Tạp chí Khoa học Trường Đại học Cần Thơ
website: sj.ctu.edu.vn

GIẢI THUẬT GIẢM GRADIENT NGẪU NHIÊN CHO PHÂN LỚP DỮ LIỆU LỚN

Đỗ Thanh Nghị[1]
[1] Khoa Công nghệ Thông tin & Truyền thông, Trường Đại học Cần Thơ

Thông tin chung:
Ngày nhận: 19/09/2015
Ngày chấp nhận: 10/10/2015

Title:
Stochastic gradient descent for classifying very large datasets

Từ khóa:
Máy học véc-tơ hỗ trợ (SVM), giảm gradient ngẫu nhiên (SGD), phân lớp dữ liệu lớn

Keywords:
Support vector machines (SVM), Stochastic gradient descent (SGD), classifying very large datasets

ABSTRACT

In this paper, we present the support vector machines algorithm using the stochastic gradient descent for classifying very large datasets. To reach the sparsity in the solution, the support vector machines algorithm uses the hinge loss in classification tasks. Thus, the direct optimization using the stochastic gradient descent is difficult due to the differentiation of the hinge loss. Our proposal is to substitute the hinge loss used in the problem formula of the support vector machines algorithm by the smooth ones to improve the convergence rate of the stochastic gradient descent. The numerical test results on two large textual datasets (RCV1, twitter) show that our approach is more efficient than the usual hinge loss.

TÓM TẮT

Trong bài viết, chúng tôi trình bày giải thuật giảm gradient ngẫu nhiên sử dụng trong máy học véc-tơ hỗ trợ cho phân lớp dữ liệu lớn. Máy học véc-tơ hỗ trợ sử dụng hàm hinge loss trong phân lớp nhằm đạt được tính chất thưa trong lời giải. Tuy nhiên, do hàm hinge loss không khả vi là nguyên nhân làm chậm hội tụ đến lời giải khi áp dụng giải thuật giảm gradient ngẫu nhiên. Chúng tôi nghiên cứu thay thế hàm hinge loss được sử dụng trong vấn đề tối ưu của giải thuật máy học véc-tơ hỗ trợ bằng các hàm xấp xỉ, khả vi nhằm cải tiến tốc độ hội tụ của giải thuật giảm gradient ngẫu nhiên. Kết quả thực nghiệm trên 2 tập dữ liệu văn bản lớn (RCV1, twitter) cho thấy hiệu quả của đề xuất sử dụng hàm xấp xỉ so với hàm hinge loss.

1 GIỚI THIỆU

Máy học véc-tơ hỗ trợ (SVM (Vapnik, 1995)) là lớp mô hình máy học hữu hiệu để giải quyết các vấn đề phân lớp, hồi quy, phát hiện phần tử cá biệt. Máy học SVM đã được áp dụng thành công trong rất nhiều ứng dụng như nhận dạng mặt người, phân loại văn bản, phân loại bệnh ung thư (tham khảo tại (Guyon, 1999)). Bằng việc kết hợp với phương pháp hàm nhân, máy học SVM cung cấp các mô hình hiệu quả chính xác cho các vấn đề phân lớp và hồi quy phi tuyến trong thực tế. Mặc dù có được những ưu điểm kể trên, việc huấn luyện của giải thuật máy học SVM rất mất thời gian và tiêu tốn nhiều không gian bộ nhớ do phải giải bài toán quy hoạch toàn phương. Độ phức tạp tối thiểu huấn luyện của giải thuật máy học SVM luôn là bậc 2 so với số lượng phần tử dữ liệu (Platt, 1999). Do đó, cần thiết phải có những cải tiến để giải thuật học SVM có thể xử lý được các tập dữ liệu với kích thước lớn về số phần tử cũng như số chiều.

Để cải tiến việc huấn luyện giải thuật máy học SVM cho các tập dữ liệu lớn. Các công trình nghiên cứu trong (Boser *et al*., 1992), (Chang & Lin, 2011), (Osuna *et al*., 1997), (Platt, 1998) đã chia bài toán quy hoạch toàn phương gốc thành các bài toán con để giải quyết. Nghiên cứu của (Mangasarian, 2001), (Suykens & Vandewalle, 1999) đã thay đổi bài toán quy hoạch toàn phương

phức tạp của giải thuật máy học SVM chuẩn về giải hệ phương trình tuyến tính đơn giản hơn. Nghiên cứu của (Liu *et al*., 1999), (Poulet & Do, 2004) đã đề nghị xây dựng giải thuật học tăng trưởng, chỉ nạp dữ liệu từng phần rồi cập nhật mô hình theo dữ liệu mà không cần nạp toàn bộ tập dữ liệu trong bộ nhớ. Công trình nghiên cứu của (Do & Poulet, 2004), (Do & Poulet, 2006), (Do & Poulet, 2008) đề nghị giải thuật song song để cải thiện tốc độ huấn luyện. (Tong & Koller, 2000), (Do & Poulet, 2005) đề nghị phương pháp chọn tập con dữ liệu thay vì phải học trên toàn bộ tập dữ liệu gốc. (Do & Fekete, 2007) kết hợp boosting (Freund & Schapire, 1999), arcing (Breiman, 1997) để cải thiện tốc độ xây dựng mô hình SVM chỉ tập trung vào những mẫu khó phân lớp.

Nghiên cứu của chúng tôi trong bài viết này nhằm phát triển từ ý tưởng sử dụng giải thuật giảm gradient ngẫu nhiên (SGD) để giải trực tiếp vấn đề tối ưu của máy học SVM, được đề xuất bởi (Bottou & Bousquet, 2008) và (Shalev-Shwartz *et al*., 2007). Tuy nhiên, vấn đề tối ưu của máy học SVM có hàm hinge loss không khả vi là nguyên nhân ảnh hưởng đến hiệu quả của giải thuật SGD. Chúng tôi đề xuất thay thế hàm hinge loss bằng các hàm xấp xỉ, khả vi (Rennie, 2004) để cải tiến tốc độ hội tụ của giải thuật SGD. Kết quả thực nghiệm trên 2 tập dữ liệu văn bản lớn RCV1 (Bottou & Bousquet, 2008), twitter (Go *et al*., 2009) cho thấy hiệu quả của đề xuất sử dụng hàm xấp xỉ so với hàm hinge loss.

Phần tiếp theo của bài được tổ chức như sau. Phần 2 sẽ trình bày tóm tắt về máy học SVM, giải thuật SGD sử dụng trong SVM và thay thế hàm hinge loss bằng các hàm xấp xỉ khả vi, cải tiến tốc độ hội tụ của giải thuật SGD. Kết quả chạy thử nghiệm sẽ được trình bày trong phần 3 trước khi kết thúc bằng kết luận và hướng phát triển.

2 GIẢI THUẬT GIẢM GRADIENT NGẪU NHIÊN CHO VẤN ĐỀ PHÂN LỚP CỦA MÁY HỌC VÉC-TƠ HỖ TRỢ

2.1 Phân lớp với máy học véc-tơ hỗ trợ

Xét ví dụ phân lớp nhị phân tuyến tính như Hình 1. Cho m phần tử $x_1, x_2, \ldots, x_m$ trong không gian n chiều với nhãn (lớp) của các phần tử tương ứng là $y_1, y_2, \ldots, y_m$ có giá trị 1 hoặc -1. Nhãn $y_i = 1$ khi x_i thuộc lớp $+1$ (lớp dương, lớp chúng ta quan tâm) và $y_i = -1$, nếu x_i thuộc lớp -1 (lớp âm hay các lớp còn lại). SVM tìm siêu phẳng tối ưu (xác định bởi véc tơ pháp tuyến w và độ lệch của siêu phẳng b) dựa trên 2 siêu phẳng hỗ trợ của 2 lớp. Các phần tử lớp $+1$ nằm bên phải của siêu phẳng hỗ trợ cho lớp $+1$, các phần tử lớp -1 nằm phía bên trái của siêu phẳng hỗ trợ cho lớp -1. Những phần tử nằm ngược phía với siêu phẳng hỗ trợ được coi như lỗi. Khoảng cách lỗi được biểu diễn bởi $z_i \geq 0$ (với x_i nằm đúng phía của siêu phẳng hỗ trợ của nó thì khoảng cách lỗi tương ứng $z_i = 0$, còn ngược lại thì $z_i > 0$ là khoảng cách từ điểm x_i đến siêu phẳng hỗ trợ tương ứng của nó). Khoảng cách giữa 2 siêu phẳng hỗ trợ được gọi là lề. Siêu phẳng tối ưu (nằm giữa 2 siêu phẳng hỗ trợ) tìm được từ 2 tiêu chí là cực đại hóa lề (lề càng lớn, mô hình phân lớp càng an toàn) và cực tiểu hóa lỗi. Vấn đề dẫn đến việc giải bài toán quy hoạch toàn phương (1):

$$\min \Psi(w, b, z) = (1/2)\,||w||^2 + c\sum_{i=1}^{m} z_i$$

$$s.t. \quad (1)$$

$$y_i(w.x_i - b) + z_i \geq 1$$

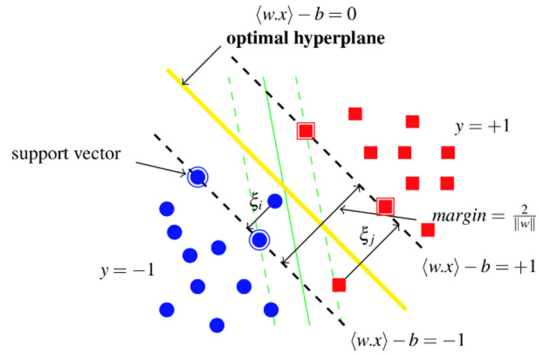
$$z_i \geq 0 \quad (i=1,m)$$

hằng $c > 0$ sử dụng để chỉnh độ rộng lề và lỗi

Giải bài toán quy hoạch toàn phương (1), thu được (w, b). Phân lớp phần tử x dựa vào biểu thức $sign(w.x - b)$.

Mặc dù giải thuật SVM cơ bản chỉ giải quyết được bài toán phân lớp tuyến tính, tuy nhiên nếu ta kết hợp SVM với phương pháp hàm nhân, sẽ cho phép giải quyết lớp các bài toán phân lớp phi tuyến (Cristianini & Shawe-Taylor, 2000).

(Platt, 1998) chỉ ra rằng các giải thuật huấn luyện được đề xuất trong (Boser *et al*., 1992), (Chang & Lin, 2011), (Osuna *et al*., 1997), (Platt, 1998) có độ phức tạp tính toán lời giải bài toán quy hoạch toàn phương (1) tối thiểu là $O(m^2)$ trong đó m là số lượng phần tử được dùng để huấn luyện. Điều này làm cho giải thuật SVM không phù hợp với dữ liệu lớn.



Hình 1: Phân lớp tuyến tính với máy học SVM

2.2 Giải thuật giảm gradient ngẫu nhiên

Một cài đặt cho giải thuật SVM của (Bottou & Boussquet, 2008), (Shalev-Shwartz *et al*., 2007) dựa trên phương pháp giảm gradient ngẫu nhiên (SGD), có độ phức tạp tuyến tính với số phần tử dữ liệu. Từ các ràng buộc (không xét độ lệch *b*) trong (1), có thể viết lại như sau:

$$z_i \geq 1 - y_i(w.x_i) \quad (2)$$

$$z_i \geq 0 \quad (i=1,m) \quad (3)$$

Các ràng buộc (2), (3) có thể được viết ngắn gọn như (4):

$$z_i = max\{0, 1 - y_i(w.x_i)\} \quad (4)$$

Bằng cách thay thế z_i vào hàm mục tiêu của (1), việc tìm siêu phẳng tối ưu của SVM có thể được thực hiện bởi (5):

$$min\ \Psi(w, x, y) = (\lambda/2)\ ||w||^2 + (1/m) \sum_{i=1}^{m} \max\{0,1 - y_i(w.x_i)\} \quad (5)$$

Phương pháp giảm gradient (GD) thực hiện tối ưu vấn đề (5) bằng cách cập nhật w tại lần lặp thứ $(t+1)$ với tốc độ học η_t, như trong (6):

$$w_{t+1} = w_t - (\eta_t /m) \sum_{i=1}^{m} \nabla_w \psi(w_t, x_i, y_i) \quad (6)$$

Phương pháp giảm gradient ngẫu nhiên (SGD) thực hiện đơn giản bước cập nhật w_{t+1} chỉ sử dụng một phần tử ngẫu nhiên (x_t, y_t) tại mỗi lần lặp:

$$w_{t+1} = w_t - \eta_t \nabla_w \psi(w_t, x_t, y_t) \quad (7)$$

Có thể thấy rằng giải thuật SGD đơn giản, thực hiện các bước lặp, mỗi bước lặp chỉ lấy 1 phần tử ngẫu nhiên từ tập dữ liệu, thực hiện cập nhật w thay vì phải giải bài toán quy hoạch toàn phương (1). Giải thuật SGD có độ phức tạp tuyến tính với số phần tử của tập dữ liệu học, phân lớp dữ liệu có số phần tử và số chiều lớn rất hiệu quả.

2.3 Thay thế hàm hinge loss bởi hàm xấp xỉ liên tục

Đại lượng lỗi $z_i = max\{0, 1 - y_i(w.x_i)\}$ trong vấn đề tối ưu (5) của máy học SVM thường được gọi là hàm lỗi hinge loss được viết dưới dạng:

$$L^{hinge}(x) = max\{0, 1 - x\} \quad (8)$$

Chú ý rằng hàm hinge loss không khả vi tại $y_i(w.x_i)=1$. Điều này ảnh hưởng đến tốc độ hội tụ đến lời giải của giải thuật SGD.

Chúng tôi đề xuất thay thế hàm hinge loss bằng các hàm xấp xỉ khả vi để cải tiến tốc độ hội tụ của giải thuật SGD. Một hàm xấp xỉ khả vi của hinge loss, được gọi là smooth hinge loss (Rennie, 2004) có dạng như sau:

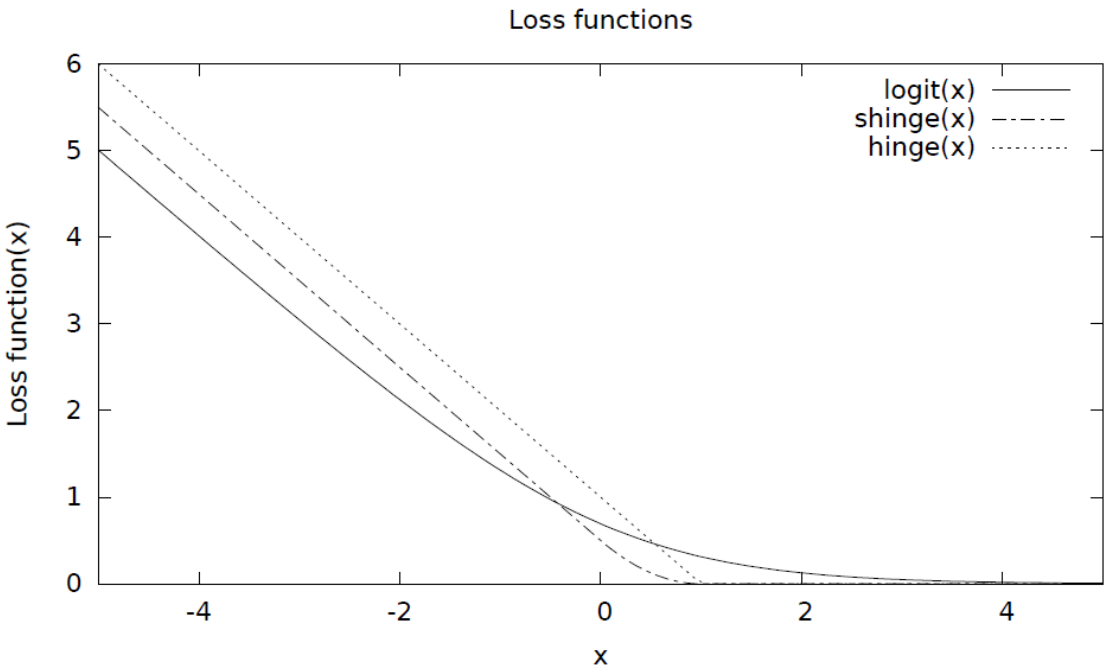
$$L^{shinge}(x) = \begin{cases} \frac{1}{2} - x & if\ (x \leq 0) \\ \frac{1}{2}(1-x)^2 & if\ (0 < x < 1) \\ 0 & if\ (x \geq 1) \end{cases} \quad (9)$$

Ngoài ra, có thể sử dụng hàm xấp xỉ khả vi khác của hinge loss đó là hàm logistic loss (logit), có dạng như sau:

$$L^{logit}(x) = log(1 + e^{-x}) \quad (10)$$

Hình 2 là đồ thị của hàm hinge loss so với hai hàm xấp xỉ khả vi là smooth hinge loss và logit loss.

Quan sát trên đồ thị, có thể thấy rằng hàm smooth hinge loss và logit loss là hàm khả vi. Nên khi thay thế cho hàm hinge loss có thể đảm bảo được tốc độ hội tụ của giải thuật SGD trong giải trực tiếp vấn đề (5) của máy học SVM. Mặc dù logit loss là hàm trơn (smooth) nhất nhưng smooth hinge loss cũng đủ trơn và vẫn duy trì được tính chất thưa trong lời giải như hàm hinge loss.



Hình 2: So sánh hàm hinge loss với hàm xấp xỉ liên tục smooth hinge loss, logit loss

3 KẾT QUẢ THỰC NGHIỆM

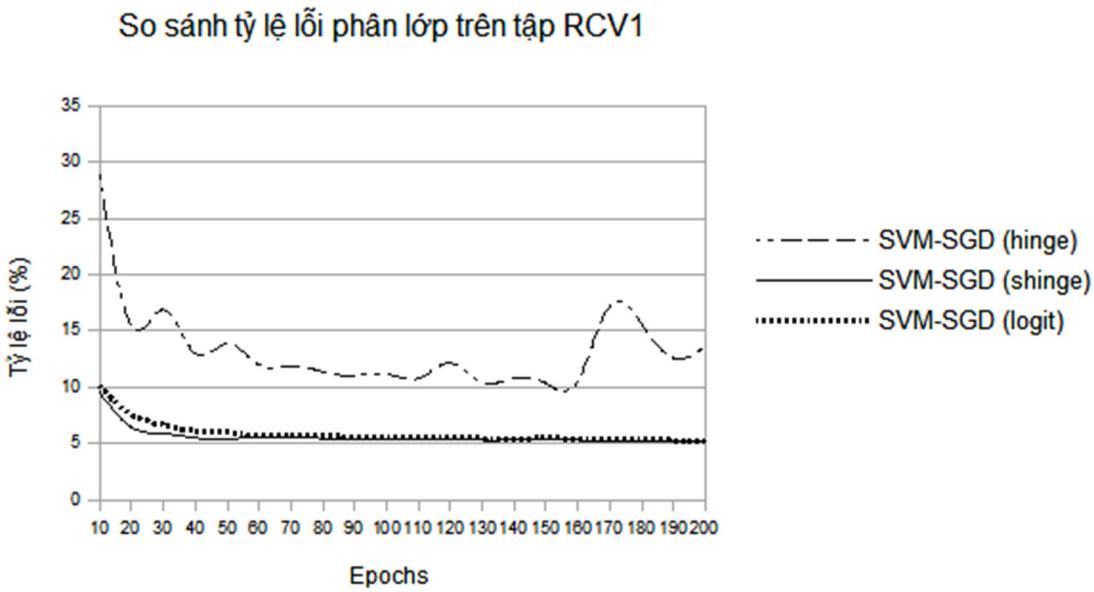
Chúng tôi tiến hành đánh giá hiệu quả của máy học SVM với 2 hàm xấp xỉ khả vi của hinge loss được giải trực tiếp bởi SGD. Chúng tôi đã cài đặt giải thuật SVM-SGD sử dụng 2 hàm xấp xỉ khả vi (smooth hinge loss, logit loss) bằng ngôn ngữ lập trình C/C++. Ngoài ra, chúng tôi cũng cần so sánh với SVM-SGD gốc sử dụng hinge loss (Bottou & Boussquet, 2008), (Shalev-Shwartz *et al*., 2007). Tất cả các giải thuật đều được thực hiện trên một máy tính cá nhân (Intel 3GHz, 4GB RAM) chạy hệ điều hành Linux (Fedora Core 20).

Chúng tôi sử dụng 2 tập dữ liệu văn bản lớn để làm thực nghiệm. Tập RCV1 được tiền xử lý bởi (Bottou & Boussquet, 2008) theo mô hình túi từ (bag-of-words), bao gồm 781265 văn bản cho tập huấn luyện và 23149 văn bản cho tập kiểm tra, với 47152 từ, được gán nhãn ±1.

Tập dữ liệu twitter được lấy từ (Go *et al*., 2009), bao gồm 1600000 ý kiến (800000 thuộc lớp dương và 800000 thuộc lớp âm). Chúng tôi sử dụng công cụ BoW (McCallum, 1998) để tiền xử lý và biểu diễn các ý kiến theo mô hình túi từ thu được 244895 từ khác nhau. Sau đó chúng tôi chia ngẫu nhiên 1066667 ý kiến cho tập huấn luyện và 533333 cho tập kiểm tra.

Để so sánh tốc độ hội tụ của SVM-SGD sử dụng hàm hinge loss (hinge), smooth hinge loss (shinge) và logit loss (logit), chúng tôi thực hiện huấn luyện các mô hình với 200 epochs, mỗi epochs, theo dõi tỷ lệ lỗi dựa trên tập kiểm tra của các mô hình theo từng epoch.

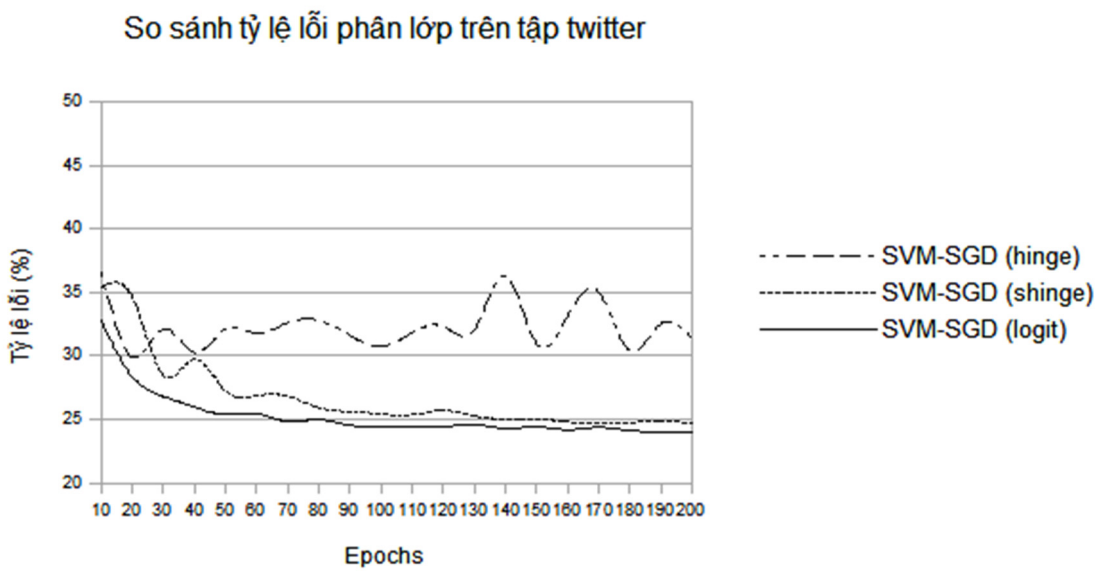
Kết quả phân lớp thu được trên tập RCV1 của các mô hình được trình bày trong Hình 3. Quan sát đồ thị, chúng ta có thể thấy rằng SVM-SGD (shinge) và SVM-SGD (logit) giảm tỷ lệ lỗi phân lớp ổn định khi tăng số epochs huấn luyện của các mô hình. Trong khi đó, SVM-SGD (hinge) thì giảm tỷ lệ lỗi không được nhanh và thiếu ổn định khi tăng số lượng epochs. Thời gian huấn luyện của SVM-SGD (hinge), SVM-SGD (shinge) và SVM-SGD (logit) tương ứng là 1.95, 1.85 và 2.35 giây.



Hình 3: Tỷ lệ lỗi phân lớp trên tập RCV1 của các mô hình theo số lượng epochs

Với tập dữ liệu twitter, kết quả phân lớp thu được từ các mô hình như trong Hình 4. Quan sát đồ thị, một lần nữa, chúng ta có thể thấy rằng SVM-SGD (shinge) và SVM-SGD (logit) giảm tỷ lệ lỗi phân lớp rất hiệu quả khi tăng số epochs huấn luyện của các mô hình. Khi tăng số lượng epochs, SVM-SGD (hinge) giảm tỷ lệ lỗi phân lớp chậm và không ổn định khi so sánh với 2 mô hình sử dụng hàm xấp xỉ khả vi. Thời gian huấn luyện của SVM-SGD (hinge), SVM-SGD (shinge) và SVM-SGD (logit) tương ứng là 1.94, 2.06 và 2.25 giây.

Với các kết quả phân lớp này, chúng tôi có thể tin rằng mô hình SVM sử dụng hàm xấp xỉ khả vi của hinge loss cho phép cải thiện hiệu quả phân lớp tập dữ liệu lớn khi giải trực tiếp bằng giải thuật SGD.



Hình 4: Tỷ lệ lỗi phân lớp trên tập RCV1 của các mô hình theo số lượng epochs

4 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Để máy học SVM có thể phân lớp nhanh, chính xác các tập dữ liệu lớn, giải pháp hiệu quả là sử dụng giải thuật SGD để giải trực tiếp vấn đề tối ưu của mô hình SVM. Tuy nhiên, SVM sử dụng hàm hinge loss không khả vi, là nguyên nhân làm ảnh hưởng đến tốc độ hội tụ đến lời giải của SGD. Chúng tôi đã đề xuất thay thế các hàm xấp xỉ khả vi của hinge loss trong mô hình SVM, nhằm cải thiện tốc độ hội tụ của SVM-SGD. Kết quả thực nghiệm trên 2 tập dữ liệu văn bản lớn RCV1, twitter cho thấy sự hiệu quả của đề xuất. SVM-SGD sử dụng (shinge hay logit) có thể phân lớp hàng triệu văn bản chỉ trong 2 giây trên một máy PC (Intel 3GHz, 4GB RAM) chạy hệ điều hành Linux (Fedora Core 20).

Trong tương lai, chúng tôi tiếp tục nghiên cứu các hàm xấp xỉ khả vi khác của hinge loss. Chúng tôi sẽ phát triển giải thuật SVM-SGD song song cho phép tăng tốc quá trình thực thi trên máy tính có nhiều bộ xử lý, nhóm hay lưới máy tính.

TÀI LIỆU THAM KHẢO

1. Boser, B., Guyon, I., Vapnik, V., “An training algorithm for optimal margin classifiers”, In proceedings of 5th ACM Annual Workshop on Computational Learning Theory, pp.144-152, 1992.
2. Bottou, L., Bousquet, O.: The tradeoffs of large scale learning. In Advances in Neural Information Processing Systems (20):161-168 (2008).
3. Breiman, L., “Arcing classifiers”, The annals of statistics, vol. 26, no. 3, pp.801-849, 1998.
4. Chang, C. C., Lin, C. J., “LIBSVM: a library for support vector machines”, ACM Transactions on Intelligent Systems and Technology, vol. 2, no. 27, pp.1-27, 2011 http://www.csie.ntu.edu.tw/~cjlin/libsvm
5. Cristianini, N., Shawe-Taylor, J., “An Introduction to Support Vector Machines: And Other Kernel-based Learning Methods”, Cambridge University Press, New York, NY, USA, 2000.
6. Do, T.N., “Parallel multiclass stochastic gradient descent algorithms for classifying million images with very-high-dimensional signatures into thousands classes”, Vietnam J. Computer Science, vol. 1, no. 2, pp.107-115, 2014.
7. Do, T. N., Nguyen, V. H., Poulet, F., “Speedup SVM algorithm for massive classification tasks”, In Proceedings of ADMA, pp.147-157, 2008.
8. Do,T.N., Fekete, J.D., “Large scale classification with support vector machine algorithms. In The Sixth International Conference on Machine Learning and Applications, ICMLA 2007, Cincinnati, Ohio, USA, pp.7-12, 2007.
9. Do, T.N., Poulet, F., “Classifying one billion data with a new distributed svm algorithm”, In proceedings of 4th IEEE Intl. Conf. on Computer Science, Research, Innovation and Vision for the Future, IEEE Press, pp.59-66, 2006.
10. Do, T.N., Poulet, F., “Mining very large datasets with svm and visualization”, In proceedings of 7th Intl. Conf. on Entreprise Information Systems, pp.127-134, 2005.
11. Freund, Y., Schapire, R., “A short introduction to boosting”, Journal of Japanese Society for Artificial Intelligence, vol. 14, no. 5, pp.771-780, 1999
12. Go, A., Bhayani, R., Huang, L.: Twitter sentiment (May 12th 2014 (accessed date)) http://help.sentiment140.com
13. Guyon, I., Web page on svm applications, 1999, http://www.clopinet.com/isabelle/Projects/-SVM/app-list.html
14. Liu H., Syed, N. and K. Sung. Incremental learning with support vector machines. ACM SIGKDD, 1999.
15. McCallum, A.: Bow: A Toolkit for Statistical Language Modeling, Text Retrieval, Classification and Clustering. 1998. http://www-2.cs.cmu.edu/~mccallum/bow
16. Mangasarian O.L.: Mathematical Programming for Support Vector Machines. INRIA Rocquencourt, France July 17 (2001).
17. Osuna, E., Freund, R., Girosi, F., “An improved training algorithm for support vector machines”, Neural Networks for Signal Processing VII, J. Principe, L. Gile, N. Morgan, and E. Wilson Eds, pp.276-285, 1997.
18. Platt J.: Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines. Microsoft Research Technical Report MSR-TR-98-14 (1998).
19. Poulet, F., Do, T.N., “Mining very large datasets with support vector machine

algorithms", Enterprise Information Systems V, O. Camp, J. Filipe, S. Hammoudi and M. Piattini Eds., pp.177-184, 2004.

20. Rennie, J.D.M.: Derivation of the f-measure. http://people.csail.mit.edu/jrennie/writing (February 2004).
21. Shalev-Shwartz, S., Singer, Y., Srebro, N., "Pegasos: Primal estimated sub-gradient solver for svm", In Proceedings of the Twenty-Fourth International Conference Machine Learning, ACM, pp.807-814, 2007
22. Suykens, J., Vandewalle, J. "Least squares support vector machines classifiers", Neural Processing Letters, vol. 9, no. 3, pp.293–300, 1999.
23. Tong, S., Koller, D., "Support vector machine active learning with applications to text classification", In proceedings of the 17th Intl. Conf. on Machine Learning, ACM, pp. 999-1006, 2000.
24. Vapnik, V.: The Nature of Statistical Learning Theory. Springer-Verlag (1995).