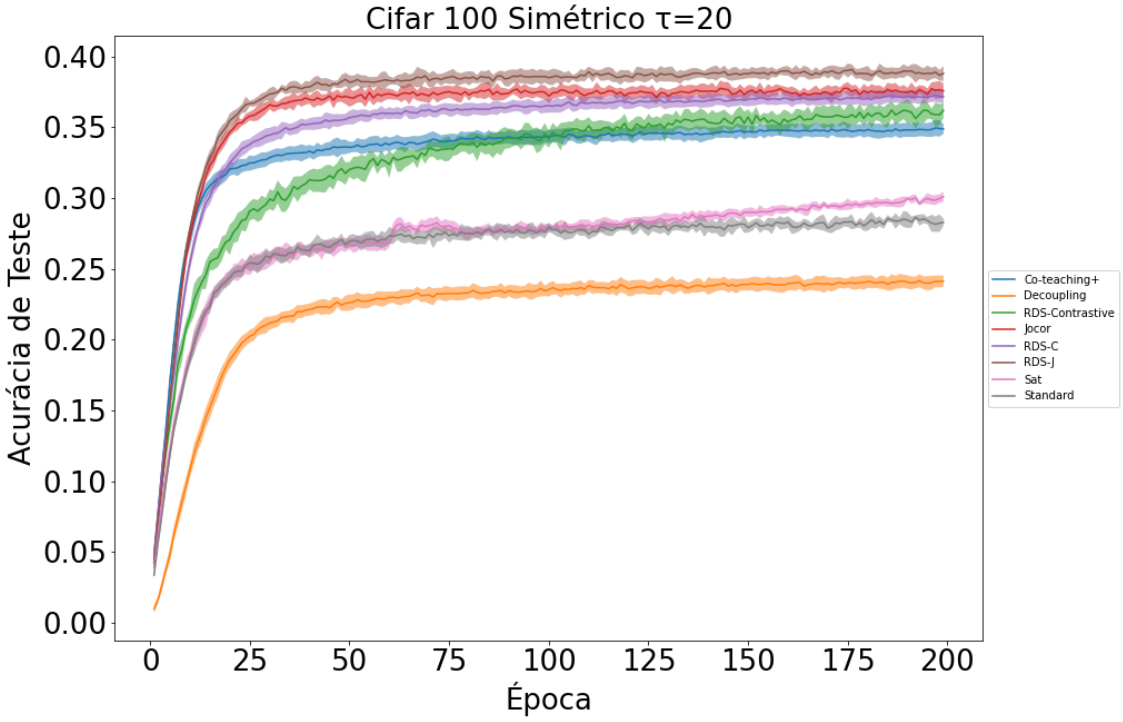


Figura 68 Acurácia de teste para o dataset Cifar 100 com ruído Simétrico t=20

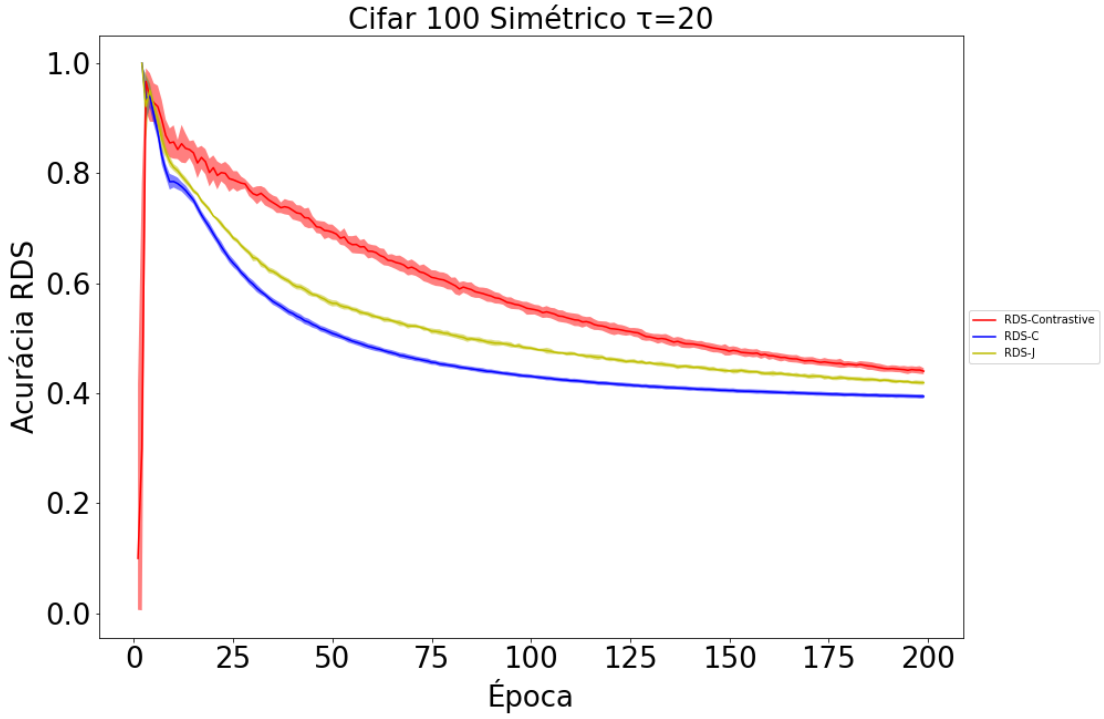


Figura 69 Acurácia RDS para o dataset Cifar 100 com ruído Simétrico t=20

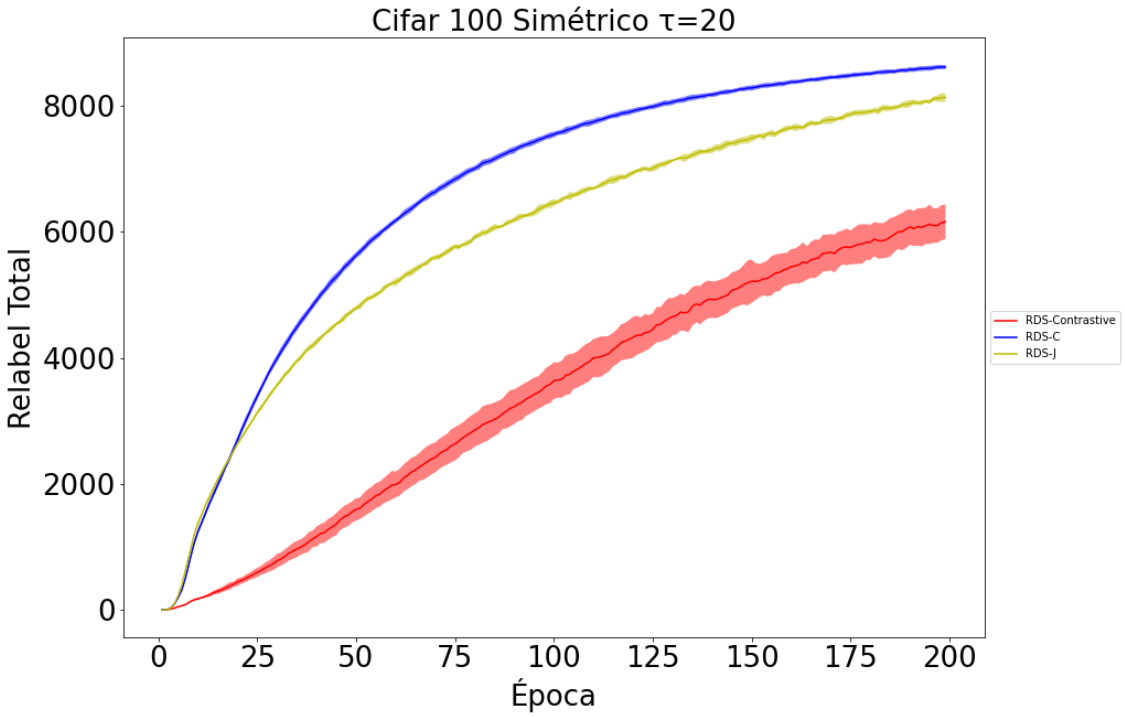


Figura 70 Relabel Total para o dataset Cifar 100 com ruído Simétrico t=20

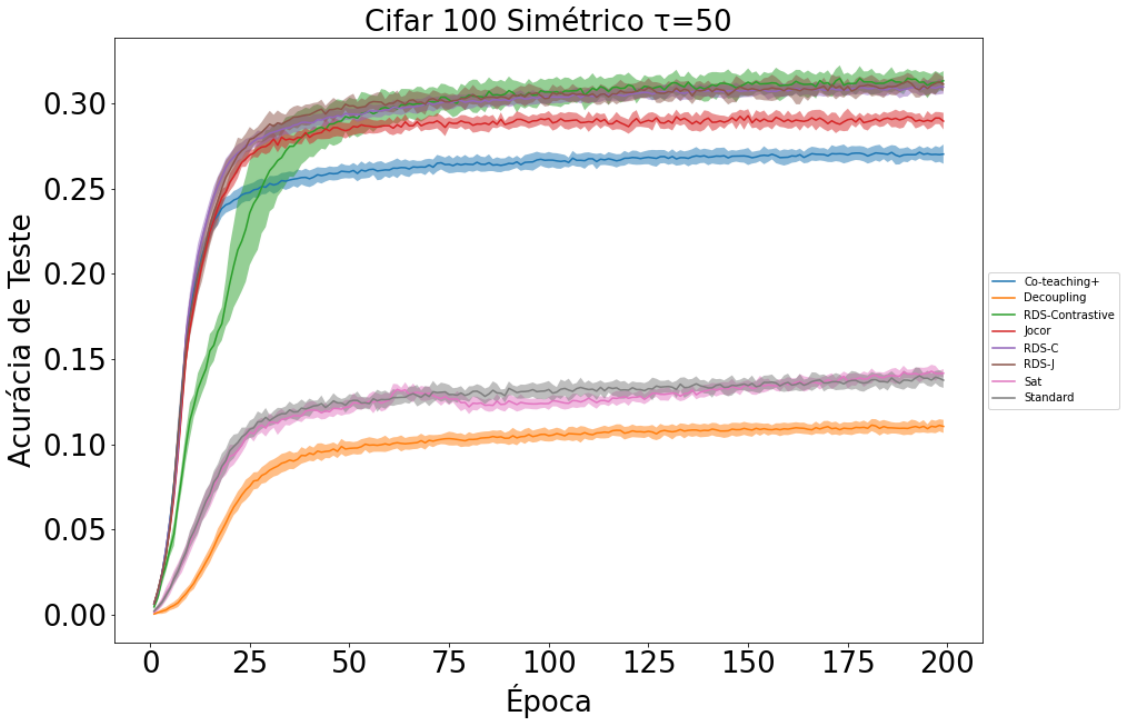


Figura 71 Acurácia de teste para o dataset Cifar 100 com ruído Simétrico t=50

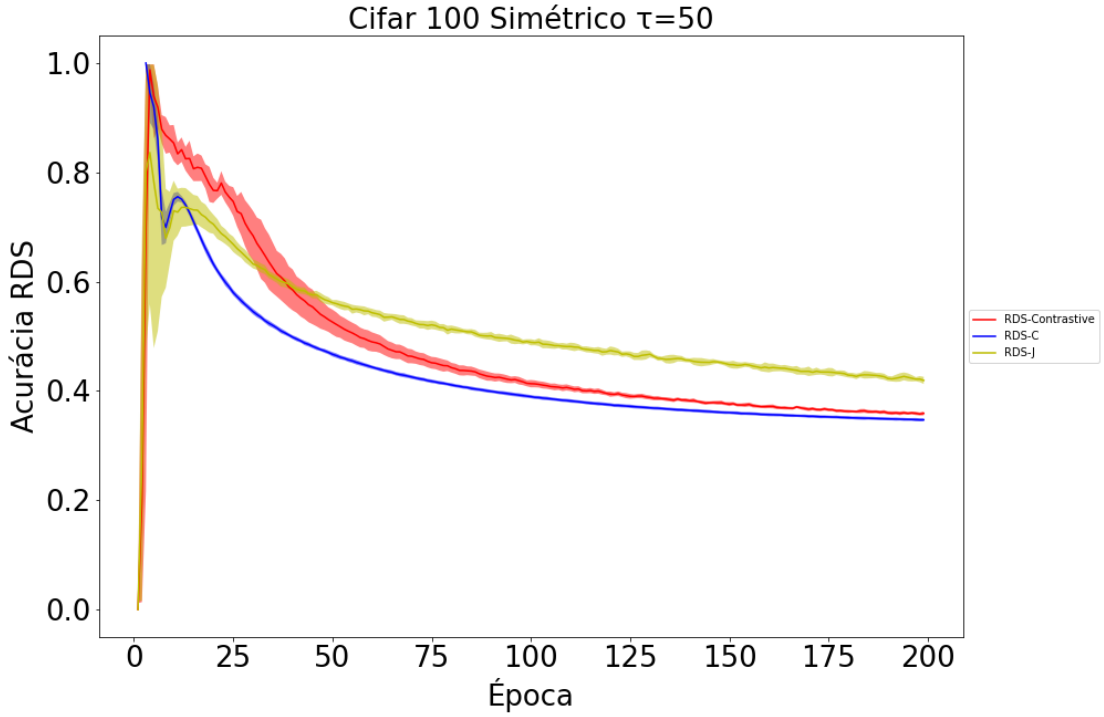


Figura 72 Acurácia RDS para o dataset Cifar 10 com ruído Simétrico t=50

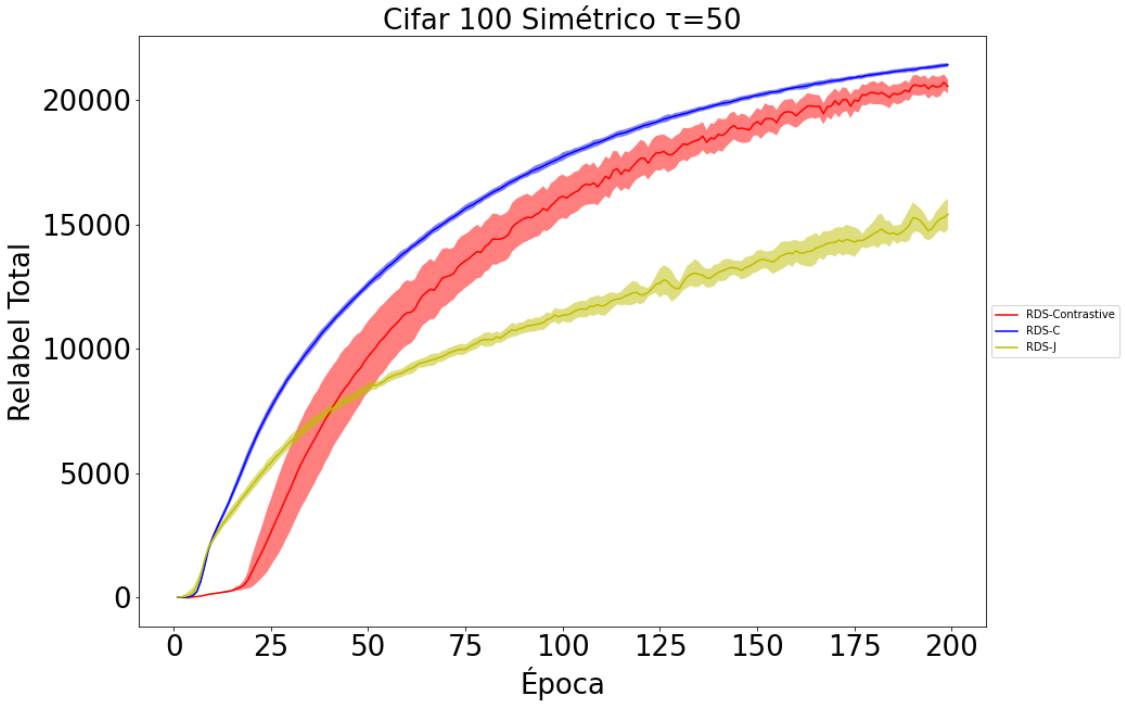


Figura 73 Relabel Total para o dataset Cifar 100 com ruído Simétrico t=50

Gráficos para o Dataset Mnist: Acurácia Teste, Acurácia RDS e Relabel Total. Ruído Pair Flip $\tau = 45$. Simétrico $\tau = 20{,}50$

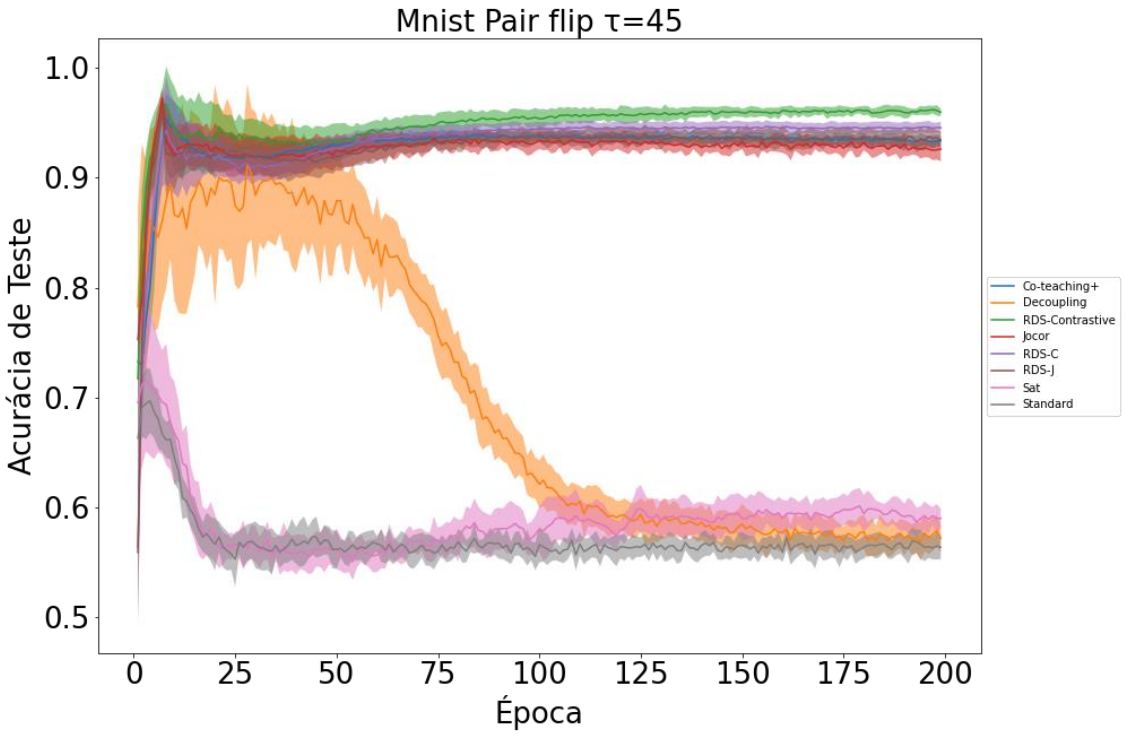


Figura 74 Acurácia de teste para o dataset Mnist com ruído Pair Flip t=45

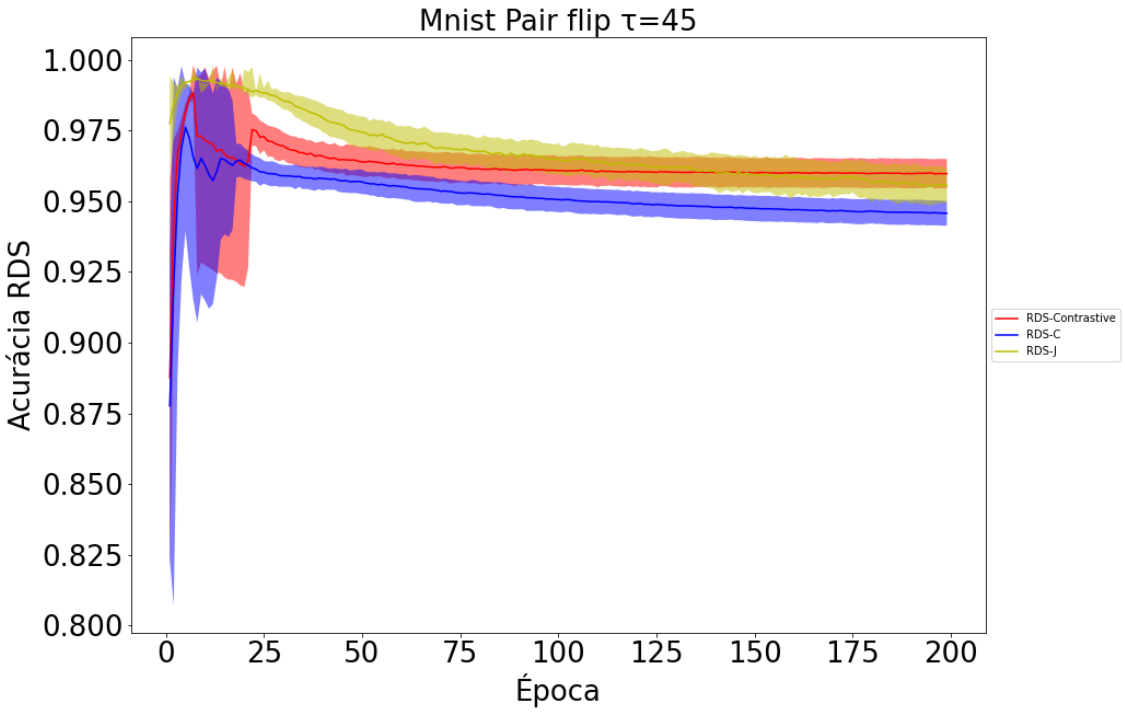


Figura 75 Acurácia RDS para o dataset Mnist com ruído Pair Flip t= 45

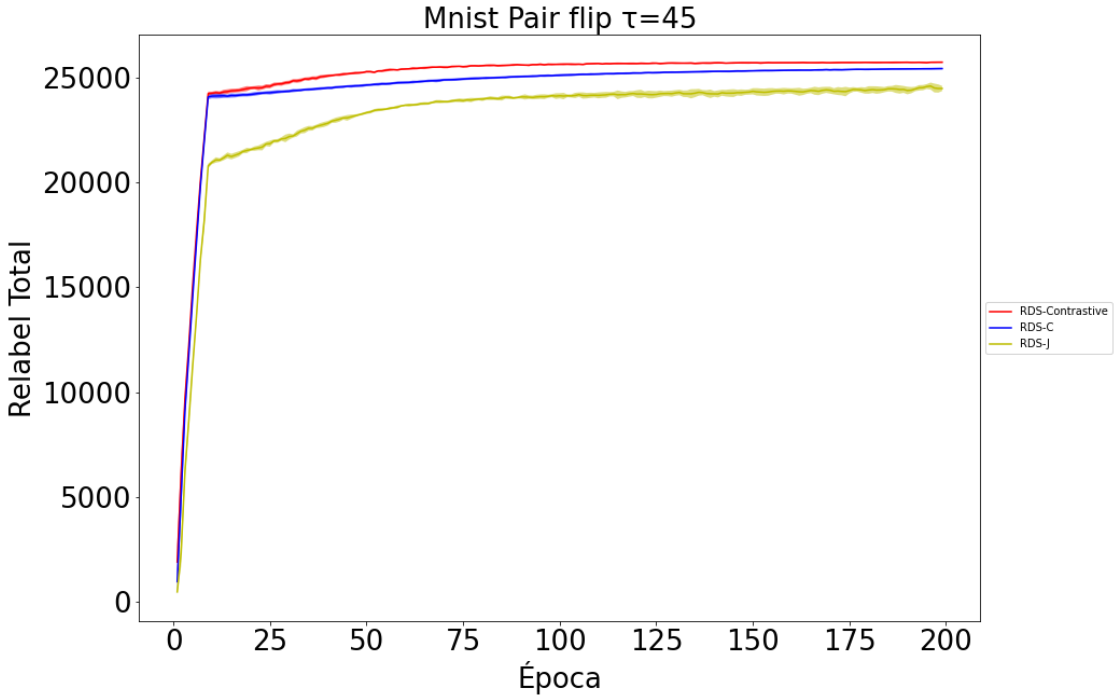


Figura 76 Relabel Total para o dataset Mnist com ruído Pair Flip t=45

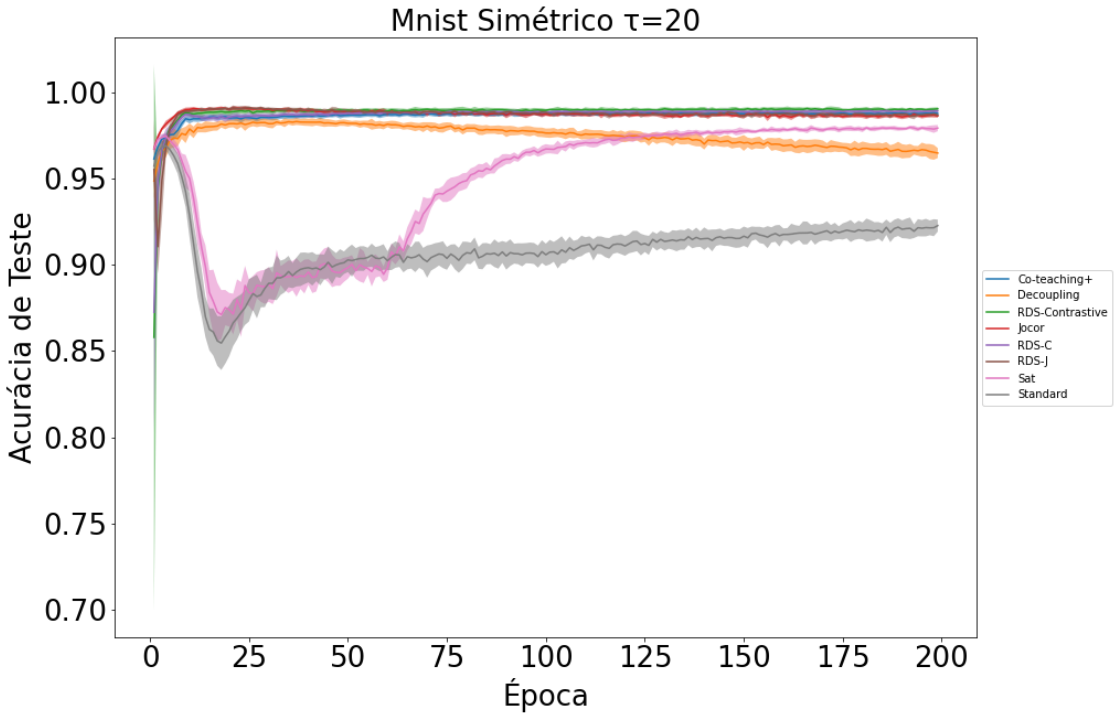


Figura 77 Acurácia de teste para o dataset Mnist com ruído Simétrico t=20

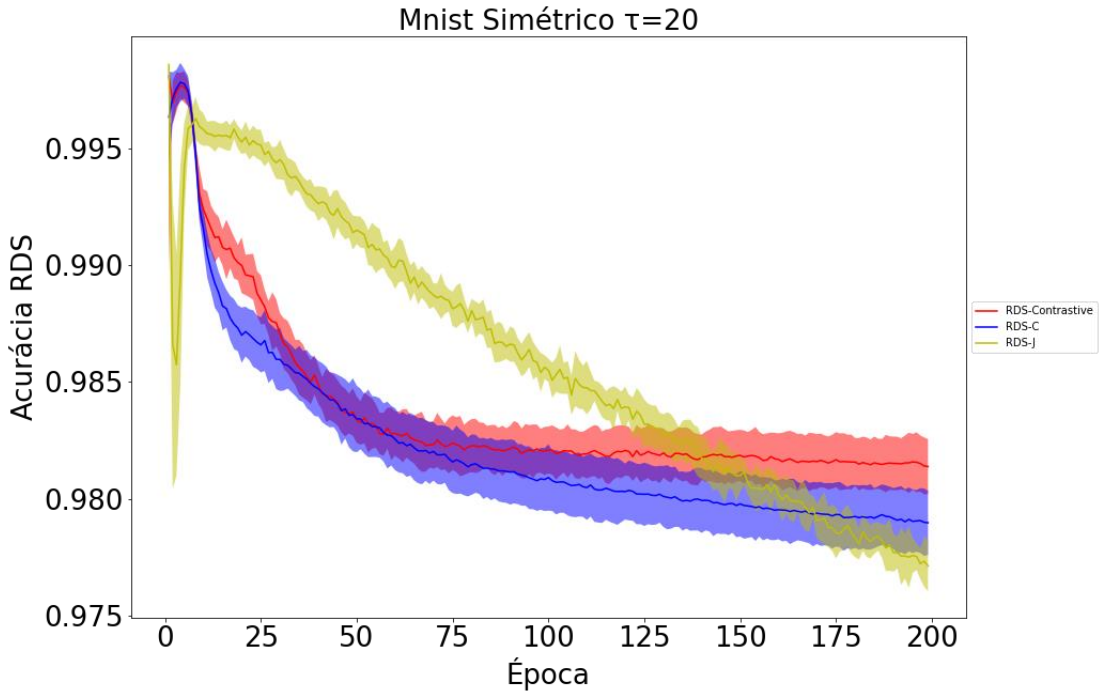


Figura 78 Acurácia RDS para o dataset Mnist com ruído Simétrico t=20

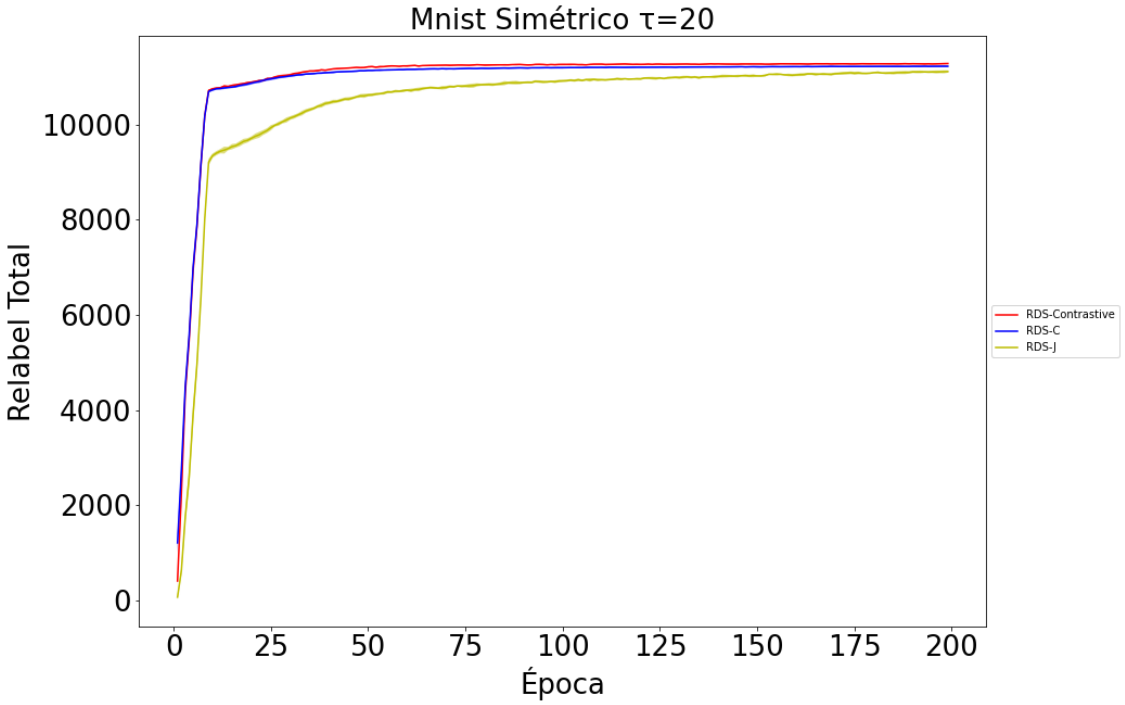


Figura 79 Relabel Total para o dataset Mnist com ruído Simétrico t=20

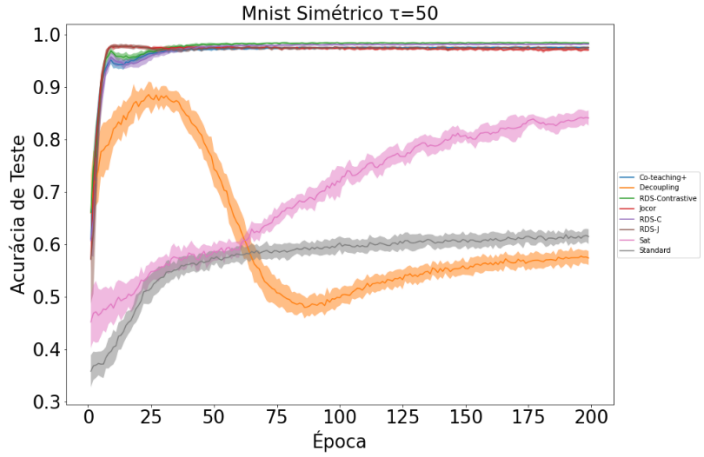


Figura 80 Acurácia de teste para o dataset Mnist com ruído Simétrico t=50

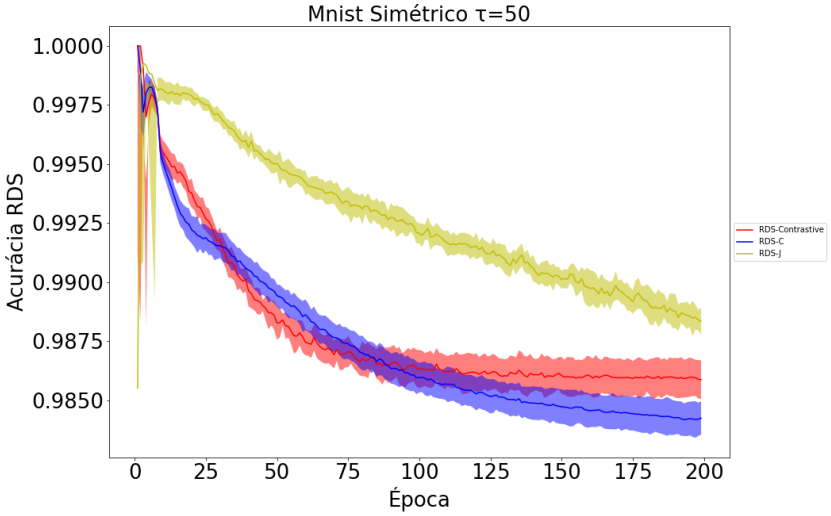


Figura 81 Acurácia RDS para o dataset Mnist com ruído Simétrico t=50

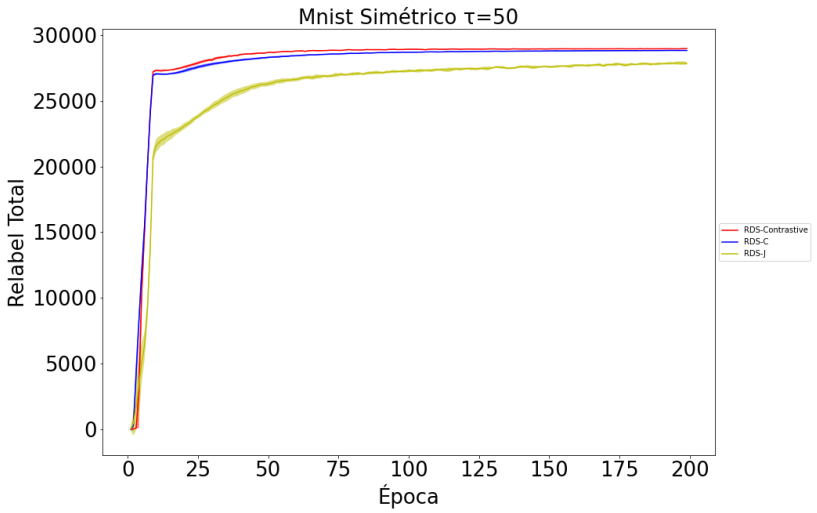


Figura 82 Relabel Total para o dataset Mnist com ruído Simétrico t=50