Evolutionary multiple instance boosting framework for weakly supervised learning

Kamanasish Bhattacharjee¹ · Millie Pant¹ · Shilpa Srivastava²

Received: 31 December 2020 / Accepted: 10 July 2021 / Published online: 13 August 2021
© The Author(s) 2021

Abstract

Multiple instance boosting (MILBoost) is a framework which uses multiple instance learning (MIL) with boosting technique to solve the problems regarding weakly labeled inexact data. This paper proposes an enhanced multiple boosting framework—evolutionary MILBoost (EMILBoost) which utilizes differential evolution (DE) to optimize the combination of weak classifier or weak estimator weights in the framework. A standard MIL dataset MUSK and a binary classification dataset Hastie_10_2 are used to evaluate the results. Results are presented in terms of bag and instance classification error and also confusion matrix of test data.

Keywords Differential evolution (DE) · Multiple instance learning (MIL) · Boosting · MILBoost

Introduction

Multiple Instance Learning (MIL) is a type of weak supervision. It falls under the inexact supervision category of weak supervision where the data are given with labels but not as exact as desired. This type of data is prevalent in medical field where the class labels are often not available at desired granularity [1]. Hence, MIL is particularly well suited for medical data analysis [2].

MILBoost was first proposed by Viola et al. [3]. It was mainly developed for object detection in images and videos. There on, MILBoost and many of its variants are used for various tasks—human action recognition [4], MIL with gradient boosting for face recognition from videos [5], human detection from artificially generated 3D human models [6], multi-class MILBoost for human parts detection [7], logistic MILBoost for pedestrian detection [8], gentle MILBoost for human detection which uses the Newton update to get an optimal weak classifier [9], confidence rated MILBoost [10], online MILBoost [11], object tracking by incorporating instance significance estimation into online MILBoost

[12], online MILBoost for object tracking [13–15], visual object tracking. In medical applications, MILBoost has been used for early temporal prediction of Type 2 diabetes risk condition [16], liver cirrhosis classification using ultrasound images [17], histopathology cancer image classification, segmentation, clustering [18–20]. The main concept behind boosting is to sequentially train several weak classifiers or weak estimators and combine them to form a strong classifier. This combining is done through weighted sum of the weak classifiers. For this, each weak classifier is assigned a weight. The main task here is to find the combination of optimized weights to generate the strongest classifier. MILBoost uses AnyBoost framework [21] where boosting classifier is trained by maximizing the log-likelihood of all bags. There is a scope of improving the MILBoost framework by enhancing the weight optimization process through population-based evolutionary technique instead of a single point gradient descent technique. This also opens up the scope for parallelizing the optimization process. Evolutionary algorithms—Genetic algorithm (GA) [22] and Differential Evolution (DE) [23] has been used for MIL to formulate pooling functions [24, 25].

The main objective of this work is to formulate a MILBoost framework based on differential evolution (DE) which will make the framework able to parallelize the optimization process.

The rest of the paper is divided into six sections. Section 2 elaborates on MILBoost. Section 3 gives a brief description

✉ Millie Pant
pant.milli@as.iitr.ac.in

¹ Department of Applied Science and Engineering, Indian Institute of Technology Roorkee, Uttarakhand 247667, India

² Christ University, Ghaziabad, Uttar Pradesh 201003, India

of DE. Section 4 presents the methodology. Section 5 discusses the experiments done and the subsequent results are discussed in Sect. 6. Finally, Sect. 7 concludes the paper.

Multiple instance boosting (MILBoost)

This section presents the formal representation of MILBoost. Suppose we have a binary classification data $(X_1, Y_1), (X_2, Y_2), \dots, (X_n, Y_n)$ where $X_i = \{x_{i1}, x_{i2}, \dots, x_{im}\}$, $i \in \{1, 2, \dots, n\}$, n is number of bags, m is the dimension of X_i and $Y_i \in [0, 1]$. $Y_i = 1$ indicates that the positive bag X_i contains at least one positive instance x_{ij} ($j = \{1, 2, \dots, m\}$). $Y_i = 0$ means that there are no positive instances in the bag X_i . The task is to identify a real-valued function $h(x_{ij})$ to infer the instance label y_{ij} corresponding to an instance x_{ij} . This function is estimated through a weak classifier. Then, through boosting, weak classifiers are combined to form a strong classifier with low error

$$H = \sum_{k=1}^K \alpha_k y_{ij}^k = \sum_{k=1}^K \alpha_k h_k(x_{ij}) \quad (1)$$

where K is the number of weak classifiers, α_k are the classifier weights or estimator weights which signify the relative importance of a weak classifier. In each phase, incorrectly classified instances receive more weights.

In MILBoost, the probability of an instance being positive is

$$p_{ij} = \frac{1}{1 + \exp(-y_{ij})} \quad (2)$$

The probability that a bag is positive is

$$p_i = 1 - \prod_{j=1}^m (1 - p_{ij}) \quad (3)$$

The log-likelihood of all bags is

$$L = \sum_{i=1}^n (y_i \log(p_i) + (1 - y_i) \log(1 - p_i)) \quad (4)$$

The main task is to train the classifier by maximizing this log-likelihood function.

Differential evolution (DE)

Differential evolution (DE) is a population-based evolutionary metaheuristic technique; used for solving the complex structured optimization problem in many application areas. DE was initially proposed by Storn and Price [23] in 1996. For a more profound understanding, article [26] could be referred. In general, DE formulation is divided into two phases—initialization and evolution. Initialization phase comprises random population generation, and evolution phase consists of mutation, crossover and selection for generating the new population for next generation. The flowchart for DE is presented in Fig. 1.

Fig. 1 Differential Evolution Flowchart

Operations in DE

Initialization

In this step, a set of the uniformly distributed random population is generated. These represent the initial solution points in the search space.

$$X_G = (X_1, X_2, \dots, X_N) \quad (5)$$

$$X_i = (x_{1i}, x_{2i}, \dots, x_{Di}) \quad (6)$$

$$x_{ji} = lb + r_{ji} * (ub - lb) \quad (7)$$

where G is the number of generations, NP is the number of individuals in population, D is dimension of an individual, lb and ub are lower and upper bounds respectively, $r \in [0, 1]$ is random number, $i \in \{1, 2, \dots, NP\}$, $j \in \{1, 2, \dots, D\}$.

Mutation

After population generation, mutation is performed to expand the search space. In mutation strategy, for each target vector a corresponding mutant vector is generated. DE has various mutation strategies. In this paper, the DE/rand/1 strategy is used to generate mutant vector $V_i = (v_{1i}, v_{2i}, \dots, v_{Di})$

$$V_i = X_{r_1} + F * (X_{r_2} - X_{r_3}) \quad (8)$$

where V_i is mutant vector, $F \in (0, 1.2]$ is scaling factor, X are individuals in population and $r_1, r_2, r_3 \in \{1, 2, \dots, NP\}$ where $r_1 \neq r_2 \neq r_3 \neq i$.

Crossover

Crossover is performed between target vector and mutant vector to increase the diversity of the population and to assimilate the best individual. After the crossover, trial vectors are generated. For a trial vector $U_i = (u_{1i}, u_{2i}, \dots, u_{Di})$ -

$$u_{ji} = \begin{cases} v_{ji}, & \text{if } r_{ji} \leq CR \cup j = j_r \\ x_{ji}, & \text{otherwise} \end{cases} \quad (9)$$

where $CR \in [0, 1]$ is crossover probability, $r \in [0, 1]$ is random number and $j_r \in \{1, 2, \dots, D\}$.

Selection

Tournament selection is done between the trial and the target vector and the one having a better fitness value move on to the next generation.

$$X_{i,G+1} = \begin{cases} U_{i,G}, & \text{if } f(U_{i,G}) \leq f(X_{i,G}) \\ X_{i,G}, & \text{otherwise} \end{cases} \quad (10)$$

where $f(\cdot)$ is the objective function.

Methodology

DE is used in the MILBoost framework to optimize the log-likelihood of all bags as defined in Eq. (4). So, the objective function for DE in this work is the log-likelihood function. A population of α_k , the classifier weights as defined in Eq. (1) is randomly initialized. The algorithm for the proposed Evolutionary MILBoost (EMILBoost) is presented below while it is pictorially represented through flowchart in Fig. 2.

Fig. 2 Evolutionary multiple instance boosting flowchart

Algorithm: Evolutionary MILBoost (EMILBoost)

START

Input: Training data $(X_i, Y_i)_{i=1}^n, X_i = \{x_{ij}\}, Y_i = \{0, 1\}, i = 1, \dots, n, j = 1, \dots, m$

Initialize: DE parameters F, CR, NP, G , mutation strategy

K = number of weak classifiers

$$\alpha_k^{pop} = \begin{pmatrix} \alpha_1^1, & \dots, & \alpha_K^1 \\ \vdots & \dots & \vdots \\ \alpha_1^{NP}, & \dots, & \alpha_K^{NP} \end{pmatrix}$$

for $g = 1$ to G

$$H_g^{pop} = \sum_{k=1}^K \alpha_k^{pop} y_{ij}^k = \sum_{k=1}^K \alpha_k^{pop} h_k(x_{ij})$$

$$y_{ij} = h(x_{ij})$$

$$p_{ij} = \frac{1}{1 + \exp(-y_{ij})}$$

$$p_i = 1 - \prod_{j=1}^m (1 - p_{ij})$$

$$L(H)^{pop} = \sum_{i=1}^n (y_i \log(p_i) + (1 - y_i) \log(1 - p_i))$$

$$\text{Mutate } \alpha_k^{pop} \text{ to generate mutant } \alpha V_k^{pop} = \begin{pmatrix} \alpha V_1^1, & \dots, & \alpha V_K^1 \\ \vdots & \dots & \vdots \\ \alpha V_1^{NP}, & \dots, & \alpha V_K^{NP} \end{pmatrix}$$

Perform crossover between α_k^{pop} and αV_k^{pop} to generate trail population

$$\alpha U_k^{pop} = \begin{pmatrix} \alpha U_1^1, & \dots, & \alpha U_K^1 \\ \vdots & \dots & \vdots \\ \alpha U_1^{NP}, & \dots, & \alpha U_K^{NP} \end{pmatrix}$$

Compute log-likelihood for trail population $L(H_u)^{pop}$

Perform selection between α_k^{pop} and αU_k^{pop} based on $L(H)^{pop}$ and $L(H_u)^{pop}$

end for

Select the best α_k

$$\text{Final strong classifier } H = \sum_{k=1}^K \alpha_k y_{ij}^k = \sum_{k=1}^K \alpha_k h_k(x_{ij})$$

STOP

As mentioned earlier, DE paves the path for parallelization of the optimization process. Unlike general optimization techniques, DE—a metaheuristic process approaches the optimal value from various directions. It generates multiple values in the search space as initial solutions and then converges towards the optimal point in the search space. Hence, rather than approaching from a single point, as DE approaches the problem from various directions, naturally parallelization will improve the process.

Experiments

Data

For this work, two classic MIL benchmark datasets are used—*MUSK1* and *MUSK2* [27] which are available in UCI Machine Learning Repository [1]. These correspond to the problem of predicting drug activity. A molecule has the desired drug effect if and only if one or more of its conformations bind to the target binding site. Since molecules can adopt multiple shapes, a bag is made up of shapes belonging to the same molecule. *MUSK1* and *MUSK2* contain 476 and 6598 instances respectively. *MUSK2* is used for training data as it contains greater number of instances. *MUSK1* is used as testing data. Both the datasets have total 168 attributes out of which 166 are features. The data attribute information is given in Table 1.

Apart from the aforementioned datasets, a standard binary classification dataset used to test boosting frameworks—Hastie_10_2 is used in this book [28] which is available in the scikit-learn dataset library [29]. The Hastie_10_2 dataset has 10 attributes $X_1, X_2, \dots, X_{10}$ which are standard independent Gaussian variates. The class is defined as

$$y = \begin{cases} 1 & \text{if } \sum_{i=1}^{10} X_i^2 > 9.34 \\ 0 & \text{otherwise} \end{cases} \quad (11)$$

Experimental setup

Decision Tree classifiers with maximum depth of 1 are used as the weak classifiers. Log-sum-exp pooling is used for bag pooling. For, implementation smoothness, the negative of

the log-likelihood function is taken and minimized. This is the same as maximizing the original function. Number of weak classifiers are varied to check its effects on training i.e. $K \in \{10, 15, 20, 25, 30, 35, 40\}$. DE parameters—Crossover probability = 0.7, Mutation strategy = best/1/bin, Scaling factor = [0.5, 1], number of generations = 1000, population size = 20.

Hardware and software specifications

Experiments have been conducted on Spyder 4.2.0 Integrated Development Environment (IDE) with Python 3.7.9 through Anaconda distribution on an Intel Xeon 2.5 GHz system with 16 GB RAM, Nvidia Quadro 2000 GPU and 64-bit Windows 10 Operating System.

Evaluation metrics

As this is a classification problem, hence the standard training error, testing error and confusion matrix of testing data are used as the evaluation metrics here. In MIL, we evaluate our model on the basis of bag classification accuracy. Therefore, bag training error, bag testing error are used here.

Results and discussions

The results of the proposed EMILBoost is compared with another two boosting frameworks—GentleBoost and LogitBoost [30]. Tables 2 and 3 records the testing and training

Table 2 Bag test errors

Number of weak classifiers	GentleBoost	LogitBoost	EMILBoost
10	0.597826	0.304348	0.206522
15	0.576087	0.282609	0.184783
20	0.445652	0.206522	0.152174
25	0.336957	0.195652	0.152174
30	0.293478	0.184783	0.141304
35	0.228261	0.184783	0.119565
40	0.217391	0.119565	0.119565

Best value in each row is highlighted in bold

Table 1 Data description

1	molecule_name	Symbolic name of each molecule. Ex. MUSK-188, NON-MUSK-jp13 etc. (Not used in predicting class)
2	conformation_name	Symbolic name of each conformation. Ex. MOL_ISO + CONF, where MOL = molecule number, ISO = stereoisomer number, CONF = conformation number. (Not used in predicting class)
3	f1-f162	Distance features
4	f163	OXY-DIS = Distance of the oxygen atom in the molecule to a designated point in 3-space
5	f164	OXY-X = X-displacement from the designated point
6	f165	OXY-Y = Y-displacement from the designated point
7	f166	OXY-Z = Z-displacement from the designated point
8	Class	0 = Non-MUSK, 1 = MUSK

Table 3 Bag train errors

Number of weak classifiers	GentleBoost	LogitBoost	EMILBoost
10	0.323529	0.215686	0.245098
15	0.254902	0.205882	0.156863
20	0.235294	0.176471	0.137255
25	0.235294	0.196078	0.127451
30	0.254902	0.176471	0.137255
35	0.196078	0.186275	0.127451
40	0.147059	0.196078	0.107843

Best value in each row is highlighted in bold

errors for bags for the MUSK dataset. Bag testing and training error rate curves or convergence curves for different boosting frameworks are presented in Figs. 3 and 4 for MUSK and Hastie_10_2 datasets respectively. The training and testing error rate curves for different numbers of weak

classifiers are presented in Figs. 5 and 6 for MUSK and Hastie_10_2 datasets respectively while Confusion matrix for different numbers of weak classifiers are presented in Fig. 7 for MUSK dataset.

From Tables 2 and 3, it is clear that EMILBoost achieves the lowest errors and hence outperforms GentleBoost and LogitBoost. Figures 3 and 4 also establishes the supremacy of EMILBoost.

From Fig. 5 and 6, it can be easily inferred that increasing the number of weak classifiers improve the learning process i.e. corresponds to lesser error.

The upper left block of confusion matrix signifies the True Positives (TP), lower right signifies True Negatives (TN) while lower left signifies False Positives (FP) and upper right signifies False Negatives (FN). Main aim of a classifier is to obtain more numbers of TP + TN and lesser numbers of FP + FN. From Fig. 7, it can be easily inferred that for EMILBoost framework, $TP + TN > FP + FN$. Hence, the framework is performing as desired.

Fig. 3 Error rate curves (MUSK) **Fig. 4** Error rate curves (Hastie_10_2)

Fig. 5 Error rates for strong classifier consisting of **a** 10 weak classifiers, **b** 15 weak classifiers, **c** 20 weak classifiers, **d** 25 weak classifiers, **e** 30 weak classifiers, **f** 35 weak classifiers, **g** 40 weak classifiers for MUSK dataset

Fig. 6 Error rates for strong classifier consisting of **a** 10 weak classifiers, **b** 15 weak classifiers, **c** 20 weak classifiers, **d** 25 weak classifiers, **e** 30 weak classifiers, **f** 35 weak classifiers, **g** 40 weak classifiers for Hastie_10_2 dataset

Fig. 7 Confusion Matrix for strong classifier consisting of **a** 10 weak classifiers, **b** 15 weak classifiers, **c** 20 weak classifiers, **d** 25 weak classifiers, **e** 30 weak classifiers, **f** 35 weak classifiers, **g** 40 weak classifiers for MUSK dataset

Conclusion

The main aim of this paper was to enhance the MILBoost framework through DE, a population-based evolutionary metaheuristic method by optimizing the weak classifier weights. DE also paves the path to parallelize this optimization process. The results show that the proposed EMILBoost outperforms GentleBoost and LogitBoost. Increasing the number of weak classifiers improves the learning process while on the other hand it increases the learning time. A trade-off between these two is needed through optimizing the number of weak classifiers which is a multi-objective problem. This can be regarded as the future extension of this work.

Funding Not applicable.

Availability of data and material Dua and Graff [1].

Code availability Not applicable.

Declarations

Conflict of interest The authors declare that there is no conflict of interest or no competing interests regarding the publication of this article.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Dua D, Graff C (2019) UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science. <http://archive.ics.uci.edu/ml>. Accessed 20 Nov 2020
- Quelleguez G, Cazuguel G, Cochener B, Lamard M (2017) Multiple-instance learning for medical image and video analysis. *IEEE Rev Biomed Eng* 10:213–234. <https://doi.org/10.1109/RBME.2017.2651164>
- Viola P, Platt JC, Zhang C (2005) Multiple Instance Boosting for Object Detection. In: *Advances in Neural Information Processing Systems*. pp 1417–1424
- Zhu S, Song D (2014) Human action recognition based on multiple instance learning. *J Appl Sci* 14:2276–2284. <https://doi.org/10.3923/jas.2014.2276.2284>
- Wohlhart P, Köstinger M, Roth PM, Bischof H (2011) Multiple instance boosting for face recognition in videos. *Lecture notes in computer science (including subseries Lecture notes in artificial intelligence and lecture notes in bioinformatics)*. Springer, Berlin, Heidelberg, pp 132–141
- Yamauchi Y, Fujiyoshi H (2011) Automatic generation of training samples and a learning method based on advanced MILBoost for human detection. In: *1st Asian Conference on Pattern Recognition*, ACPR 2011. pp 603–607
- Chen YT, Chen CS, Hung YP, Chang KY (2009) Multi-class multi-instance boosting for part-based human detection. In: *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops 2009*. pp 1177–1184
- Pang J, Huang Q, Jiang S, Gao W (2008) Pedestrian detection via logistic multiple instance boosting. In: *Proceedings - International Conference on Image Processing, ICIP*. pp 1464–1467
- Shen J, Yang W, Sun C (2013) Real-time human detection based on gentle MILBoost with variable granularity HOG-CSLBP. *Neural Comput Appl* 23:1937–1948. <https://doi.org/10.1007/s00521-012-1153-5>
- Ali K, Saenko K (2014) Confidence-rated multiple instance boosting for object detection. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, pp 2433–2440
- Qi Z, Xu Y, Wang L, Song Y (2011) Online multiple instance boosting for object detection. *Neurocomputing* 74:1769–1775. <https://doi.org/10.1016/j.neucom.2011.02.011>
- Liu J, Lu Y, Zhou T (2016) Instance significance guided multiple instance boosting for robust visual tracking. In: *Proceedings—International Conference on Image Processing, ICIP*. IEEE Computer Society, pp 1694–1698
- Zhang K, Song H (2013) Real-time visual tracking via online weighted multiple instance learning. *Pattern Recognit* 46:397–411. <https://doi.org/10.1016/j.patcog.2012.07.013>
- Babenko B, Yang MH, Belongie S (2011) Robust object tracking with online multiple instance learning. *IEEE Trans Pattern Anal Mach Intell* 33:1619–1632. <https://doi.org/10.1109/TPAMI.2010.226>
- Babenko B, Yang M-H, Belongie S (2010) Visual tracking with online Multiple Instance Learning. In: *IEEE Conference on Computer Vision and Pattern Recognition*. Institute of Electrical and Electronics Engineers (IEEE), pp 983–990
- Bernardini M, Moretti M, Romeo L et al (2020) Early temporal prediction of type 2 diabetes risk condition from a general practitioner electronic health record: a multiple instance boosting approach. *Artif Intell Med* 105:101847. <https://doi.org/10.1016/j.artmed.2020.101847>
- Fujita Y, Mitani Y, Hamamoto Y, et al (2016) Training ROI selection based on MILBoost for liver cirrhosis classification using ultrasound images. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer, pp 451–459
- Xu Y, Zhu JY, Chang E, Tu Z (2012) Multiple clustered instance learning for histopathology cancer image classification, segmentation and clustering. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. pp 964–971
- Xu Y, Zhu JY, Chang EIC et al (2014) Weakly supervised histopathology cancer image segmentation and classification. *Med Image Anal* 18:591–604. <https://doi.org/10.1016/j.media.2014.01.010>
- Li W, Zhang J, McKenna SJ (2015) Multiple instance cancer detection by boosting regularised trees. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial*

- Intelligence and Lecture Notes in Bioinformatics). Springer, pp 645–652
21. Mason L, Bartlett P, Baxter J, Frean M (1999) Boosting algorithms as gradient descent. In: *Advances in Neural Information Processing Systems*. pp 512–518
  22. Holland JH (1992) Genetic algorithms. *Sci Am* 267:66–73. <https://doi.org/10.2307/24939139>
  23. Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J Glob Optim* 11:341–359. <https://doi.org/10.1023/A:1008202821328>
  24. Bhattacharjee K, Pant M, Zhang Y-D, Satapathy SC (2020) Multiple instance learning with genetic pooling for medical data analysis. *Pattern Recognit Lett*. <https://doi.org/10.1016/j.patrec.2020.02.025>
  25. Bhattacharjee K, Tiwari A, Pant M, Ahn CW (2020) A pooling function based on differential evolution for multiple instance learning. In: *Proceedings of 9th International Conference on Smart Media and Applications (SMA 2020)*. Jeju, South Korea
  26. Bilal PM, Zaheer H et al (2020) Differential evolution: a review of more than two decades of research. *Eng Appl Artif Intell* 90:103479. <https://doi.org/10.1016/j.engappai.2020.103479>
  27. Dietterich TG, Lathrop RH, Lozano-Pérez T (1997) Solving the multiple instance problem with axis-parallel rectangles. *Artif Intell* 89:31–71. [https://doi.org/10.1016/s0004-3702\(96\)00034-3](https://doi.org/10.1016/s0004-3702(96)00034-3)
  28. Hastie T, Tibshirani R, Friedman J (2009) *The Elements of statistical learning*. Springer, New York,
  29. Hastie_10_2. https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_hastie_10_2.html. Accessed 5 Mar 2021
  30. Friedman J, Hastie T, Tibshirani R (2000) Additive logistic regression: a statistical view of boosting. *Ann Stat* 28:337–407

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.