

**ĐẠI HỌC BÁCH KHOA HÀ NỘI**  
**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

---



**BÁO CÁO BÀI TẬP LỚN**  
**NHẬP MÔN CÁC PHƯƠNG PHÁP TỐI ƯU**

**Thuật toán tự thích nghi cho lập trình tựa lỗi  
và ứng dụng trong học máy**

**Giảng viên hướng dẫn: PGS. TS. Trịnh Ngọc Hải**  
(Khoa Toán - Tin)

**Danh sách sinh viên thực hiện:**

Nguyễn Ngọc Tuấn Anh – 202400029	Nguyễn Hồng Đạt – 202416153
Trần Tuấn Anh – 202416124	Phạm Đình Minh Đức – 202400038
Nguyễn Hữu Chính – 202416143	Bùi Tiến Dũng – 202416167
Trần Mạnh Cường – 202416147	Trần Huy Dương – 20230025
Đỗ Hải Đăng – 202400035	Nguyễn Khắc Duy – 202400100
Bùi Tuấn Đạt – 202400096	Nguyễn Hoàng Gia – 202400040

**HÀ NỘI, 12/2025**

# Mục lục

1	Giới thiệu . . . . .	2
2	Cơ sở lí thuyết . . . . .	3
3	Các kết quả chính . . . . .	4
4	Kết quả thực nghiệm . . . . .	7
5	Ứng dụng trong học máy . . . . .	7
6	Kết luận . . . . .	12

# 1 Giới thiệu

Các phương pháp gradient descent từ lâu đã đóng vai trò quan trọng trong việc giải các bài toán tối ưu, từ những bài toán tối ưu lồi đến những bài toán phi lồi phức tạp, và được ứng dụng rộng rãi trong nhiều lĩnh vực như tối ưu, xử lý tín hiệu và học máy (xem [3], [4], [10]). Ở mỗi vòng lặp, thuật toán gradient descent tìm nghiệm tối ưu thông qua hướng gradient và cỡ bước. Trong nhiều năm, phần lớn nghiên cứu tập trung vào việc cải thiện hướng cập nhật nhằm tăng tốc độ hội tụ, trong khi lựa chọn cỡ bước lại chủ yếu dựa trên các phương pháp quen thuộc như tìm kiếm theo đường thẳng (line-search) hoặc sử dụng hằng số Lipschitz (xem [3], [14]).

Tuy nhiên, cùng với sự xuất hiện của nhiều lĩnh vực ứng dụng học máy mới với dữ liệu có số chiều rất lớn và hàm mục tiêu không lồi, nhu cầu về những chiến lược chọn cỡ bước hiệu quả, ít tốn chi phí tính toán ngày càng trở nên quan trọng (xem [4], [10]). Dù tìm chính xác hay gần đúng, các phương pháp tìm kiếm theo đường thẳng đòi hỏi chi phí tính toán lớn trong mỗi bước lặp, đặc biệt trong những trường hợp việc tính giá trị của hàm gần như tương đương với việc tính đạo hàm của nó và đòi hỏi phải giải quyết các bài toán phụ phức tạp [3]. Ngược lại, các kỹ thuật sử dụng giá trị có sẵn từ trước như hằng số Lipschitz để tính cỡ bước thường thiếu ổn định, dẫn đến tốc độ hội tụ chậm hoặc không tối ưu. Một số nghiên cứu khác, như quy tắc chuỗi phân kỳ, cũng có hạn chế tương tự (xem [8], [14]).

Bên cạnh đó, nhiều phương pháp mở rộng của phương pháp gradient dành cho các hàm tựa lồi, giả lồi hoặc các bài toán có ràng buộc phức tạp đã được đề xuất trước đây, nhưng hiệu quả còn hạn chế. Ví dụ, phương pháp giảm dần cỡ bước cổ điển trong bài toán tựa lồi [8] dẫn tới tốc độ hội tụ rất chậm; các phương pháp khác yêu cầu hàm mục tiêu thỏa điều kiện Hölder (xem [20], [7]), hoặc chỉ hoạt động với tập ràng buộc bị chặn. Ngoài ra, những mô hình dựa trên mạng nơ-ron hồi quy (RNN) cho các bài toán lập trình giả lồi có ràng buộc lại sử dụng cỡ bước cố định mà không có sự điều chỉnh (xem [2], [11]).

Một số nghiên cứu trước đây đã đề xuất các thuật toán tự điều chỉnh cỡ bước nhằm cải thiện hiệu quả của các phương pháp gradient descent. Các cách tiếp cận này thường hoạt động tốt đối với các bài toán giả lồi khi tập ràng buộc bị chặn [9], và đã có những thuật toán hiệu quả trong trường hợp tập ràng buộc không bị chặn [6]. Tuy nhiên, các phương pháp đó vẫn chưa áp dụng được cho những bài toán không ràng buộc, do còn tồn tại các giới hạn trong cách xây dựng và phân tích cỡ bước.

Từ những hạn chế trên, bài báo [17] đề xuất một thuật toán mới nhằm xây dựng cơ chế điều chỉnh cỡ bước tự thích nghi, không cần tìm kiếm theo đường thẳng và có khả năng áp dụng cho một lớp rộng các bài toán tối ưu có hàm mục tiêu không lồi, trơn, và tập ràng buộc lồi, đóng nhưng không bị chặn. Điểm then chốt của thuật toán là giảm dần cỡ bước một cách có kiểm soát cho đến khi thỏa mãn một điều kiện nhất định. Mặc dù tính liên tục Lipschitz của gradient của hàm mục tiêu là điều kiện cần cho sự hội tụ, thuật toán được đề xuất không sử dụng hằng số Lipschitz để tính toán cỡ bước. Thuật toán cũng bao gồm phép chiếu trực giao, đảm bảo nghiệm tìm được luôn nằm trong tập ràng buộc.

Một trong những đóng góp quan trọng nhất của bài báo [17] là mở rộng kỹ thuật điều chỉnh cỡ bước tự thích nghi sang trường hợp tập ràng buộc không bị chặn. Tuy nhiên, việc mở rộng này gặp phải một số khó khăn. Thứ nhất, cần đảm bảo tính hội tụ của thuật toán mà không phải đưa thêm các điều kiện phụ trợ phức tạp. Các kết quả về sự hội tụ được chứng minh bằng cách khai thác các tính chất của hàm mục tiêu cùng với các biến đổi bất đẳng thức phù hợp. Thứ hai, mặc dù toán tử chiếu bảo đảm rằng điểm  $x^{k+1}$  được sinh ra từ  $x^k$  luôn nằm trong tập khả thi, ta vẫn phải chứng minh rằng sự xuất hiện của

phép chiếu không làm ảnh hưởng đến tính hội tụ của thuật toán. Cuối cùng, thuật toán thích nghi được đề xuất phải bao hàm cả trường hợp sử dụng cỡ bước cố định. Trường hợp này cho thấy thuật toán là một phương pháp mở rộng từ phương pháp Gradient Descent truyền thống và rất hữu ích trong các ứng dụng thực tế, đặc biệt đối với những hàm mục tiêu không lồi. Các ví dụ tính toán trên những bài toán có quy mô lớn với hàm mục tiêu không lồi đã củng cố cho nhận định này.

Phần chứng minh lý thuyết trong bài báo [17] cho thấy dãy nghiệm sinh ra bởi thuật toán hội tụ về điểm dừng của bài toán; trong trường hợp hàm tựa lồi hoặc giả lồi, nghiệm thu được còn là nghiệm tối ưu. Nhìn chung, các kết quả trong bài báo [17] khẳng định rằng cơ chế điều chỉnh cỡ bước của thuật toán đủ tốt để đảm bảo tính ổn định và hội tụ ngay cả trong bối cảnh tối ưu phi lồi và tập ràng buộc không bị chặn.

Để đánh giá hiệu quả, các tác giả thực hiện nhiều thí nghiệm bao gồm các bài toán phi lồi kinh điển, các bài toán có ràng buộc phức tạp, và các bài toán quy mô lớn. Ngoài ra, thuật toán còn được ứng dụng vào nhiều bài toán học máy như chọn đặc trưng có giám sát, hồi quy logistic đa biến và huấn luyện mạng nơ-ron. Kết quả cho thấy phương pháp đề xuất có độ chính xác tốt, tốc độ tính toán vượt trội so với các thuật toán gradient descent và các mô hình thần kinh động học (RNN) trong cùng điều kiện.

Mục tiêu của báo cáo này là trình bày lại các ý chính từ bài báo [17]. Phần tiếp theo của bài báo cáo gồm: Mục 2 giới thiệu cơ sở lý thuyết liên quan; Mục 3 mô tả thuật toán được đề xuất và chứng minh những kết quả quan trọng; Mục 4 phân tích các thực nghiệm; Mục 5 thảo luận các ứng dụng của thuật toán trong học máy; và Mục 6 đưa ra kết luận.

## 2 Cơ sở lý thuyết

Trong toàn bộ bài báo [17], ta giả định rằng  $C$  là một tập khác rỗng, đóng và lồi trong  $\mathbb{R}^m$ ,  $f : \mathbb{R}^m \rightarrow \mathbb{R}$  là một hàm khả vi trên một tập mở chứa  $C$ , ánh xạ  $\nabla f$  là liên tục Lipschitz, tức là tồn tại một hằng số  $L > 0$  sao cho  $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$  với mọi  $x, y \in C$ . Ta xem xét bài toán tối ưu hóa:

$$\min_{x \in C} f(x). \quad (\text{OP}(f, C))$$

Giả sử rằng tập nghiệm của  $\text{OP}(f, C)$  là không rỗng. Trước hết, ta nhắc lại một số định nghĩa và kết quả cơ bản sẽ được sử dụng trong phần tiếp theo của bài báo [17]. Chứng minh chi tiết được trình bày trong [1] và [15].

Với  $x \in \mathbb{R}^m$  ký hiệu bởi  $P_C(x)$  là hình chiếu của  $x$  lên  $C$ , tức là:

$$P_C(x) := \operatorname{argmin}\{\|z - x\| : z \in C\}.$$

**Mệnh đề 2.1** (Bauschke và Combettes [1]) *Ta có*

- (i)  $\|P_C(x) - P_C(y)\| \leq \|x - y\|$  với mọi  $x, y \in \mathbb{R}^m$ ,
- (ii)  $\langle y - P_C(x), x - P_C(x) \rangle \leq 0$  với mọi  $x \in \mathbb{R}^m, y \in C$ .

**Định nghĩa 2.1** (Mangasarian [13]) Hàm  $f : \mathbb{R}^m \rightarrow \mathbb{R}$  được gọi là:

- lồi trên  $C$  nếu với mọi  $x, y \in C, \lambda \in [0, 1]$ , thỏa mãn rằng

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y).$$

- giả lồi trên  $C$  nếu với mọi  $x, y \in C$ , thỏa mãn rằng

$$\langle \nabla f(x), y - x \rangle \geq 0 \Rightarrow f(y) \geq f(x).$$

- tựa lồi trên  $C$  nếu với mọi  $x, y \in C$ ,  $\lambda \in [0; 1]$ , thỏa mãn rằng

$$f(\lambda x + (1 - \lambda)y) \leq \max\{f(x); f(y)\}.$$

**Mệnh đề 2.2** (Dennis và Schnabel [5]) *Hàm khả vi  $f$  là tựa lồi trên  $C$  khi và chỉ khi*

$$f(y) \leq f(x) \Rightarrow \langle \nabla f(x), y - x \rangle \leq 0.$$

Dáng chú ý là “ $f$  là lồi”  $\Rightarrow$  “ $f$  là giả lồi”  $\Rightarrow$  “ $f$  là tựa lồi” (xem Mangasarian [13]).

**Mệnh đề 2.3** (Dennis và Schnabel [5]) *Giả sử rằng  $\nabla f$  liên tục Lipschitz trên  $C$ . Với mọi  $x, y \in C$ , ta có*

$$|f(y) - f(x) - \langle \nabla f(x), y - x \rangle| \leq \frac{L}{2} \|y - x\|^2.$$

**Bổ đề 2.1** (Xu [19]) *Cho  $\{a_k\}; \{b_k\} \subset (0; \infty)$  là các dãy sao cho*

$$a_{k+1} \leq a_k + b_k \quad \forall k \geq 0; \quad \sum_{k=0}^{\infty} b_k < \infty.$$

*Khi đó, tồn tại giới hạn  $\lim_{k \rightarrow \infty} a_k = c \in \mathbb{R}$ .*

### 3 Các kết quả chính

Trong mục này, ta trình bày thuật toán đề xuất và chứng minh sự hội tụ của nó. Dựa trên các mệnh đề và bổ đề đã nêu ở Mục 2, ta đưa ra kết quả chính như sau.

**Thuật toán 1 (Thuật toán Thích nghi Gradient Descent - GDA)**

**Bước 0.** Chọn  $x^0 \in C$ ,  $\lambda_0 \in (0, +\infty)$ ,  $\sigma, \kappa \in (0, 1)$ . Đặt  $k = 0$ .

**Bước 1.** Cho  $x^k$  và  $\lambda_k$ , tính toán  $x^{k+1}$  và  $\lambda_{k+1}$  như sau:

$$x^{k+1} = P_C(x^k - \lambda_k \nabla f(x^k)),$$

Nếu  $f(x^{k+1}) \leq f(x^k) - \sigma \langle \nabla f(x^k), x^k - x^{k+1} \rangle$  thì đặt  $\lambda_{k+1} := \lambda_k$ , ngược lại đặt  $\lambda_{k+1} := \kappa \lambda_k$ .

**Bước 2.** Cập nhật  $k := k + 1$ . Nếu  $x^{k+1} = x^k$  thì DỪNG, ngược lại quay về Bước 1.

**Nhận xét 3.1** Nếu Thuật toán GDA dừng tại bước  $k$ , thì  $x^k$  là một điểm dừng của bài toán OP(f, C).

Thật vậy, vì  $x^{k+1} = P_C(x^k - \lambda_k \nabla f(x^k))$ , áp dụng Mệnh đề 2.1-(ii), ta có:

$$\langle z - x^{k+1}, x^k - \lambda_k \nabla f(x^k) - x^{k+1} \rangle \leq 0 \quad \forall z \in C. \quad (1)$$

Nếu  $x^{k+1} = x^k$ , ta có:

$$\langle \nabla f(x^k), z - x^k \rangle \geq 0 \quad \forall z \in C, \quad (2)$$

điều này có nghĩa là  $x^k$  là một điểm dừng của bài toán. Bên cạnh đó, nếu  $f$  là hàm giả lồi thì từ (2), ta suy ra  $f(z) \geq f(x^k)$  với mọi  $z \in C$ , hay nói cách khác  $x^k$  là một nghiệm của bài toán OP(f, C).

Bây giờ, giả sử rằng thuật toán tạo ra một dãy vô hạn. Ta sẽ chứng minh rằng dãy này hội tụ đến một nghiệm của bài toán OP(f, C).

**Định lý 3.1** *Giả sử rằng dãy  $\{x^k\}$  được tạo ra bởi Thuật toán GDA. Khi đó, dãy  $\{f(x^k)\}$  hội tụ và mỗi điểm giới hạn (nếu có) của dãy  $\{x^k\}$  là một điểm dừng của bài toán. Hơn nữa,*

- nếu  $f$  là tựa lồi trên  $C$ , thì dãy  $\{x^k\}$  hội tụ đến một điểm dừng của bài toán.
- nếu  $f$  là giả lồi trên  $C$ , thì dãy  $\{x^k\}$  hội tụ đến một nghiệm của bài toán.

*Chứng minh.* Áp dụng Mệnh đề 2.3, ta có

$$f(x^{k+1}) \leq f(x^k) + \langle \nabla f(x^k), x^{k+1} - x^k \rangle + \frac{L}{2} \|x^{k+1} - x^k\|^2. \quad (3)$$

Ở (1), lấy  $z = x^k \in C$ , ta có

$$\langle \nabla f(x^k), x^{k+1} - x^k \rangle \leq -\frac{1}{\lambda_k} \|x^{k+1} - x^k\|^2. \quad (4)$$

Kết hợp (3) và (4), ta thu được

$$f(x^{k+1}) \leq f(x^k) - \sigma \langle \nabla f(x^k), x^k - x^{k+1} \rangle - \left( \frac{1-\sigma}{\lambda_k} - \frac{L}{2} \right) \|x^{k+1} - x^k\|^2. \quad (5)$$

Bây giờ ta khẳng định rằng  $\{\lambda_k\}$  không tiến về 0, hay nói cách khác, kích thước cỡ bước thay đổi hữu hạn lần. Thật vậy, giả sử ngược lại rằng  $\lambda_k \rightarrow 0$ . Từ (5), tồn tại  $k_0 \in \mathbb{N}$  thỏa mãn

$$f(x^{k+1}) \leq f(x^k) - \sigma \langle \nabla f(x^k), x^k - x^{k+1} \rangle \quad \forall k \geq k_0.$$

Theo cách xây dựng  $\lambda_k$ , bất đẳng thức cuối cùng cho ta thấy rằng  $\lambda_k = \lambda_{k_0}$  với mọi  $k \geq k_0$ . Điều này là mâu thuẫn. Và do đó, tồn tại  $k_1 \in \mathbb{N}$  sao cho với mọi  $k \geq k_1$ , ta có  $\lambda_k = \lambda_{k_1}$  và

$$f(x^{k+1}) \leq f(x^k) - \sigma \langle \nabla f(x^k), x^k - x^{k+1} \rangle. \quad (6)$$

Lưu ý rằng  $\langle \nabla f(x^k), x^k - x^{k+1} \rangle \geq 0$ , ta suy ra rằng dãy  $\{f(x^k)\}$  là hội tụ và

$$\sum_{k=0}^{\infty} \langle \nabla f(x^k), x^k - x^{k+1} \rangle < \infty; \quad \sum_{k=0}^{\infty} \|x^{k+1} - x^k\|^2 < \infty. \quad (7)$$

Từ (1), với mọi  $z \in C$ , ta có

$$\begin{aligned} \|x^{k+1} - z\|^2 &= \|x^k - z\|^2 - \|x^{k+1} - x^k\|^2 + 2\langle x^{k+1} - x^k, x^{k+1} - z \rangle \\ &\leq \|x^k - z\|^2 - \|x^{k+1} - x^k\|^2 + 2\lambda_k \langle \nabla f(x^k), z - x^{k+1} \rangle. \end{aligned} \quad (8)$$

Gọi  $\bar{x}$  là một điểm giới hạn của  $\{x^k\}$ . Tồn tại một dãy con  $\{x^{k_i}\} \subset \{x^k\}$  sao cho  $\lim_{i \rightarrow \infty} x^{k_i} = \bar{x}$ . Trong (8), đặt  $k = k_i$  và lấy giới hạn khi  $i \rightarrow \infty$ . Lưu ý rằng  $\|x^k - x^{k+1}\| \rightarrow 0$ ,  $\nabla f$  là liên tục, ta có

$$\langle \nabla f(\bar{x}), z - \bar{x} \rangle \geq 0 \quad \forall z \in C,$$

điều này có nghĩa là  $\bar{x}$  là một điểm dừng của bài toán.

Bây giờ, giả sử rằng  $f$  là tựa lồi trên  $C$ . Ký hiệu

$$U := \{x \in C : f(x) \leq f(x^k) \quad \forall k \geq 0\}.$$

Lưu ý rằng  $U$  chứa tập nghiệm của OP(f, C), nên nó không rỗng. Lấy  $\hat{x} \in U$ . Vì  $f(x^k) \geq f(\hat{x})$  với mọi  $k \geq 0$ , ta có

$$\langle \nabla f(x^k), \hat{x} - x^k \rangle \leq 0 \quad \forall k \geq 0. \quad (9)$$

Kết hợp (8) và (9), ta có

$$\|x^{k+1} - \hat{x}\|^2 \leq \|x^k - \hat{x}\|^2 - \|x^{k+1} - x^k\|^2 + 2\lambda_k \langle \nabla f(x^k), x^k - x^{k+1} \rangle. \quad (10)$$

Áp dụng Bổ đề 2.1 với  $a_k = \|x^k - \hat{x}\|^2$ ,  $b_k = 2\lambda_k \langle \nabla f(x^k), x^k - x^{k+1} \rangle$ , ta suy ra rằng dãy  $\{\|x^k - \hat{x}\|\}$  là hội tụ với mọi  $\hat{x} \in U$ . Vì dãy  $\{x^k\}$  là bị chặn, tồn tại một dãy con  $\{x^{k_i}\} \subset \{x^k\}$  sao cho  $\lim_{i \rightarrow \infty} x^{k_i} = \bar{x} \in C$ . Từ (6), ta biết rằng dãy  $\{f(x^k)\}$  là không tăng và hội tụ. Điều này ngụ ý rằng  $\lim_{k \rightarrow \infty} f(x^k) = f(\bar{x})$  và  $f(\bar{x}) \leq f(x^k)$  với mọi  $k \geq 0$ . Điều này có nghĩa là  $\bar{x} \in U$  và dãy  $\{\|x^k - \bar{x}\|\}$  là hội tụ.

Do đó,

$$\lim_{k \rightarrow \infty} \|x^k - \bar{x}\| = \lim_{i \rightarrow \infty} \|x^{k_i} - \bar{x}\| = 0.$$

Lưu ý rằng mỗi điểm giới hạn của  $\{x^k\}$  là một điểm dừng của bài toán. Khi đó, toàn bộ dãy  $\{x^k\}$  hội tụ về  $\bar{x}$  – một điểm dừng của bài toán. Hơn nữa, khi  $f$  là giả lồi, điểm dừng này trở thành một nghiệm của OP(f, C).  $\square$

**Nhận xét 3.2** Trong Thuật toán GDA, ta có thể chọn  $\lambda_0 = \lambda$ , với hằng số  $\lambda \leq 2(1-\sigma)/L$ . Khi đó, ta có  $(1-\sigma)/\lambda_0 - L/2 \geq 0$ . Kết hợp với (5), điều này ngụ ý rằng điều kiện  $f(x^{k+1}) \leq f(x^k) - \sigma \langle \nabla f(x^k), x^k - x^{k+1} \rangle$  được thỏa mãn và cỡ bước  $\lambda_k = \lambda$  cho mọi bước  $k$ . Do đó, Thuật toán GDA vẫn có thể áp dụng cho cỡ bước là hằng số  $\lambda \leq 2(1-\sigma)/L$ . Đối với bất kỳ  $\lambda \in (0, 2/L)$ , tồn tại  $\sigma \in (0, 1)$  sao cho  $\lambda \leq 2(1-\sigma)/L$ . Kết quả là, nếu giá trị của hằng số Lipschitz  $L$  đã được biết trước, ta có thể chọn cỡ bước là hằng số  $\lambda \in (0, 2/L)$  như trong thuật toán gradient descent (GD) để giải các bài toán quy hoạch lồi. Thuật toán GD này đã được đề xuất trong các công trình trước đây. Vì nó là một trường hợp đặc biệt của Thuật toán GDA, sự hội tụ của nó được đảm bảo như các khẳng định trong Định lý 3.1.

## Thuật toán 2 (Thuật toán Gradient Descent - GD)

**Bước 0.** Chọn  $x^0 \in C$ ,  $\lambda \in (0, 2/L)$ . Đặt  $k = 0$ .

**Bước 1.** Cho  $x^k$ , tính toán  $x^{k+1}$  như sau:

$$x^{k+1} = P_C(x^k - \lambda \nabla f(x^k)).$$

**Bước 2.** Cập nhật  $k := k + 1$ . Nếu  $x^{k+1} = x^k$  thì DỪNG, ngược lại quay về Bước 1.

Lưu ý rằng tất cả các khẳng định của Định lý 3.1 vẫn đúng cho dãy  $\{x^k\}$  được tạo ra bởi Thuật toán GD. Bây giờ, ta ước lượng tốc độ hội tụ của Thuật toán GDA trong việc giải các bài toán tối ưu hóa không ràng buộc.

**Hệ quả 3.1** Giả sử rằng  $f$  là lồi,  $C = \mathbb{R}^m$  và  $\{x^k\}$  là dãy được tạo ra bởi Thuật toán GDA. Khi đó,

$$f(x^k) - f(x^*) = O\left(\frac{1}{k}\right)$$

trong đó  $x^*$  là một nghiệm của bài toán.

*Chứng minh.* Gọi  $x^*$  là một nghiệm của bài toán. Ký hiệu  $\Delta_k := f(x^k) - f(x^*)$ . Từ (6), lưu ý rằng  $x^k - x^{k+1} = \lambda_k \nabla f(x^k)$ , ta có

$$\Delta_{k+1} \leq \Delta_k - \sigma \lambda_{k_1} \|\nabla f(x^k)\|^2 \quad \forall k \geq k_1. \quad (11)$$

Mặt khác, vì dãy  $\{x^k\}$  bị chặn và  $f$  là lồi, ta có

$$\begin{aligned} 0 \leq \Delta_k &\leq \langle \nabla f(x^k), x^k - x^* \rangle \\ &\leq M \|\nabla f(x^k)\|, \end{aligned} \quad (12)$$

trong đó  $M := \sup\{\|x^k - x^*\| : k \geq k_1\} < \infty$ . Từ (11) và (12), ta có

$$\Delta_{k+1} \leq \Delta_k - Q\Delta_k^2 \quad \forall k \geq k_1, \quad (13)$$

trong đó  $Q := \frac{\sigma\lambda_{k_1}}{M^2}$ . Lưu ý rằng  $\Delta_{k+1} \leq \Delta_k$ , từ (13), ta thu được

$$\frac{1}{\Delta_{k+1}} \geq \frac{1}{\Delta_k} + Q \geq \dots \geq \frac{1}{\Delta_{k_1}} + (k - k_1)Q,$$

điều này ngụ ý rằng

$$f(x^k) - f(x^*) = O\left(\frac{1}{k}\right).$$

□

Để kết thúc phần này, bài báo [17] trình bày một biến thể ngẫu nhiên của Thuật toán GDA để áp dụng trong học sâu quy mô lớn. Xét bài toán:

$$\min_x \mathbb{E}[f_\xi(x)],$$

trong đó  $\xi$  là tham số ngẫu nhiên và hàm  $f_\xi$  là L-smooth (tức là gradient của nó liên tục Lipschitz). Ta đang tạo ra một gradient ngẫu nhiên  $\nabla f_{\xi^k}(x^k)$  bằng cách lấy mẫu  $\xi^k$  tại mỗi lần lặp  $k$ . Biến thể ngẫu nhiên của phương pháp gradient descent, đặc biệt là trong bối cảnh học sâu quy mô lớn, đóng vai trò quan trọng trong việc tối ưu hóa các mô hình phức tạp một cách hiệu quả. Khi ta xem xét bài toán tối ưu hóa có dạng:

$$\min_x \mathbb{E}[f_\xi(x)],$$

trong đó  $x$  đại diện cho các tham số của mô hình (như các trọng số trong mạng nơ-ron),  $\xi$  là một tham số ngẫu nhiên, và  $f_\xi(x)$  là một hàm L-smooth, chúng tôi đang đối phó với một kịch bản nơi hàm mục tiêu được định nghĩa là giá trị kỳ vọng của một số hàm ngẫu nhiên  $f_\xi(x)$  nào đó. Khuôn khổ này là điển hình trong học máy, nơi  $f_\xi(x)$  thường đại diện cho hàm mất mát (loss function) được tính toán trên một tập con (lô/batch) của dữ liệu huấn luyện, và  $\xi$  đại diện cho tính ngẫu nhiên trong việc lựa chọn tập con này.

Dưới đây là mô tả chi tiết của thuật toán SGDA. Các tác giả dành lại các kết quả lý thuyết của Thuật toán SGDA cho các nghiên cứu sau này.

**Thuật toán 3 (Thuật toán Thích nghi Gradient Descent Ngẫu nhiên - SGDA)**

**Bước 0.** Chọn  $x^0 \in C$ ,  $\lambda_0 \in (0, +\infty)$ ,  $\sigma, \kappa \in (0, 1)$ . Đặt  $k = 0$ .

**Bước 1.** Lấy mẫu  $\xi^k$  và tính toán  $x^{k+1}$  và  $\lambda_{k+1}$  như sau:

$$x^{k+1} = P_C(x^k - \lambda_k \nabla f_{\xi^k}(x^k)),$$

Nếu  $f_{\xi^k}(x^{k+1}) \leq f_{\xi^k}(x^k) - \sigma \langle \nabla f_{\xi^k}(x^k), x^k - x^{k+1} \rangle$  thì đặt  $\lambda_{k+1} := \lambda_k$ , ngược lại đặt  $\lambda_{k+1} := \kappa \lambda_k$ .

**Bước 2.** Cập nhật  $k := k + 1$ . Nếu  $x^{k+1} = x^k$  thì DỪNG, ngược lại quay về Bước 1.

## 4 Kết quả thực nghiệm

## 5 Ứng dụng trong học máy

Phương pháp được đề xuất, giống như thuật toán GD, có nhiều ứng dụng trong học máy. Trong báo cáo này, nhóm chúng tôi trình bày và tái triển khai ba thực nghiệm ứng với



ba ứng dụng tiêu biểu được trình bày trong bài báo [17], bao gồm lựa chọn đặc trưng có giám sát (*supervised feature selection*), hồi quy (*regression*) và phân loại (*classification*), để chứng minh độ chính xác và hiệu quả tính toán so với các phương pháp thay thế khác.

Đầu tiên, bài toán lựa chọn đặc trưng có thể được mô hình hóa như một bài toán cực tiểu hóa một hàm phân thức giả lồi trên một tập lồi, đây là một lớp con của Bài toán OP(f, C). Bài toán này được sử dụng để so sánh phương pháp tiếp cận được đề xuất với các phương pháp thần kinh động (*neurodynamic approaches*).

Thứ hai, vì bài toán hồi quy logistic đa biến là một bài toán quy hoạch lồi, nên thuật toán GDA và các biến thể có sẵn của thuật toán GD có thể được sử dụng để giải nó.

Cuối cùng, một mô hình mạng nơ-ron cho bài toán phân loại ảnh cũng giống như một bài toán quy hoạch với hàm mục tiêu không lồi cũng không tựa lồi. Để huấn luyện mô hình này, các tác giả sử dụng biến thể ngẫu nhiên của phương pháp GDA (Thuật toán SGDA) như một kỹ thuật heuristic. Mặc dù sự hội tụ của thuật toán không thể được đảm bảo như trong các trường hợp hàm mục tiêu giả lồi và tựa lồi, nghiên cứu lý thuyết đã chỉ ra rằng nếu dãy điểm có một điểm giới hạn, nó sẽ hội tụ về một điểm dừng của bài toán (xem Định lý 3.1). Các thực nghiệm tính toán chỉ ra rằng phương pháp được đề xuất vượt trội hơn so với các phương pháp thần kinh động và gradient descent hiện có.

## 5.1 Chọn đặc trưng có giám sát

Bài toán chọn đặc trưng được thực hiện trên tập dữ liệu với tập gồm  $p$  đặc trưng  $\mathcal{F} = \{F_1, \dots, F_p\}$  và tập gồm  $n$  mẫu  $\{(x_i, y_i) \mid i = 1, \dots, n\}$ , trong đó  $x_i = (x_{i1}, \dots, x_{ip})^T$  là vectơ đặc trưng  $p$  chiều của mẫu thứ  $i$  và  $y_i \in \{1, \dots, m\}$  đại diện cho các nhãn tương ứng chỉ ra các lớp hoặc giá trị mục tiêu. Trong [18], một tập con tối ưu gồm  $k$  đặc trưng  $\{F_1, \dots, F_k\} \subseteq \mathcal{F}$  được chọn với độ dư thừa thấp nhất và độ liên quan cao nhất đến lớp mục tiêu  $y$ . Độ dư thừa của đặc trưng được đặc trưng bởi một ma trận bán xác định dương  $Q$ . Khi đó, mục tiêu đầu tiên là cực tiểu hóa hàm bậc hai lồi  $w^T Q w$ . Độ liên quan của đặc trưng được đo lường bởi  $\rho^T w$ , trong đó  $\rho = (\rho_1, \dots, \rho_p)^T$  là một vectơ tham số liên quan. Do đó, mục tiêu thứ hai là cực đại hóa hàm tuyến tính  $\rho^T w$ . Kết hợp hai mục tiêu này, ta suy ra bài toán tương đương như sau:

$$\begin{aligned} \min \quad & \frac{w^T Q w}{\rho^T w} \\ \text{v.đ.k} \quad & e^T w = 1 \\ & w \geq 0, \end{aligned} \tag{14}$$

trong đó  $w = (w_1, \dots, w_p)^T$  là vectơ điểm số đặc trưng cần xác định. Vì hàm mục tiêu của bài toán (14) có dạng phân thức, trong đó tử là một hàm lồi và mẫu là một hàm tuyến tính dương, nên nó là giả lồi trên tập ràng buộc. Do đó, chúng ta có thể giải bài toán (14) bằng Thuật toán GDA.

Trong thực nghiệm, bài báo [17] triển khai các thuật toán với tập dữ liệu Parkinsons, bao gồm 23 đặc trưng và 197 mẫu, được tải xuống tại <https://archive.ics.uci.edu/ml/datasets/parkinsons>. Ma trận hệ số tương đồng  $Q$  được xác định bởi  $Q = \delta I_p + S$  [18], trong đó  $S = (s_{ij})$  là ma trận  $p \times p$  với:

$$s_{ij} = \max \left\{ 0, \frac{I(F_i; F_j; y)}{H(F_i) + H(F_j)} \right\},$$

entropy thông tin của một vectơ biến ngẫu nhiên  $\hat{X}$  là:

$$H(\hat{X}) = - \sum_{\hat{x} \in \hat{X}} p(\hat{x}) \log p(\hat{x}),$$

đa thông tin (multi-information) của ba vectơ ngẫu nhiên  $\hat{X}, \hat{Y}, \hat{Z}$  là  $I(\hat{X}; \hat{Y}; \hat{Z}) = I(\hat{X}; \hat{Y}) - I(\hat{X}; \hat{Y} \mid \hat{Z})$  với thông tin tương hỗ của hai vectơ ngẫu nhiên  $\hat{X}, \hat{Y}$  được định nghĩa bởi:

$$I(\hat{X}; \hat{Y}) = \sum_{\hat{x} \in \hat{X}} \sum_{\hat{y} \in \hat{Y}} p(\hat{x}, \hat{y}) \log \frac{p(\hat{x}, \hat{y})}{p(\hat{x})p(\hat{y})}$$

và thông tin tương hỗ có điều kiện giữa  $\hat{X}, \hat{Y}$  và  $\hat{Z}$  được định nghĩa bởi:

$$I(\hat{X}; \hat{Y} \mid \hat{Z}) = \sum_{\hat{x} \in \hat{X}} \sum_{\hat{y} \in \hat{Y}} \sum_{\hat{z} \in \hat{Z}} p(\hat{x}, \hat{y}, \hat{z}) \log \frac{p(\hat{x}, \hat{y} \mid \hat{z})}{p(\hat{x} \mid \hat{z})p(\hat{y} \mid \hat{z})}.$$

Vectơ độ liên quan đặc trưng  $\rho = (\rho_1, \dots, \rho_p)^T$  được xác định bởi điểm Fisher:

$$\rho(F_i) = \frac{\sum_{j=1}^K n_j (\mu_{ij} - \mu_i)^2}{\sum_{j=1}^K n_j \sigma_{ij}^2},$$

trong đó  $n_j$  biểu thị số lượng mẫu trong lớp  $j$ ,  $\mu_{ij}$  biểu thị giá trị trung bình của đặc trưng  $F_i$  cho các mẫu trong lớp  $j$ ,  $\mu_i$  là giá trị trung bình của đặc trưng  $F_i$ , và  $\sigma_{ij}^2$  biểu thị giá trị phương sai của đặc trưng  $F_i$  cho các mẫu trong lớp  $j$ .

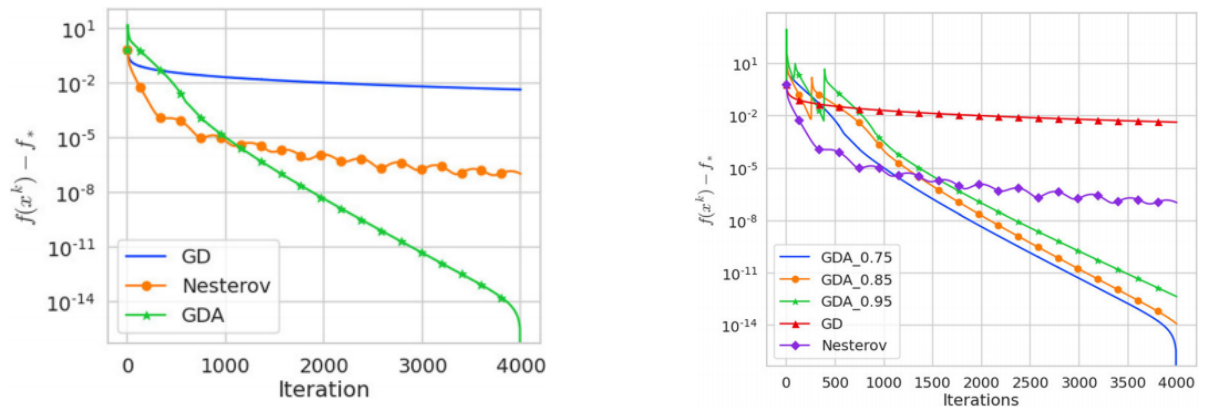
Giá trị tối ưu xấp xỉ của bài toán (14) là  $f(w^*) = 0.153711$  với thời gian tính toán  $T = 6.096260s$  cho thuật toán đề xuất, trong khi  $f(w^*) = 0.154013$ ,  $T = 11.030719$  cho Thuật toán RNN. So sánh với Thuật toán RNN, thuật toán được đề xuất trong bài báo [17] vượt trội hơn cả về độ chính xác và thời gian tính toán.

## 5.2 Hồi quy logistic đa biến

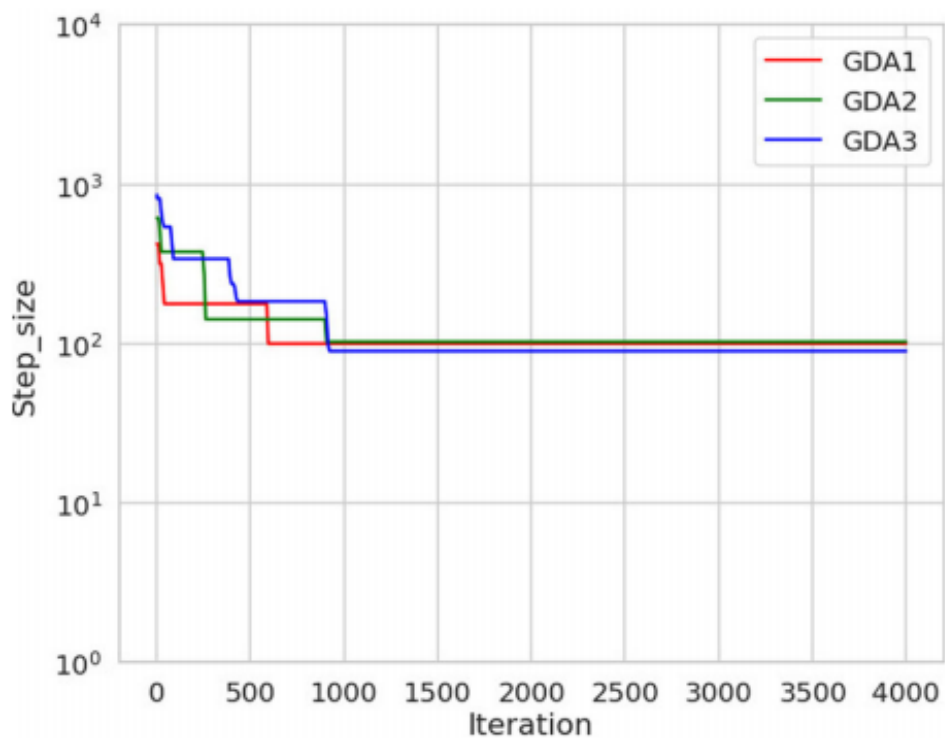
Các thực nghiệm được thực hiện với tập dữ liệu bao gồm  $N$  quan sát  $(\mathbf{a}_i, b_i) \in \mathbb{R}^d \times \mathbb{R}, i = 1, \dots, n$ . Hàm mất mát cross-entropy cho hồi quy logistic đa biến được cho bởi  $J(x) = -\sum_{i=1}^N (b_i \log(\sigma(-x^T \mathbf{a}_i)) + (1 - b_i) \log(1 - \sigma(-x^T \mathbf{a}_i)))$ , trong đó  $\sigma$  là hàm sigmoid. Kết hợp với điều chuẩn  $\ell_2$ , chúng ta có hàm mất mát được điều chuẩn  $\bar{J}(x) = J(x) + \frac{1}{2N} \|x\|^2$ . Hệ số Lipschitz  $L$  được ước tính bởi  $\frac{1}{2N} (\|A\|^2/2 + 1)$ , trong đó  $A = (a_1^T, \dots, a_n^T)^T$ . Trong bài báo [17], các tác giả so sánh các thuật toán huấn luyện bài toán hồi quy logistic bằng cách sử dụng các tập dữ liệu Mushrooms và W8a (xem Malitsky và Mishchenko 2020) [12]. Phương pháp GDA được so sánh với thuật toán GD có kích thước bước là  $1/L$  và phương pháp tăng tốc Nesterov. Chúng tôi đã tiến hành triển khai lại thí nghiệm, kết quả chi tiết được minh họa trong Hình 3 và 4. Kết quả thu được hoàn toàn tương thích với kết quả đã trình bày trong bài báo [17]. Các hình này cho thấy rằng Thuật toán GDA vượt trội hơn Thuật toán GD và phương pháp tăng tốc Nesterov về giá trị hàm mục tiêu trong các lần lặp. Đặc biệt, Hình 4 thể hiện sự thay đổi của giá trị hàm mục tiêu theo các hệ số  $\kappa$  khác nhau. Trong hình này, ký hiệu "GDA\_0.75" tương ứng với trường hợp  $\kappa = 0.75$ . Hình 5 trình bày việc giảm kích thước bước từ kích thước bước ban đầu tương ứng với kết quả trong Hình 4.

## 5.3 Mạng nơ-ron cho phân loại

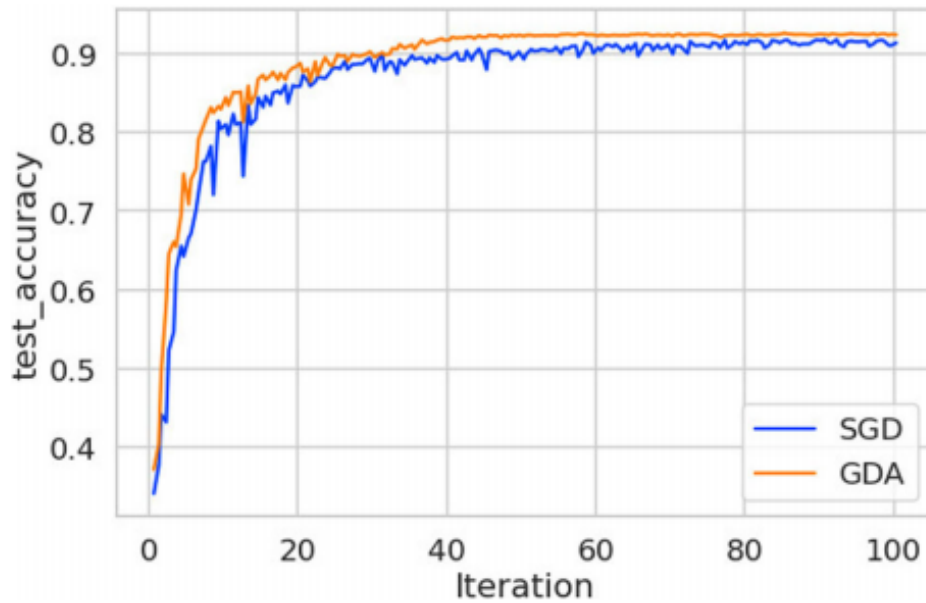
Để cung cấp một ví dụ về cách thuật toán đề xuất có thể được triển khai vào một mô hình huấn luyện mạng nơ-ron, các tác giả đã sử dụng các kiến trúc ResNet-18 tiêu chuẩn đã được triển khai trong PyTorch và huấn luyện chúng để phân loại các hình ảnh được lấy từ tập dữ liệu Cifar10 (được tải xuống tại <https://www.cs.toronto.edu/~kriz/cifar.html>), trong khi xem xét hàm mất mát cross-entropy. Trong các nghiên cứu với ResNet-18, các tác giả đã sử dụng các thiết lập mặc định của Adam cho các tham số của nó.



**Hình 1** Kết quả tính toán cho hồi quy logistic với tập dữ liệu Mushrooms

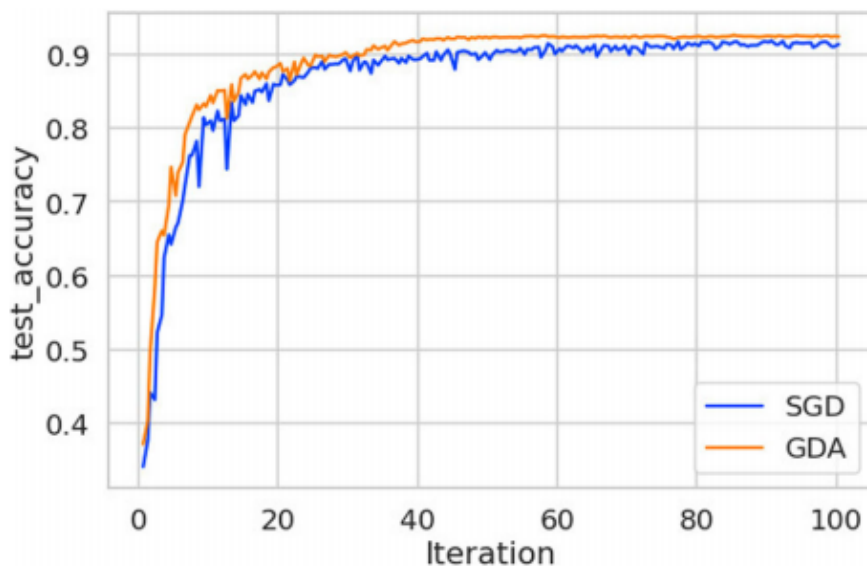


**Hình 2** Kết quả tính toán hồi quy logistic với tập dữ liệu W8a

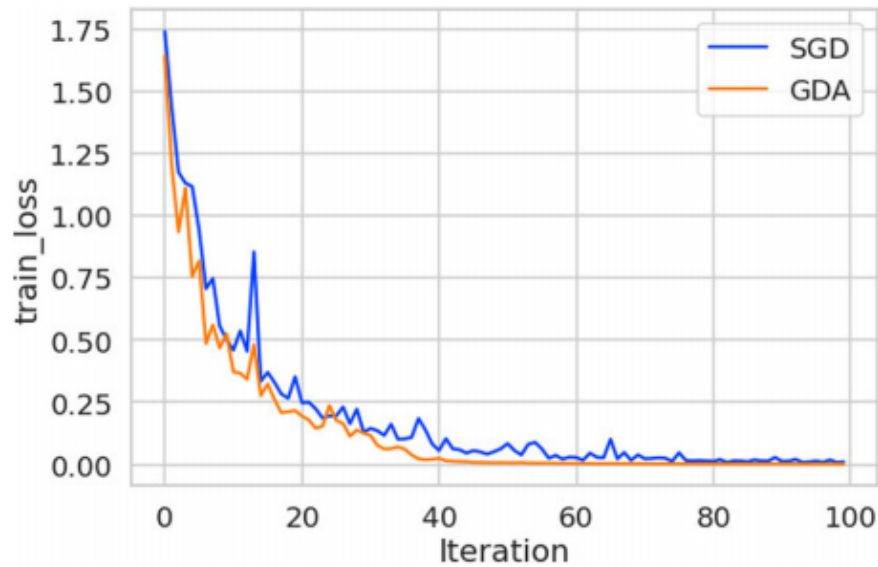


**Hình 3** Các bước nhảy thay đổi cho mỗi lần lặp trong hồi quy logistic với tập dữ liệu W8a

Để huấn luyện mô hình mạng nơ-ron này, bài báo [17] sử dụng biến thể ngẫu nhiên của phương pháp GDA (Thuật toán SGDA) để so sánh với các thuật toán Stochastic Gradient Descent (SGD). Các kết quả tính toán được hiển thị trong Hình 6 và 7. Tương tự như kết quả đã công bố trong bài báo [17], các kết quả của chúng tôi cho thấy rằng Thuật toán SGDA vượt trội hơn so với Thuật toán SGD thông thường. Cụ thể, SGDA đạt được độ chính xác kiểm tra (testing accuracy) cao hơn và độ mất mát huấn luyện (train loss) thấp hơn trong cùng số vòng lặp.



**Hình 4** Độ chính xác kiểm thử qua các lần lặp của mô hình ResNet-18



**Hình 5** Mức độ mất mát dữ liệu trong quá trình huấn luyện qua các lần lặp của mô hình ResNet-18.

## 6 Kết luận

Bài báo cáo đã mô tả lại và phân tích một thuật toán gradient descent với cơ chế điều chỉnh cỡ bước tự thích nghi, cho phép áp dụng hiệu quả trên các bài toán tối ưu phi lồi và tập ràng buộc lồi, đóng nhưng không bị chặn. Thuật toán không cần tìm kiếm theo đường thẳng hay biết trước hằng số Lipschitz, mà tự động điều chỉnh cỡ bước trong quá trình lặp, giúp giảm chi phí tính toán và tăng tốc độ hội tụ. Các kết quả lý thuyết và thí nghiệm số cho thấy phương pháp được đề xuất ổn định và đạt hiệu suất cao trên các bài toán lớn, bao gồm cả ứng dụng trong học máy như chọn đặc trưng có giám sát, hồi quy logistic đa biến và huấn luyện mạng nơ-ron. Nhìn chung, thuật toán cung cấp một công cụ linh hoạt và hiệu quả cho nhiều bài toán tối ưu phức tạp, đồng thời mở ra hướng nghiên cứu tiếp theo cho các bài toán tối ưu đa mục tiêu và hàm mục tiêu không trơn [16].

# Tài liệu tham khảo

- [1] Bauschke HH, Combettes PL. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer, 2011.
- [2] Bian W, Ma L, Qin S, Xue X. Neural network for nonsmooth pseudoconvex optimization with general convex constraints. *Neural Networks*, 101:1–14, 2018.
- [3] Boyd SP, Vandenberghe L. *Convex Optimization*. Cambridge University Press, Cambridge, 2009.
- [4] Cevher V, Becker S, Schmidt M. Convex optimization for big data. *IEEE Signal Processing Magazine*, 31:32–44, 2014.
- [5] Dennis JE, Schnabel RB. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, New Jersey, 1983.
- [6] Ferreira OP, Sosa WS. On the frank–wolfe algorithm for non-compact constrained optimization problems. *Optimization*, 71(1):197–211, 2022.
- [7] Hu Y, Li J, Yu CK. Convergence rates of subgradient methods for quasiconvex optimization problems. *Computational Optimization and Applications*, 77:183–212, 2020.
- [8] Kiwiel KC. Convergence and efficiency of subgradient methods for quasiconvex minimization. *Mathematical Programming, Series A*, 90(1):1–25, 2001.
- [9] Konnov IV. Simplified versions of the conditional gradient method. *Optimization*, 67(12):2275–2290, 2018.
- [10] Lan GH. *First-order and Stochastic Optimization Methods for Machine Learning*. Springer Series in the Data Sciences. Springer Nature, 2020.
- [11] Liu N, Wang J, Qin S. A one-layer recurrent neural network for nonsmooth pseudoconvex optimization with quasiconvex inequality and affine equality constraints. *Neural Networks*, 147:1–14, 2022.
- [12] Malitsky Y, Mishchenko K. Adaptive gradient descent without descent. In *Proceedings of Machine Learning Research*, volume 119, pages 6702–6712, 2020.
- [13] Mangasarian O. Pseudo-convex functions. *SIAM Journal on Control*, 8:281–289, 1965.
- [14] Nesterov Y. *Introductory Lectures on Convex Optimization: A Basic Course*, volume 87. Springer Science & Business Media, 2013.
- [15] Rockafellar RT. *Convex Analysis*. Princeton University Press, Princeton, 1970.

- [16] Thang TN, Solanki VK, Dao TA, Anh NTN, Hai PV. A monotonic optimization approach for solving strictly quasiconvex multiobjective programming problems. *Journal of Intelligent & Fuzzy Systems*, 38:6053–6063, 2020.
- [17] Trần Ngọc Thăng, Trịnh Ngọc Hải. Self-adaptive algorithms for quasiconvex programming and applications to machine learning. *Computational and Applied Mathematics*, 43(249), 2024.
- [18] Wang Y, Li X, Wang J. A neurodynamic optimization approach to supervised feature selection via fractional programming. *Neural Networks*, 136:194–206, 2021.
- [19] Xu HK. Iterative algorithms for nonlinear operators. *Journal of the London Mathematical Society*, 66:240–256, 2002.
- [20] Yu CK, Hu Y, Yang X, Choy SK. Abstract convergence theorem for quasi-convex optimization problems with applications. *Optimization*, 68(7):1289–1304, 2019.