TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN - ĐHQG HCM
KHOA CÔNG NGHỆ THÔNG TIN

HỌC CÓ GIÁM SÁT VỚI DỮ LIỆU NHIỄU BẰNG PHƯƠNG PHÁP LỰA CHỌN DỮ LIỆU

GVHD
ThS. Trần Trung Kiên
TS. Nguyễn Ngọc Thảo

SVTH
Lê Xuân Hoàng 20120089
Lê Xuân Huy 20120494

TP Hồ Chí Minh, tháng 07, năm 2024

Nội dung

1. Giới thiệu đề tài
2. Mô hình mạng nơ-ron với phương pháp CRUST
3. Thí nghiệm
4. Tổng kết và hướng phát triển

Nội dung

1. Giới thiệu đề tài

2. Mô hình mạng nơ-ron với phương pháp CRUST

3. Thí nghiệm

4. Tổng kết và hướng phát triển

Giới thiệu đề tài

- Chất lượng của dữ liệu là yếu tố quan trọng quyết định chất lượng của mô hình học máy.
- Trong thực tế, dữ liệu nhiễu tồn tại rất phổ biến.
- Ví dụ: Tỉ lệ dữ liệu nhiễu trong tập ImageNet có thể lên đến 20%.

Một số mẫu dữ liệu nhiễu trên tập ImageNet (labelerrors.com)

Giới thiệu đề tài

Dữ liệu nhiễu:

- Là dữ liệu không chính xác, không đầy đủ hoặc không liên quan đến nhiệm vụ học máy. Dữ liệu này có thể ảnh hưởng tiêu cực đến hiệu suất của mô hình học máy.
- Có nhiều loại dữ liệu nhiễu, ví dụ như: dữ liệu nhiễu có nhãn nhiễu, dữ liệu không đầy đủ, dữ liệu không nhất quán, dữ liệu ngoại lệ, dữ liệu nhiễu ngẫu nhiên. Trong khóa luận, chỉ tập trung vào dữ liệu có nhãn nhiễu.

Giới thiệu đề tài

Định nghĩa bài toán:

Cho tập dữ liệu có chứa nhãn nhiễu $D = \{(x_i, y_i)\}_{i=1}^{n} \subset \mathbb{R}^d \times \mathbb{R}$ với (x_i, y_i) lần lượt là dữ liệu đầu vào và nhãn của mẫu thứ i. Chúng ta chỉ có thể quan sát được dữ liệu đầu vào và nhãn của nó $\{y_i\}_{y=1}^{n}$ nhưng không quan sát được nhãn thật của nó $\{\tilde{y}_i\}_{y=1}^{n}$.

Tìm hàm f: $\mathbb{R}^d \to \mathbb{R}$ có thể dự đoán nhãn thật $\tilde{y} \in \mathbb{R}$ của $x \in \mathbb{R}^d$ mới ở ngoài D.

Giới thiệu đề tài

Thách thức của bài toán:

- Khó khăn trong việc xác định dữ liệu sạch, dữ liệu nhiễu.
- Mô hình dễ bị học theo các nhãn nhiễu.

Giới thiệu đề tài

Ý nghĩa của bài toán:

- Nâng cao hiệu quả và độ chính xác của mô hình khi huấn luyện với dữ liệu nhiễu.
- Tiết kiệm chi phí và thời gian trong việc thu thập dữ liệu.
- Thúc đẩy sự phát triển, mở rộng khả năng ứng dụng của học máy trong các lĩnh vực mới.

Giới thiệu đề tài

Các phương pháp giải quyết:

- Regularization.
- Co-teaching.
- INCV.
- CRUST.

Phương pháp Regularization

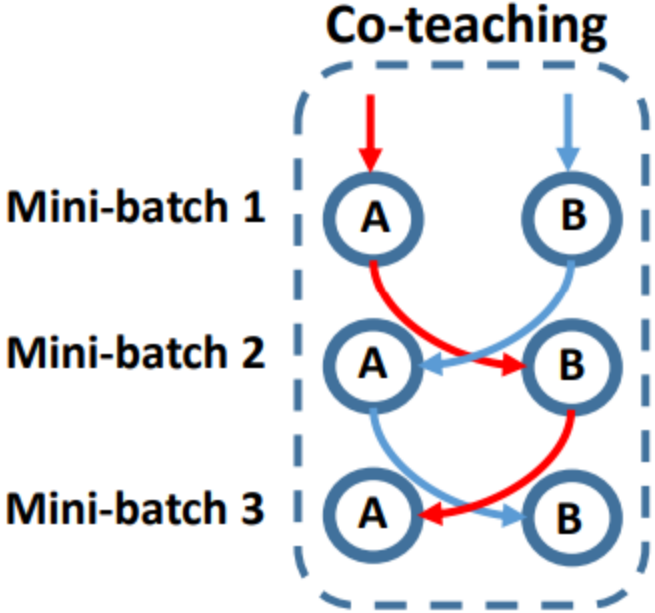
Thêm các ràng buộc, các điều kiện vào quá trình huấn luyện để ngăn chặn mô hình học theo các nhãn nhiễu.

Tiêu biểu: Weight decay (Krogh, A. and Hertz, J.A., 1992).

Điểm yếu: khó đạt được hiệu suất tốt khi dữ liệu có tỉ lệ nhãn nhiễu cao.

Phương pháp Co-teaching (Han et al., 2018)

- Huấn luyện đồng thời hai mạng nơ-ron học sâu và cho phép chúng "dạy" lẫn nhau trên từng mini-batch dữ liệu.
- Với mỗi batch, mỗi mạng sẽ xem xét những trường hợp có độ mất mát nhỏ làm kiến thức hữu ích (hay là dữ liệu sạch) để dạy cho mạng kia.
- Việc trao đổi dữ liệu sạch như vậy giúp giảm thiểu ảnh hưởng của dữ liệu nhiễu hơn.



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

Phương pháp INCV (Chen et al., 2019)

- Iterative Noisy Cross-Validation (INCV) là một phương pháp huấn luyện với dữ liệu nhiễu phát triển từ Co-teaching.
- Lặp đi lặp lại việc cross validation để loại bỏ dữ liệu nhiễu => Sử dụng dữ liệu sạch còn lại bằng phương pháp Co-teaching.

Cross-Validation + Co-teaching

Phương pháp CRUST (Mirzasoleiman et al., 2020)

- Được đề xuất trong bài báo “Coresets for Robust Training of Neural Networks against Noisy Labels” được trình bày trong hội nghị NeurIPS 2020.
- Chọn tập con từ dữ liệu gốc sao cho chúng giữ lại các thông tin quan trọng mà không bị ảnh hưởng quá nhiều bởi nhiễu. Dựa trên một số quan sát về sự phân bố của dữ liệu trên không gian gradient.
- Đây là phương pháp cho kết quả tốt nhất và cũng là phương pháp mà nhóm chọn để nghiên cứu và tìm hiểu sâu.

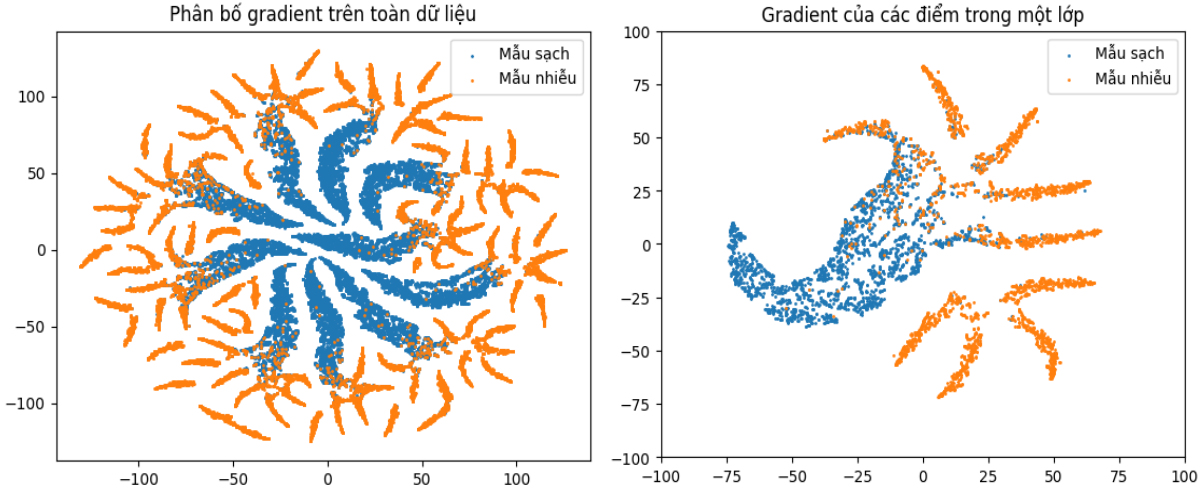
Nội dung

1. Giới thiệu đề tài
2. **Mô hình mạng nơ-ron với phương pháp CRUST**
3. Thí nghiệm
4. Tổng kết và hướng phát triển

Phương pháp CRUST (Mirzasoleiman et al., 2020)

Ý tưởng:

Dựa trên quan sát về sự phân bố của gradient đối với các mẫu dữ liệu sạch và mẫu dữ liệu nhiễu.



Phân bố của gradient trên tập dữ liệu CIFAR-10 (nhiễu 50%)

Phương pháp CRUST (Mirzasoleiman et al., 2020)

Đầu vào: Tập dữ liệu D có chứa nhãn nhiễu.

Đầu ra: Tham số W của mạng.

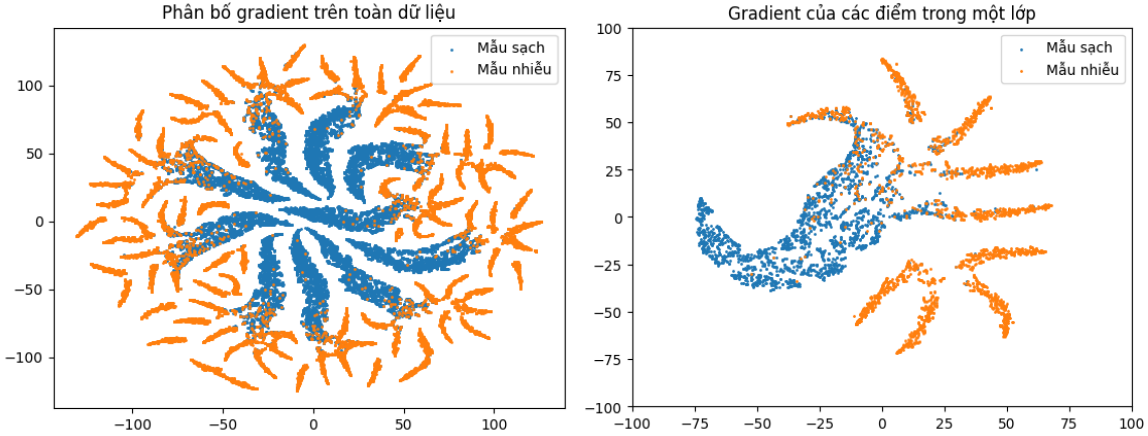
Với mỗi epoch:

- Đối với mỗi lớp c trong các lớp dữ liệu:
  - Chọn ra tập U thuộc D gồm các mẫu có nhãn c.
  - Tính khoảng cách gradient giữa các điểm thuộc U.
  - Sử dụng thuật toán tìm cụm lớn nhất để tìm ra coreset s theo lớp c.
- Gộp các coreset s theo từng lớp thành coreset S để huấn luyện.
- Huấn luyện mô hình trên coreset S và cập nhật W.

Phương pháp CRUST (Mirzasoleiman et al., 2020)

Tại sao tìm kiếm coreset theo từng lớp?

Dữ liệu thường gom cụm theo lớp trước rồi mới gom cụm theo nhãn sạch và nhãn nhiễu.

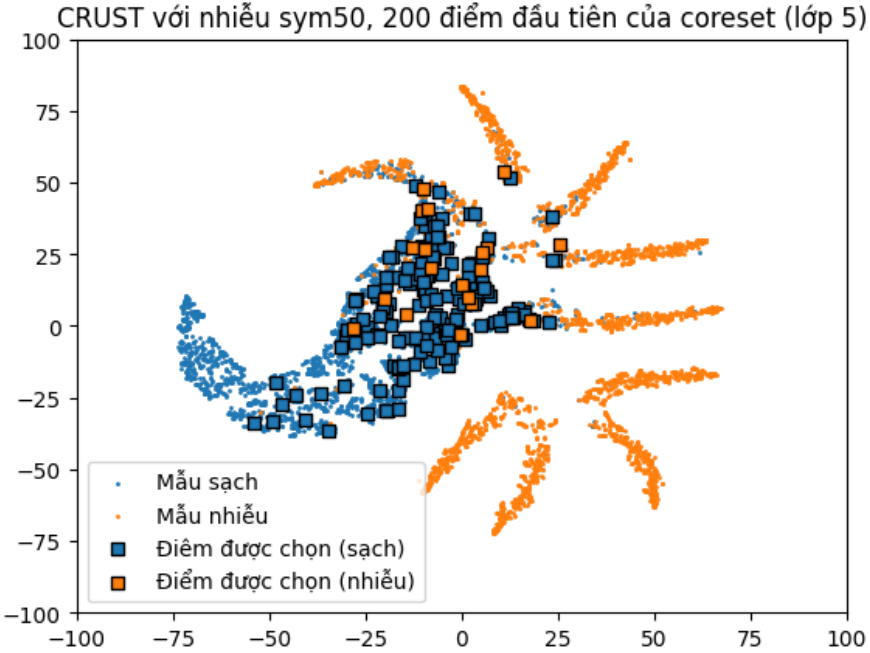


Phân bố của gradient trên tập dữ liệu CIFAR-10 (nhiễu 50%)

Phương pháp CRUST (Mirzasoleiman et al., 2020)

Thuật toán tìm cụm lớn nhất trên từng lớp:

Heuristic: các điểm thuộc cụm lớn thường có tổng khoảng cách đến các điểm khác nhỏ, các điểm thuộc cụm nhỏ thường có tổng khoảng cách đến các điểm khác lớn.



Minh họa cách hoạt động của thuật toán tìm cụm lớn nhất

Phương pháp CRUST (Mirzasoleiman et al., 2020)

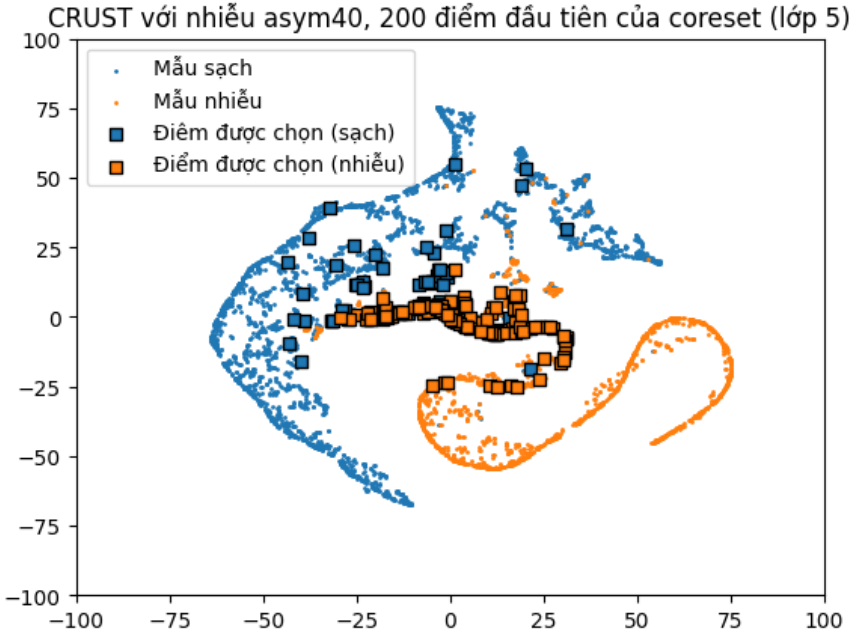
Thuật toán tìm cụm lớn nhất trên từng lớp:

- **Khởi tạo:**
  - Bắt đầu với tập rỗng $S = \emptyset$.
  - Xác định số điểm cần chọn k.
- **Lặp k lần:**
  - Chọn điểm i có tổng khoảng cách gradient đến các điểm còn lại nhỏ nhất trong các điểm chưa chọn của tập đang xét.
  - Thêm điểm i vào tập S.
- **Kết quả:** Trả về tập coreset S.

Phương pháp CRUST (Mirzasoleiman et al., 2020)

Vấn đề của phương pháp CRUST

Khi các cụm nhỏ hơn có kích thước khá lớn và nằm gần cụm lớn nhất, nó có thể chọn nhiều điểm thuộc cụm nhỏ.

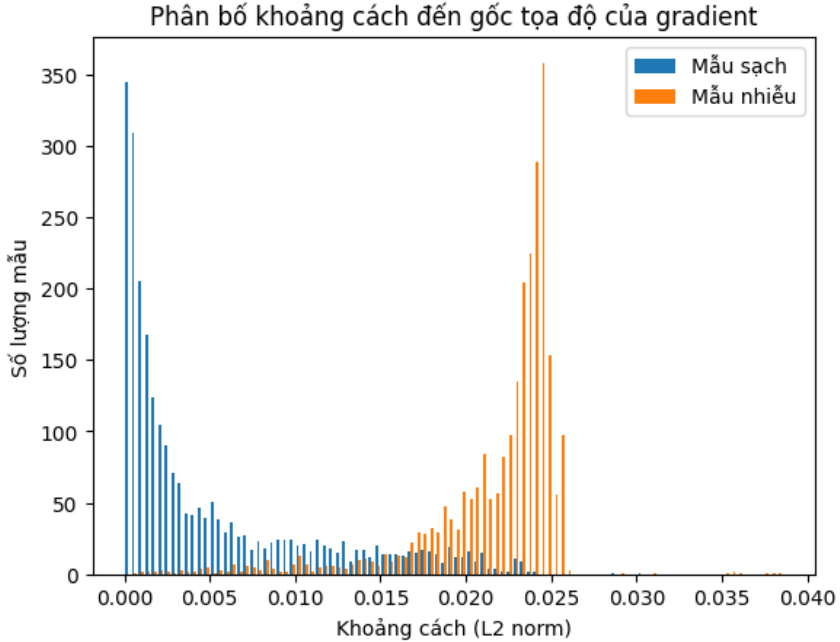


Minh họa cách hoạt động của thuật toán tìm cụm lớn nhất với mức nhiễu asym40, CIFAR10

Phương pháp CRUST (Mirzasoleiman et al., 2020)

Cải tiến phương pháp CRUST bằng cách thay đổi thuật toán tìm cụm lớn nhất

- Gradient dữ liệu sạch gần gốc tọa độ hơn dữ liệu nhiễu.
- Thay vì tính tổng khoảng cách đến tất cả các điểm thì tính khoảng cách đến gốc tọa độ.



Khoảng cách đến gốc tọa độ của gradient với dữ liệu nhiễu sym50, CIFAR 10

Nội dung

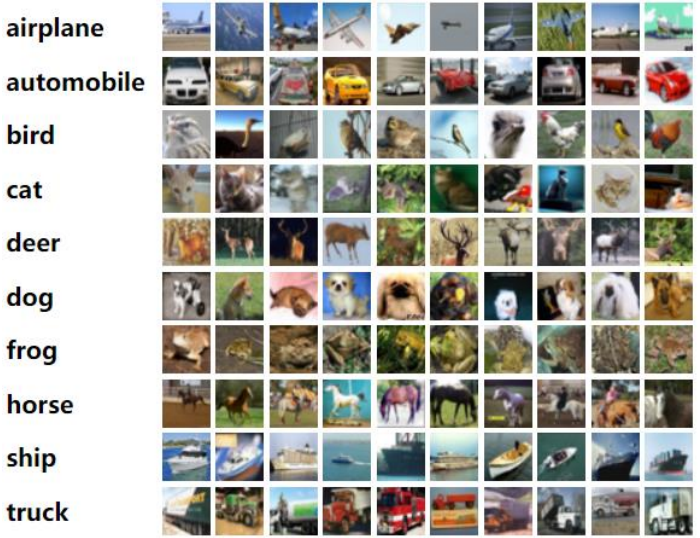
1. Giới thiệu đề tài
2. Mô hình mạng nơ-ron với phương pháp CRUST
3. **Thí nghiệm**
4. Tổng kết và hướng phát triển

Thí nghiệm với CIFAR

Dữ liệu:

- 50000 ảnh huấn luyện, 10000 ảnh validation.
- Thí nghiệm với 2 loại nhiễu khác nhau với các tỷ lệ nhiễu khác nhau:
  - Nhiễu đối xứng: 0.2, 0.5, 0.8

  (0.8 chỉ trên CIFAR-10)
  - Nhiễu không đối xứng: 0.4



Bộ dữ liệu CIFAR-10

Thí nghiệm với CIFAR

Thông tin khác:

- ResNet-32 trong 120 epoch với minibatch 128.
- SGD với learning rate 0.1, giảm tại epoch 80 và 100 với hệ số 10.
- Áp dụng momentum 0.9 và weight decay 5×10^{-4}.
- Tăng cường dữ liệu: đệm 4 pixel mỗi cạnh, cắt ngẫu nhiên ảnh 32×32, lật ngang với xác suất 0.5.
- Với CRUST, chọn coresets bằng 50% kích thước tập dữ liệu trừ khi có chỉ định khác.
- Sử dụng T4 GPU (Kaggle).

Thí nghiệm với CIFAR

Thí nghiệm 1: So sánh kết quả với bài báo gốc:

	CIFAR-10				CIFAR-100		
	Sym			Asym	Sym		Asym
	20	50	80	40	20	50	40
INCV (gốc)	89.7	84.8	52.3	86.0	60.2	53.1	50.7
CRUST (gốc)	**91.1**	86.3	**58.3**	**88.8**	**65.2**	**56.4**	**53.0**
INCV (chạy lại)	89.9	85.0	52.9	86.2	60.4	53.9	49.4
CRUST (cài đặt lại)	85.4	**86.5**	36.5	78.3	63.9	54.5	52.4

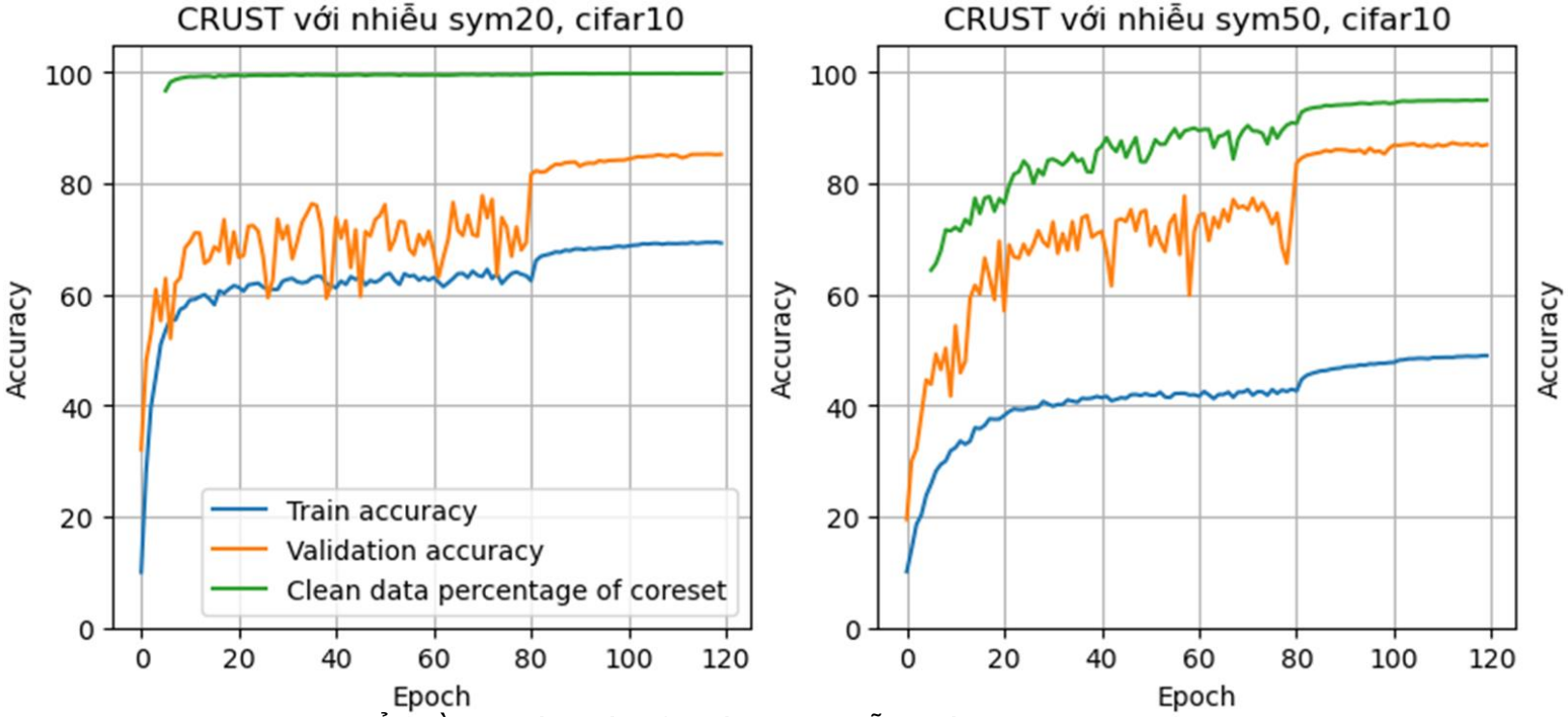
Thí nghiệm với CIFAR

Thí nghiệm 1: So sánh kết quả với bài báo gốc:

	CIFAR-10				CIFAR-100		
	Sym			Asym	Sym		Asym
	20	50	80	40	20	50	40
INCV (gốc)	89.7	84.8	52.3	86.0	60.2	53.1	50.7
CRUST (gốc)	**91.1**	86.3	**58.3**	**88.8**	**65.2**	**56.4**	**53.0**
INCV (chạy lại)	89.9	85.0	52.9	86.2	60.4	53.9	49.4
CRUST (cài đặt lại)	85.4	**86.5**	36.5	78.3	63.9	54.5	52.4

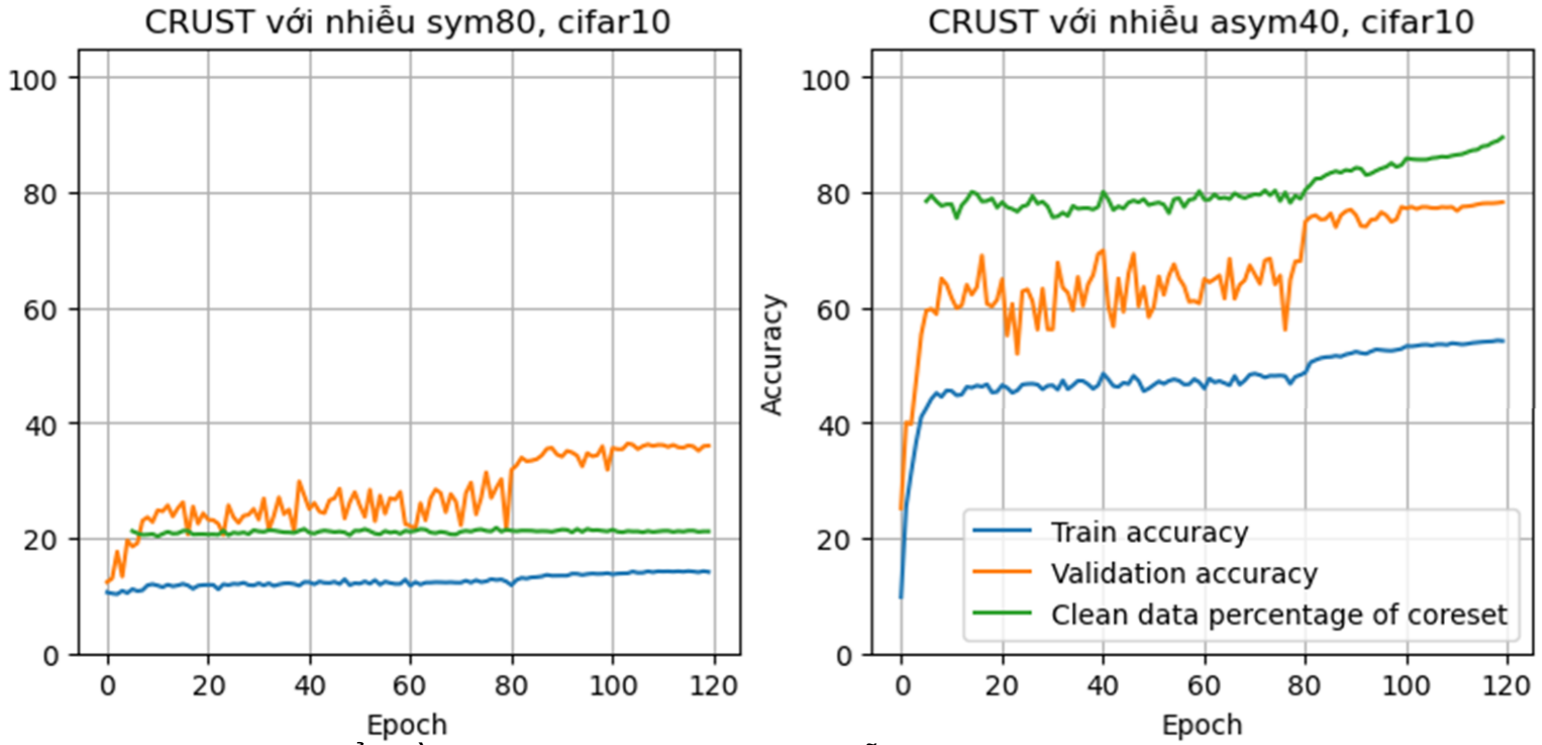
sym20 thấp hơn sym50

Thí nghiệm với CIFAR



Biểu đồ độ chính xác của các mức nhiễu khác nhau trên CIFAR-10.

Thí nghiệm với CIFAR



Biểu đồ độ chính xác của các mức nhiễu khác nhau trên CIFAR-10.

Thí nghiệm với CIFAR

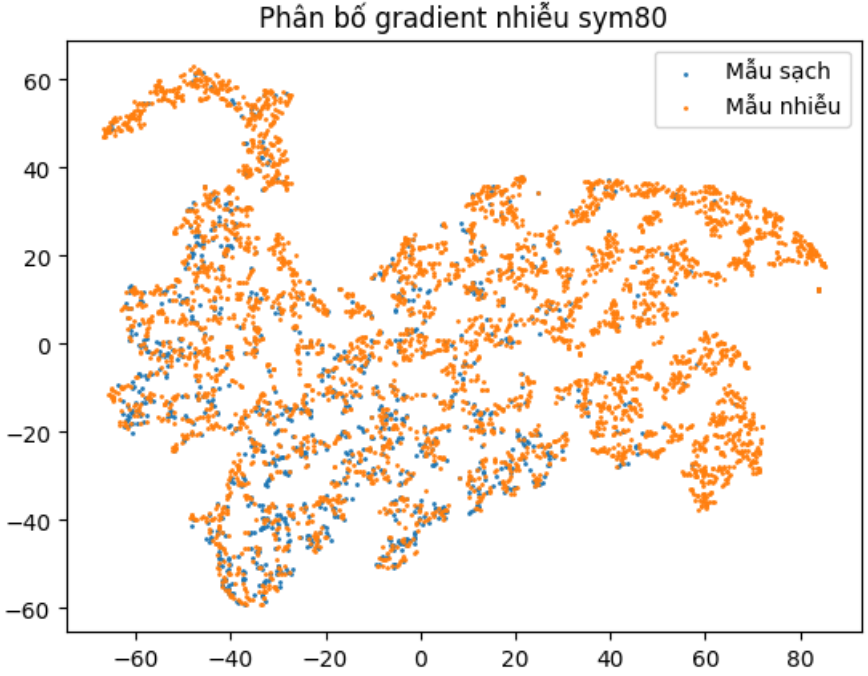
Thí nghiệm 1: So sánh kết quả với bài báo gốc:

	CIFAR-10				CIFAR-100		
	Sym			Asym	Sym		Asym
	20	50	80	40	20	50	40
INCV (gốc)	89.7	84.8	52.3	86.0	60.2	53.1	50.7
CRUST (gốc)	**91.1**	86.3	**58.3**	**88.8**	**65.2**	**56.4**	**53.0**
INCV (chạy lại)	89.9	85.0	52.9	86.2	60.4	53.9	49.4
CRUST (cài đặt lại)	85.4	**86.5**	36.5	78.3	63.9	54.5	52.4

Không hoạt động tốt trên sym80

Thí nghiệm với CIFAR

	CIFAR-10
	sym80
INCV (gốc)	52.3
CRUST (gốc)	**58.3**
INCV (chạy lại)	52.9
CRUST (cài đặt lại)	36.5



Phân bố gradient của nhiễu sym 80 của các mẫu có nhãn quan sát được là 5 sau khi huấn luyện 120 epoch

Thí nghiệm với CIFAR

Thí nghiệm 1: So sánh kết quả với bài báo gốc:

	CIFAR-10				CIFAR-100		
	Sym			Asym	Sym		Asym
	20	50	80	40	20	50	40
INCV (gốc)	89.7	84.8	52.3	86.0	60.2	53.1	50.7
CRUST (gốc)	**91.1**	86.3	**58.3**	**88.8**	**65.2**	**56.4**	**53.0**
INCV (chạy lại)	89.9	85.0	52.9	86.2	60.4	53.9	49.4
CRUST (cài đặt lại)	85.4	**86.5**	36.5	78.3	63.9	54.5	52.4

Thấp hơn INCV trên CIFAR-10 nhưng cao hơn INCV trên CIFAR-100 ở 1 số thí nghiệm

Thí nghiệm với CIFAR

Thí nghiệm 2: cải thiện kết quả của phương pháp CRUST bằng cách tinh chỉnh siêu tham số

	CIFAR-10				CIFAR-100		
	Sym			Asym	Sym		Asym
	20	50	80	40	20	50	40
CRUST (gốc)	**91.1**	86.3	**58.3**	**88.8**	**65.2**	**56.4**	**53.0**
CRUST (cài đặt lại) coreset size 50 learning rate 0.1	85.4	**86.5**	36.5	78.3	63.9	54.5	52.4
CRUST coreset size 50 learning rate tuned	85.4	**86.5**	40.2	78.3	63.9	54.5	52.4
CRUST coreset size tuned learning rate tuned	90.4	**86.5**	40.2	78.3	64.4	54.5	52.4

Thí nghiệm với CIFAR

Thí nghiệm 2: cải thiện kết quả của phương pháp CRUST bằng cách tinh chỉnh siêu tham số

- Thay đổi learning rate ban đầu trong hầu hết trường hợp đều không giúp tăng hiệu suất của mô hình
- Thay đổi kích thước coreset mang lại hiệu quả, nhưng chỉ tốt khi tỷ lệ nhiễu thấp.

Thí nghiệm 3: cải thiện kết quả của phương pháp CRUST bằng cách thay đổi thuật toán tìm cụm lớn nhất

	CIFAR-10				CIFAR-100		
	Sym			Asym	Sym		Asym
	20	50	80	40	20	50	40
CRUST (gốc)	**91.1**	86.3	**58.3**	**88.8**	65.2	56.4	**53.0**
CRUST (cài đặt lại) coreset size 50 learning rate 0.1	85.4	86.5	36.5	78.3	63.9	54.5	52.4
CRUST coreset size tuned learning rate tuned	90.4	86.5	40.2	78.3	64.4	54.5	52.4
CRUST (cải tiến) coreset size 50 learning rate tuned	84.5	**86.9**	47.0	86.3	62.6	**61.7**	51.2
CRUST (cải tiến) coreset size tuned learning rate tuned	90.3	**86.9**	48.4	86.3	**67.4**	**61.7**	51.2

Thí nghiệm 3: cải thiện kết quả của phương pháp CRUST bằng cách thay đổi thuật toán tìm cụm lớn nhất

	CIFAR-10				CIFAR-100		
	Sym			Asym	Sym		Asym
	20	50	80	40	20	50	40
CRUST (gốc)	**91.1**	86.3	**58.3**	**88.8**	65.2	56.4	**53.0**
CRUST (cài đặt lại) coreset size 50 learning rate 0.1	85.4	86.5	36.5	78.3	63.9	54.5	52.4
CRUST coreset size tuned learning rate tuned	90.4	86.5	40.2	78.3	64.4	54.5	52.4
CRUST (cải tiến) coreset size 50 learning rate tuned	84.5	**86.9**	47.0	86.3	62.6	**61.7**	51.2
CRUST (cải tiến) coreset size tuned learning rate tuned	90.3	**86.9**	48.4	86.3	**67.4**	**61.7**	51.2

Thí nghiệm 3: cải thiện kết quả của phương pháp CRUST bằng cách thay đổi thuật toán tìm cụm lớn nhất

	CIFAR-10				CIFAR-100		
	Sym			Asym	Sym		Asym
	20	50	80	40	20	50	40
CRUST (gốc)	**91.1**	86.3	**58.3**	**88.8**	65.2	56.4	**53.0**
CRUST (cài đặt lại) coreset size 50 learning rate 0.1	85.4	86.5	36.5	78.3	63.9	54.5	52.4
CRUST coreset size tuned learning rate tuned	90.4	86.5	40.2	78.3	64.4	54.5	52.4
CRUST (cải tiến) coreset size 50 learning rate tuned	84.5	**86.9**	47.0	86.3	62.6	**61.7**	51.2
CRUST (cải tiến) coreset size tuned learning rate tuned	90.3	**86.9**	48.4	86.3	**67.4**	**61.7**	51.2

Thí nghiệm với CIFAR

Thí nghiệm 3: cải thiện kết quả của phương pháp CRUST bằng cách thay đổi thuật toán tìm cụm lớn nhất

- Với mức kích thước coreset 50, CRUST cải tiến giúp tăng mạnh hiệu suất mô hình .
- Với kích thước coreset tối ưu nhất, CRUST cải tiến đạt hiệu quả tốt nhất, đa số các mức nhiễu đều có kết quả tăng khi chọn coreset phù hợp hơn.

Nội dung

1. Giới thiệu đề tài
2. Mô hình mạng nơ-ron với phương pháp CRUST
3. Thí nghiệm
4. **Tổng kết và hướng phát triển**

Tổng kết

Những điều mà chúng em đã làm trong khóa luận:

- Nghiên cứu về bài toán học có giám sát với dữ liệu nhiễu, hiểu rõ sự ảnh hưởng của dữ liệu nhiễu, tìm hiểu các phương pháp để giải quyết nó và ưu/nhược điểm các phương pháp đó.
- Tập trung nghiên cứu và tìm hiểu sâu phương pháp CRUST, một phương pháp bắt nguồn từ ý tưởng mới về sự phân bố gradient, nhưng đạt hiệu quả tốt.
- Đề xuất cải tiến mới cho CRUST, giúp tăng mạnh kết quả.

Hướng phát triển

- Áp dụng các phương pháp khác kết hợp với CRUST để có kết quả tốt hơn.
- Ứng dụng ý tưởng của phương pháp CRUST vào các bài toán khác.

Tài liệu tham khảo

- Baharan Mirzasoleiman, Kaidi Cao, and Jure Leskovec. “Coresets for Robust Training of Neural Networks against Noisy Labels”. In: Advances in Neural Information Processing Systems 33 (2020).
- Bo Han et al. “Co-teaching: Robust training of deep neural networks with extremely noisy labels”. In: NeurIPS. 2018, pp. 8535–8545
- Pengfei Chen et al. “Understanding and Utilizing Deep Neural Networks Trained with Noisy Labels”. In: International Conference on Machine Learning. 2019, pp. 1062–1070

Tài liệu tham khảo

- Anders Krogh and John A Hertz. “A simple weight decay can improve generalization”. In: Proc, NeurIPS. 1992, pp. 950–957.
- Chiyuan Zhang et al. “Understanding deep learning requires rethinking generalization”. In: Communications of the ACM 64 (2016).
- Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: Advances in neural information processing systems 25 (2012)

Cảm ơn thầy cô đã chú ý lắng nghe!